## संदर्भ संहिता

## Reference Manual

# उच्च संकाय प्रशिक्षण केंद्र

# Centre for Advanced Faculty Training

# कृषि में जैव सूचना विज्ञान डेटा विश्लेषण के लिए सांख्यिकीय और कम्प्यूटेशनल प्रगति: प्रायोगिक स्वरूप

# Statistical and Computational Advances for Bioinformatics Data Analysis in Agriculture: Practical Aspects

CAFT Director: Dr. Rajender Parsad
Course Coordinator: Dr. Girish Kumar Jha
Course Co-Coordinator: Dr. Sudhir Srivastava
Course Co-Coordinator: Dr. Neeraj Budhlakoti

काफ्ट निदेशक: डॉ. राजेन्द्र प्रसाद
पाठ्यक्रम समन्वयक: डॉ. गिरीश कुमार झा
पाठ्यक्रम सह-समन्वयक: डॉ. सुधीर श्रीवास्तव
पाठ्यक्रम सह-समन्वयक: डॉ. नीरज बुढ़लाकोटी

Division of Agricultural Bioinformatics
ICAR-Indian Agricultural Statistics Research Institute
Library Avenue, PUSA, New Delhi - 110012
https://iasri.icar.gov.in/

# प्रस्तावना

भा.कृ.अनु.प.- भा.कृ.सां.अ.सं., सांख्यिकीय विज्ञान (सांख्यिकी, संगणक अनुप्रयोग और जैव सूचना विज्ञान) में प्रासंगिक कार्यों में कार्यरत एक प्रमुख संस्थान है और कृषि अनुसंधान की गुणवत्ता को समृद्ध करने और नीतिगत निर्णय लेने के लिए कृषि विज्ञान में इनके विवेकपूर्ण संलयन में इसका प्रमुख योगदान है। 1930 में अपनी स्थापना के बाद से, तत्कालीन इंपीरियल काउंसिल ऑफ एग्रीकल्चरल रिसर्च के एक छोटे सांख्यिकीय अनुभाग के रूप से, संस्थान राष्ट्रीय और अंतरराष्ट्रीय स्तर पर अपनी उपस्थिति दर्ज कराने में सक्षम हुआ। संस्थान बहुत सक्रिय रूप से शोधकर्ताओं को सलाहकार सेवाएँ प्रदान कर रहा है जिसने संस्थान को राष्ट्रीय कृषि अनुसंधान एवं शिक्षा प्रणाली और राष्ट्रीय कृषि सांख्यिकी प्रणाली दोनों में अपनी उपस्थिति दर्ज कराने में सक्षम हुआ है। संस्थान ने राष्ट्रीय कृषि अनुसंधान एवं शिक्षा प्रणाली में एक उच्च स्तरीय सांख्यिकीय संगणना पर्यावरण बनाने में अग्रणी भूमिका निभाई है।

जैव सूचना विज्ञान वस्तुतः जीव विज्ञान, कंप्यूटर विज्ञान और सांख्यिकी का अंतःविषय क्षेत्र है। पिछले दो दशकों के दौरान जैविक विज्ञान के क्षेत्र में बृहद डेटा उत्पन्न किया गया जिसमें सबसे पहले जीवों के जीनोम अनुक्रमण के विषय में जानकारी प्राप्त की गई। इसके उपरान्त इन प्राप्त जानकारियों को उच्च प्रशिक्षणात्मक तकनीक से जैव प्रौद्योगिकी अनुसंधान प्रयोगशालाओं में किये गये प्रयोगों तथा इसके प्रभावों की गतिशीलता का अध्ययन किया जा रहा है। जैविक अनुसंधान के क्षेत्र में विभिन्न जैवसूचना विज्ञान तकनीकों/ टूल्स के प्रयोग, डेटा की संचयन एवं पुनःप्राप्ति, विश्लेषण, एनोटेसन और परिणाम के प्रत्योक्षकरण, और संपूर्ण जैविक प्रणालियों को बेहतर ढंग से समझने में सहायक है। इससे टिकाऊ कृषि के लिए टूल्स और तकनीकों के विकास को बढ़ावा मिलेगा। संस्थान द्वारा आयोजित प्रशिक्षण कार्यक्रम शोधकर्ताओं के लिए कृषि जैव सूचना विज्ञान और कम्प्यूटेशनल जीव विज्ञान में हुई प्रगति को समझने में बहुत उपयोगी हैं।

प्रशिक्षण कार्यक्रम **कृषि में जैव सूचना विज्ञान डेटा विश्लेषण के लिए सांख्यिकीय और कम्प्यूटेशनल प्रगतिः प्रायोगिक स्वरूप** विशेष रूप से संकाय सदस्यों और साथी सहभागियों के बीच पारस्परिक विचार-विमर्श के माध्यम से अधिकतम शैक्षणिक लाभ प्राप्त करने के लिए तैयार किया गया है। मुझे विश्वास है कि इस प्रशिक्षण कार्यक्रम से प्राप्त ज्ञान सहभागियों को कम्प्यूटेशनल जीव विज्ञान और जैव सूचना विज्ञान की बेहतर समझ प्राप्त करने में सक्षम करेगा, जिससे उन्हें उपयुक्त टूल्स और सॉफ्टवेयर का उपयोग करके जैविक आँकरों को संभालने और विश्लेषण करने में भी लाभ होगा।

पाठ्यक्रम सामग्री सिद्धांत और अनुप्रयोग के मध्य समाहित हैं। विषयवस्तु विभिन्न मॉड्यूल के अंतर्गत रखे गए हैं: (1) कम्प्यूटेशनल टूल्स और तकनीकों की मूल बातें [ लिनक्स का परिचय; आर/पायथन/पर्ल प्रोग्रामिंग भाषा; कम्प्यूटेशनल जीवविज्ञान और जैव सूचना विज्ञान के लिए उपयुक्त तरीके/टूल/सॉफ्टवेयर/डेटाबेस ], (2) एनजीएस डेटा विश्लेषण के लिए कम्प्यूटेशनल तरीके [ डेटा प्री-प्रोसेसिंग; जीनोम असेंबली और एनोटेशन; ट्रांसक्रिप्टोमिक्स, मेटाजीनोमिक्स और नॉन-कोडिंग आरएनए डेटा का विश्लेषण; जीनोम-वाइड एसोसिएशन अध्ययन और जीनोमिक चयन ], और (3) प्रोटिओमिक्स डेटा विश्लेषण के लिए टूल और तकनीक [ प्रोटीन संरचना प्रेडिक्शन; आणविक डॉकिंग; आणविक गतिकी और सिमुलेशन; प्रोटिओमिक्स एक्सप्रेशन डेटा विश्लेषण ]।

इस पाठ्यक्रम में शामिल संकाय सदस्य जैव सूचना विज्ञान/कम्प्यूटेशनल जीवविज्ञान/कृषि सांख्यिकी/कंप्यूटर अनुप्रयोग/जीनोमिक्स और अन्य विषयों के क्षेत्र में सुस्थापित प्रतिष्ठित वैज्ञानिक हैं। संदर्भ संहिता में दिए गए व्याख्यान नोट्स विषय का विवरण प्रदान करते हैं। मुझे आशा है कि संदर्भ संहिता सहभागियों के लिए काफी उपयोगी होगी। होगी। मैं इस अवसर पर सभी संकाय सदस्यों को उत्कृष्ट कार्य करने के लिए धन्यवाद देता हूं। मैं इस महत्वपूर्ण दस्तावेज को समय पर प्रकाशित करने के लिए इस प्रशिक्षण कार्यक्रम के पाठ्यक्रम समन्वयक एवं प्रभागाध्यक्ष, कृषि जैवसूचना विज्ञान, डॉ. गिरीश कुमार झा और पाठ्यक्रम सह-समन्वयक, डॉ. सुधीर श्रीवास्तव और डॉ. नीरज बुढ़लाकोटी को अपनी शुभकामनाएं देता हूँ। इस संदर्भ संहिता को और बेहतर बनाने के लिए आप सभी के सुझावों का स्वागत है।

**(राजेन्द्र प्रसाद)**
**निदेशक, भा.कृ.अनु.प.-भा.कृ.सां.अ.सं.**

**नई दिल्ली**
**01 जनवरी, 2024**

# FOREWORD

ICAR-IASRI is a premier Institute of relevance in Statistical Sciences (Statistics, Computer Applications and Bioinformatics) and their judicious fusion in agricultural sciences for enriching quality of agricultural research and informed policy decision making. Ever since its inception in 1930, as a small Statistical Section of the then Imperial Council of Agricultural Research, the Institute has grown in stature and made its presence felt both nationally and internationally. The Institute has been very actively pursuing advisory service that has enabled the institute to make its presence felt both in National Agricultural Research and Education System (NARES) and National Agricultural Statistics System (NASS). The Institute has taken a lead in creating a high-end statistical computing environment in NARES.

Bioinformatics is an interdisciplinary field comprising of biology, statistics and computer science. During the last two decades enormous sequence data have been generated in biological science, firstly with the onset of sequencing the genomes of living organisms and, secondly, rapid application of high throughput experimental techniques in laboratory research. Application of various bioinformatics tools in biological research enables storage, retrieval, analysis, annotation and visualization of results, and promotes better understanding of biological systems in their entirety. This will further lead to development of tools and techniques for sustainable agriculture. The training programmes organized by the Institute are very useful in understanding the advances in agricultural bioinformatics and computational biology to the researchers.

The training programme **Statistical and Computational Advances for Bioinformatics Data Analysis in Agriculture: Practical Aspects** has been especially designed to derive the maximum academic advantage through interaction with faculty members and fellow participants. I am sure that the knowledge assimilated from this training programme will enable the participants to have better understanding of bioinformatics and computational biology, which will also benefit them in handling and analyzing the bioinformatics data by using appropriate tools and software.

The course contents are intertwining of theory and application. The topics are covered under different modules: (1) Basics of Computational Tools and Techniques [Introduction to Linux; R/Python/Perl Programming Languages; Methods/Tools/Software/Databases relevant to Bioinformatics], (2) NGS Data Analysis [NGS Data Pre-processing; Genome Assembly and Annotation; Analysis of Transcriptomics, Metagenomics and Non-coding RNA Data; Genome-Wide Association Studies and Genomic Selection], and (3) Proteomics Data Analysis [Protein Structure Prediction; Molecular Docking; Protein-Protein Interaction Network; Molecular Dynamics and Simulation; Proteomics Expression Data Analysis].

The faculty for this course comprises of eminent scientists well established in the field of Bioinformatics/ Computational Biology/ Agricultural Statistics/ Computer Applications/ Genomics and other disciplines. The lecture notes given in the reference manual provide an exposition of the subject. I hope that the reference manual will be quite useful to the participants. I take this opportunity to thank the entire faculty for doing a wonderful job. I wish to complement Course Coordinator & Head, Division of Agricultural Bioinformatics, Dr. Girish K. Jha and Course Co-coordinators, Dr. Sudhir Srivastava and Dr. Neeraj Budhlakoti of this training programme, for bringing out this valuable document in time. We look forward to suggestions from every corner in improving this reference manual.

**(Rajender Parsad)**
**Director, ICAR-IASRI**

**New Delhi**
**January 01, 2024**

# आमुख

भा.कृ.अनु.प. - भारतीय कृषि सांख्यिकी अनुसंधान संस्थान देश में कृषि सांख्यिकी, संगणक अनुप्रयोग और जैव सूचना विज्ञान विषयों में कार्य करने वाला एक प्रमुख संस्थान है। संस्थान परीक्षण अभिकल्पना, नमूनाकरण तकनीक, सांख्यिकीय आनुवंशिकी, पूर्वानुमान तकनीक, जैव सूचना विज्ञान और संगणक अनुप्रयोगों पर विशेष जोर देने के साथ कृषि सांख्यिकी में अनुसंधान, शिक्षण और प्रशिक्षण कार्यक्रम आयोजित करने में कार्यरत है। संस्थान बहुत सक्रिय रूप से परामर्श सेवा प्रदान कर रहा है जिसने संस्थान को राष्ट्रीय कृषि अनुसंधान एवं शिक्षा प्रणाली और राष्ट्रीय कृषि सांख्यिकी प्रणाली दोनों में अपनी उपस्थिति दर्ज कराने में सक्षम हुआ है। संस्थान ने कृषि अनुसंधान के लिए उपयोगी सांख्यिकीय सॉफ्टवेयर पैकेज विकसित करने में अग्रणी भूमिका निभाई है।

जैव सूचना विज्ञान वस्तुतः जीव विज्ञान, कंप्यूटर विज्ञान और सांख्यिकी का अंतःविषय क्षेत्र है। पिछले दो दशकों के दौरान जैविक विज्ञान के क्षेत्र में बृहद डेटा उत्पन्न किया गया जिसमें सबसे पहले जीवों के जीनोम अनुक्रमण के विषय में जानकारी प्राप्त की गई। इसके उपरान्त इन प्राप्त जानकारियों को उच्च प्रशिक्षणात्मक तकनीक से जैव प्रौद्योगिकी अनुसंधान प्रयोगशालाओं में किये गये प्रयोगों तथा इसके प्रभावों की गतिशीलता का अध्ययन किया जा रहा है। जैविक अनुसंधान के क्षेत्र में विभिन्न जैवसूचना विज्ञान तकनीकों/ टूल्स के प्रयोग, डेटा की संचयन एवं पुनःप्राप्ति, विश्लेषण, एनोटेसन और परिणाम के प्रत्योक्षकरण, और संपूर्ण जैविक प्रणालियों को बेहतर ढंग से समझने में सहायक है। इससे टिकाऊ कृषि के लिए टूल्स और तकनीकों के विकास को बढ़ावा मिलेगा। प्रशिक्षण कार्यक्रम का उद्देश्य सहभागियों को कृषि में जैव सूचना विज्ञान आँकड़ों के विश्लेषण के लिए सांख्यिकीय और कम्प्यूटेशनल दृष्टिकोण से परिचित कराना और अनुसंधान, शिक्षण और प्रशिक्षण में उनकी क्षमताओं को बढ़ाना है।

प्रशिक्षण जीनोमिक्स, ट्रांसक्रिप्टोमिक्स, मेटाजेनोमिक्स और प्रोटिओमिक्स डेटा के विश्लेषण में शामिल कम्प्यूटेशनल टूल और तकनीकों, सांख्यिकीय और कम्प्यूटेशनल दृष्टिकोण की मूल बातें पर केंद्रित है। कृषि जैवसूचना विज्ञान से संबंधित अवधारणाओं, मुद्दों और समाधानों पर विशेष जोर दिया गया है। इस प्रशिक्षण कार्यक्रम में विभिन्न व्याख्यान शामिल किए गए हैं: सुपर-कंप्यूटिंग सुविधा अशोका; लिनक्स और आर/ पायथन/ पर्ल प्रोग्रामिंग भाषाओं की मूल बातें; जैविक डेटाबेस; अनुक्रम और वंशावली विश्लेषण; एसएनपी और एसएसआर माइनिंग; एनजीएस डेटा विश्लेषण का परिचय; जीनोम असेंबली और एनोटेशन; ट्रांसक्रिप्टोमिक्स, मेटाजीनोमिक्स और नॉन-कोडिंग आरएनए डेटा का विश्लेषण; जीनोम-वाइड एसोसिएशन अध्ययन और जीनोमिक चयन; प्रोटीन संरचना प्रेडिक्शन; आणविक डॉकिंग; आणविक गतिकी और सिमुलेशन; प्रोटिओमिक्स एक्सप्रेशन डेटा विश्लेषण; पोस्ट-ट्रांसलेशनल संशोधन।

हम इस अवसर पर संस्थान के संकाय सदस्यों को धन्यवाद देना चाहते हैं जिन्होंने इस पाठ्यक्रम को सार्थक और सफल बनाने में अपना बहुमूल्य समय दिया जिससे इस संदर्भ संहिता को समय पर प्रकाशित करने में मदद मिली। हम इस प्रशिक्षण कार्यक्रम में अपने अधिकारियों को प्रतिनियुक्त करने के लिए विभिन्न भा.कृ.अनु.प. संस्थानों, राज्य कृषि विश्वविद्यालयों और ब्यूरो के भी आभारी हैं। हम डॉ. राजेंद्र प्रसाद, निदेशक, भा.कृ.अनु.प.-भा.कृ.सां.अ.सं. के बहुमूल्य मार्गदर्शन और पाठ्यक्रम के सुचारू संचालन के लिए सभी आवश्यक सुविधाएं उपलब्ध कराने के लिए अत्यंत आभारी हैं। हम उन सभी के आभारी हैं जिन्होंने इस प्रशिक्षण संदर्भ संहिता को तैयार करने के लिए प्रत्यक्ष या अप्रत्यक्ष रूप से सहयोग दिया है।

| | | |
|---|---|---|
| गिरीश झा | सुधीर श्रीवास्तव | नीरज |
| **(गिरीश कुमार झा)** | **(सुधीर श्रीवास्तव)** | **(नीरज बुढ़लाकोटी)** |
| **पाठ्यक्रम समन्वयक एवं** | **पाठ्यक्रम सह-समन्वयक** | **पाठ्यक्रम सह-समन्वयक** |
| **प्रभागाध्यक्ष, कृषि जैवसूचना विज्ञान** | | |
| **भा.कृ.अनु.प.-भा.कृ.सां.अ.सं.** | | |

# PREFACE

The ICAR-Indian Agricultural Statistics Research Institute is a premier Institute in the disciplines of Agricultural Statistics, Computer Applications and Bioinformatics in the country. The Institute has been engaged in conducting research, teaching and organizing training programmes in Agricultural Statistics with special emphasis on Experimental Designs, Sampling Techniques, Statistical Genetics, Forecasting Techniques, Bioinformatics and Computer Applications. The Institute has been very actively pursuing advisory service that has enabled the institute to make its presence felt both in National Agricultural Research and Education System (NARES) and National Agricultural Statistics System (NASS). The Institute has taken a lead in developing Statistical Software Packages useful for Agricultural Research.

During the last two decades enormous sequence data have been generated in biological science, firstly with the onset of sequencing the genomes of living organisms and, secondly, rapid application of high throughput experimental techniques in laboratory research. Application of various bioinformatics tools in biological research enables storage, retrieval, analysis, annotation and visualization of results, and promotes better understanding of biological systems in their entirety. This will further lead to development of tools and techniques for sustainable agriculture. The aim of the training programme is to familiarize the participants to statistical and computational approaches for bioinformatics data analysis in agriculture and in upgrading their capabilities in research, teaching and training.

The training focus on the basics of computational tools & techniques, statistical and computational approaches involved in the analysis of genomics, transcriptomics, metagenomics, and proteomics data. Special emphasis has been laid on concepts, issues and solutions related to agricultural bioinformatics. Various lectures were included in this training programme: Super-Computing Facility ASHOKA; Basics of Linux and R/Python/Perl Programming Languages; Biological Databases; Sequence and Phylogenetic Analysis; SNP and SSR Mining; Introduction to NGS Data Analysis; Genome Assembly and Annotation; Analysis of Transcriptomics, Metagenomics and Non-coding RNA Data; Genome-Wide Association Studies and Genomic Selection; Protein Structure Prediction; Molecular Docking; Molecular Dynamics and Simulation; Proteomics Expression Data Analysis; Post-Translational Modifications.

We would like to take this opportunity to thank the faculty of the Institute who spared their valuable time in making this course meaningful and successful that helped in bringing out this manual in time. We are also thankful to the various ICAR Institutes, State Agricultural Universities and Bureaus for deputing their employees in this training programme. We are grateful to Dr. Rajender Parsad, Director, ICAR-IASRI for his valuable guidance and making all necessary facilities available for smooth conduct of the course. We are thankful to each one who supported directly or indirectly for preparing this training manual.

गिरीश झा

**(Girish K. Jha)**
**Course Coordinator &**
**Head, Division of Bioinformatics**
**ICAR-IASRI**

सुधीर श्रीवास्तव

**(Sudhir Srivastava)**
**Course Co-Coordinator**

**(Neeraj Budhlakoti)**
**Course Co-Coordinator**

# CONTENTS

# Introduction to Bioinformatics

Girish Kumar Jha, Sneha Murmu, Soumya Sharma and Ritwika Das

ICAR-Indian Agricultural Statistics Research Institute, New Delhi

## History

- 1950-70s:

In the early 1950s, there was still controversy surrounding DNA's role as the carrier of genetic information. DNA's genetic role was firmly established in 1952 through the Hershey-Chase experiment. While the double-helix structure of DNA was revealed in 1953, it took more years to decipher the genetic code and develop DNA sequencing methods. Meanwhile, significant progress was made in protein analysis, especially with the publication of insulin's amino acid sequence in the late 1950s. This achievement spurred the development of protein sequencing methods, like the Edman degradation method, which allowed for automated sequencing of more than 15 protein families. However, a challenge with protein sequencing was assembling the complete sequence for large proteins, leading to the early development of bioinformatics software to address this issue.

Margaret Dayhoff, often referred to as the "mother and father of bioinformatics," was a physical chemist who recognized the potential of applying computational methods to biology and medicine. She collaborated with physicist Robert S. Ledley and together, in the late 1950s, they developed COMPROTEIN, one of the earliest bioinformatics software, for determining protein primary structure using Edman peptide sequencing data. They used this software to tackle the challenge of assembling complete sequences for large proteins, which was a significant computational problem. Dayhoff contributed to simplifying the handling of protein sequence data by developing the one-letter amino acid code, which is still in use today.

- The Birth of Sequence Databases:

Dayhoff and Eck's 1965 "Atlas of Protein Sequence and Structure" was the first biological sequence database. It contained 65 protein sequences, providing a basis for early computational analysis. Researchers began to consider the idea that protein sequences might reveal evolutionary history, similar to how language evolves, where the arrangement of letters conveys meaning.

- The Concept of Orthology:

Emile Zuckerkandl and Linus Pauling introduced the term "Paleogenetics" in 1963 to explore the evolutionary aspects of biomolecular sequences. They observed that orthologous proteins from different species showed varying degrees of similarity, correlating with their evolutionary divergence. Orthology, defined by Walter M. Fitch in 1970, described homology resulting from speciation events. This observation led to the hypothesis that orthologous proteins evolved from a common ancestor, and their sequences could be used to predict ancestral sequences and trace evolutionary history.

- Challenges in Sequence Alignment:

Initial efforts in sequence-based phylogenetic studies focused on closely related proteins that could be assessed visually for homology. However, for more distant or unequal-length protein sequences, visual comparison was impractical and often led to errors.

In 1970, Needleman and Wunsch developed the first dynamic programming algorithm for pairwise protein sequence alignments. Multiple sequence alignment (MSA) algorithms emerged in the early 1980s, addressing the challenge of aligning numerous sequences of different lengths more efficiently. In 1987, Da-Fei Feng and Russell F. Doolitle developed a practical approach to multiple sequence alignment (MSA) known as "progressive sequence alignment." Their method involved several steps:

- Performing a Needleman–Wunsch alignment for all possible sequence pairs.
- Extracting pairwise similarity scores from each of these pairwise alignments.
- Using these similarity scores to construct a guide tree, which represents the relationships between sequences.
- Aligning the sequences in a stepwise manner, starting with the two most similar sequences and then progressively adding the next most similar sequences according to the guide tree.

In 1988, the popular MSA software CLUSTAL was developed as a simplification of the Feng–Doolittle algorithm. CLUSTAL has remained in use and continued to be maintained up to the present day. This software made MSA more accessible and efficient, allowing researchers to align multiple sequences effectively.

➢ A Mathematical Framework for Amino Acid Substitutions (1978):

Margaret Dayhoff, Schwartz, and Orcutt developed the first probabilistic model of amino acid substitutions. The model was based on 1572 point accepted mutations (PAMs) in the phylogenetic trees of 71 protein families. They created a 20x20 asymmetric substitution matrix containing probability values based on observed amino acid mutations. This matrix introduced the concept of substitutions as a measurement of evolutionary change, shifting from the previous concept of evolutionary distance based on the least number of changes.

- Paradigm Shift from Protein to DNA Analysis (1970-1980):

Francis Crick's sequence hypothesis confirmed that DNA encodes information for proteins. DNA sequencing methods, including Maxam-Gilbert (1976) and Sanger's "plus and minus" method (1977), made DNA sequencing more accessible. The Sanger chain termination method (1977) remains in use today. DNA sequences could potentially provide information about all proteins in an organism. Manual tasks like comparisons, calculations, and pattern matching were more efficiently performed by computers.

➢ Development of Sequence Analysis Software (1979):

Roger Staden's software (1979) was one of the first to analyze Sanger sequencing reads. The software could search for overlaps, verify, edit, and join sequence reads, and annotate and manipulate sequence files. It introduced additional characters ("uncertainty codes") to record basecalling uncertainties in sequence reads. Staden's Package is still developed and maintained today.

➢ Using DNA Sequences in Phylogenetic Inference:

Early phylogenetic trees were reconstructed from protein sequences with a focus on maximum parsimony. Parsimony methods assumed minimal evolutionary changes but could fail with moderate to large changes. DNA sequences provided additional information, such as synonymous mutations. Joseph Felsenstein introduced maximum likelihood (ML) methods for phylogenetic tree inference from DNA sequences. ML estimation involved finding the tree with the highest probability of evolving the observed data. Bioinformatics tools and statistical methods based on ML and Bayesian statistics have been developed and are still in use today.

- Overcoming Technical Limitations in the Late 1970s:

The late 1970s faced technical limitations that needed addressing to broaden computer use in DNA analysis. The subsequent decade played a pivotal role in addressing these issues and advancing the field.

- Molecular Methods for Targeting and Amplifying Specific Genes:

Genes are less abundant and cannot be individually sequenced, as they are contiguous on DNA molecules and present in low copies per cell. A solution emerged when Jackson, Symons, and Berg (1972) used restriction endonucleases and DNA ligase to cut and insert circular SV40 viral DNA into lambda DNA. E. coli cells were transformed with this construct, and the inserted DNA was replicated and amplified in the host organism. This experiment pioneered the isolation and amplification of genes independently from their source organism. Concerns about ethical issues led to a moratorium on the use of recombinant DNA, and guidelines were established during the 1975 Asilomar conference.

- Invention of Polymerase Chain Reaction (PCR):

The polymerase chain reaction (PCR) was a significant development that allows DNA amplification without cloning procedures. The first description of "repair synthesis" using DNA polymerase was in 1971 by Kjell Kleppe et al. The invention of PCR is credited to Kary Mullis for his substantial optimizations, including the use of thermostable Taq polymerase and the development of the thermal cycler. Mullis patented the process and gained recognition for inventing PCR. Both gene cloning and PCR are widely used in DNA library preparation, critical for obtaining sequence data.

- DNA Sequencing and Bioinformatics in the 1980s:

The late 1970s saw the emergence of DNA sequencing, along with enhanced DNA manipulation techniques. DNA sequencing and manipulation led to increased availability of sequence data. Access to computers and bioinformatics software also grew during the 1980s, facilitating the analysis of sequence data.

- 1990-2000: Genomics, Structural Bioinformatics, and the Information Superhighway

- Dawn of the Genomics Era:

In 1995, the first complete genome sequencing of a free-living organism (Haemophilus influenzae) was achieved by The Institute for Genomic Research (TIGR), led by J. Craig Venter. The Human Genome Project, initiated in 1991 by the U.S. National Institutes of Health, aimed to sequence the human genome and cost $2.7 billion over 13 years. Celera Genomics led a private effort to sequence the human genome in competition with the publicly funded Human Genome Project, achieving it at one-tenth of the cost due to different experimental strategies.

- Challenges in Early Genomics:

Sequencing genomes was costly and time-consuming; for example, sequencing a human genome with 2018 technology would cost $1000 and take less than a week, but older methods were much slower. Specialized software was needed to handle the massive amount of sequencing data. Several Perl-based software tools were developed in the mid to late 1990s for assembling whole-genome sequencing reads.

- Emergence of the Internet:

The rise of the World Wide Web (WWW) in the mid-1990s revolutionized communication and enabled the creation of online bioinformatics resources. Nucleotide sequence databases like EMBL and GenBank became accessible online in the early 1990s. The National Center for Biotechnology Information (NCBI) made its website and tools, including BLAST, available online in 1994. Major databases such as Genomes (1995), PubMed (1997), and Human Genome (1999) were established and are still in use today.

- Structural Bioinformatics:

Advances allowed computers to predict protein secondary and tertiary structures with varying degrees of certainty. Molecular dynamics simulations became possible, although they required significant computational resources. The use of graphics processing units (GPUs) and supercomputers aided in making molecular dynamics simulations more accessible.

- 2000-2010: High-Throughput Bioinformatics

- Second-Generation Sequencing:

Second-generation sequencing (next-generation sequencing or NGS) began with the '454' pyrosequencing technology. These technologies enabled the sequencing of thousands to millions of DNA molecules in a single machine run.

- Biological Big Data:

The drop in DNA sequencing costs and the adoption of massively parallel sequencing resulted in exponential growth in sequence data in public databases. Sequencing data has exceeded the exabyte (10^18) level. New repository infrastructure for model organisms and general genomic databases emerged to store, organize, and make data accessible. The Genomic Standards Consortium was established in 2005 to define the minimum information required for genomic sequences.

**Aim**

- Data acquisition and database development
  To organize data in a way that allows researchers to access existing information and to submit new entries as they are produced.

- Tool development
  To develop tools and resources that aid in the analysis of data.

- Data analysis
  To use different tools to analyze the data and interpret the results in a biologically meaningful manner

**Branches**

There are several branches of Bioinformatics (Figure 1). Some of them are explained below.

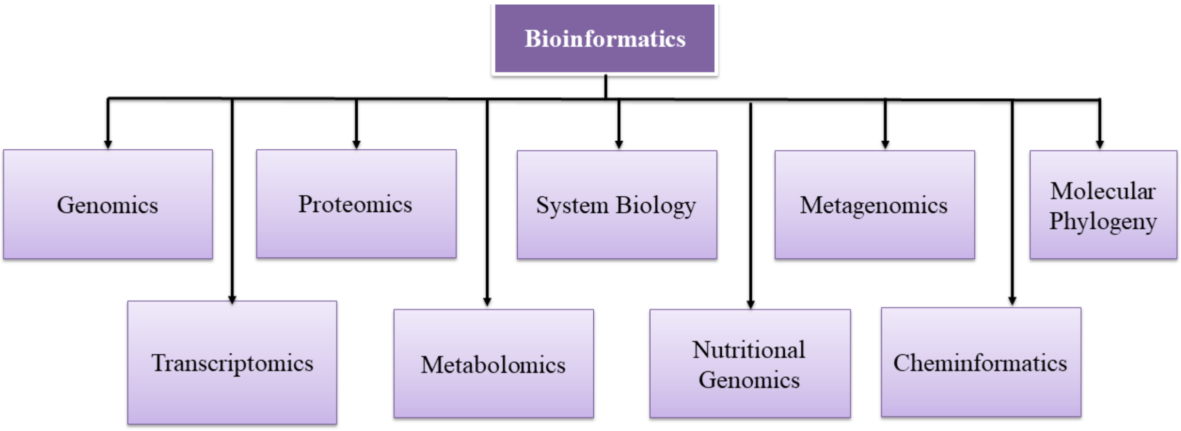


Figure 1: Branches of Bioinformatics

- Genomics

Genomics is a fundamental field in bioinformatics that focuses on the study of an organism's entire genetic material, which is stored in its DNA (or RNA for some viruses). This genetic material, often referred to as the genome, contains all the information needed to build and maintain an organism. Genomics aims to understand and analyze the structure, function, evolution, and variations in the genome. Here are the key components of genomics in bioinformatics:

- Sequencing: Genomic research often begins with DNA sequencing. This process involves determining the order of nucleotides (A, T, C, G) in a DNA molecule. There are various sequencing technologies, such as Sanger sequencing and next-generation sequencing (NGS), which allow scientists to read and decode the genetic information.

- Genome Assembly: The raw sequencing data obtained is fragmented into smaller pieces, and the bioinformatics part of genomics involves assembling these pieces to create a complete genome. Genome assembly algorithms help organize and connect these sequences to form a coherent picture of the genome.

- Functional Annotation: Once the genome is assembled, the next step is to identify and annotate the functional elements. This includes finding genes (coding regions), regulatory sequences, non-coding regions, and other structural components. Bioinformatics tools predict the locations of genes and their functions based on sequence similarity, conserved motifs, and other features.

- Comparative Genomics: Genomic sequences of different organisms, both within the same species and across species, are compared to identify similarities and differences. Comparative genomics helps in understanding evolutionary relationships, studying gene conservation, and discovering genes responsible for specific traits or diseases.

- Structural Genomics: Structural genomics focuses on determining the three-dimensional structures of proteins and other macromolecules encoded by the genome. This is crucial for understanding protein functions and interactions. Techniques like X-ray crystallography and nuclear magnetic resonance (NMR) spectroscopy are used in structural genomics.

- Functional Genomics: This field aims to understand the functions of genes and their products. Functional genomics methods, such as transcriptomics (studying gene expression), proteomics (studying proteins), and metabolomics (studying metabolites), provide insights into how genes are expressed and how they influence an organism's biology.

- Phylogenomics: Phylogenomics combines genomics and phylogenetics to study evolutionary relationships among species. It uses genomic data to reconstruct phylogenetic trees and understand the evolutionary history of different organisms.

- Genomic Variation: Genomic variation studies focus on identifying variations in the genome, such as single nucleotide polymorphisms (SNPs), insertions, deletions, and copy number variations. These variations can be associated with diseases and traits.

- Transcriptomics

Transcriptomics is a branch of bioinformatics and genomics that focuses on the study of transcriptomes, which are the complete sets of RNA transcripts produced in a cell, tissue, or organism. These RNA transcripts, often referred to as messenger RNA (mRNA), provide critical information about which genes are actively being expressed and to what extent in a specific biological sample. Understanding transcriptomes is vital for unraveling the molecular mechanisms underlying various biological processes and diseases. Here are the key components of transcriptomics in bioinformatics:

- Data Generation: Transcriptomics begins with the generation of RNA sequencing (RNA-Seq) data. RNA-Seq is a high-throughput technology that allows researchers to identify and quantify the RNA molecules present in a biological sample. It provides information about gene expression levels, alternative splicing, and the presence of non-coding RNAs, among other things.

- Data Preprocessing: The raw RNA-Seq data typically contains errors, biases, and artifacts. Preprocessing involves cleaning and quality-checking the data to ensure its reliability. This step includes tasks like adapter removal, read alignment to the reference genome or transcriptome, and removal of duplicate reads.

- Gene Expression Quantification: After preprocessing, bioinformaticians quantify gene expression levels. This step involves determining the number of RNA fragments that map to each gene, which serves as a measure of the gene's activity. Different algorithms and tools are available for this purpose.

- Differential Expression Analysis: One of the key objectives in transcriptomics is to identify genes that are differentially expressed under different experimental conditions. This analysis helps researchers understand how gene expression is altered in response to various stimuli, diseases, or genetic mutations. Statistical methods are used to compare expression levels between conditions.

- Functional Analysis: Transcriptomics data can be further analyzed to gain insights into the biological functions and pathways affected by changes in gene expression. Tools and databases, such as gene ontology (GO) analysis and pathway enrichment analysis, help in understanding the roles of differentially expressed genes.

- Alternative Splicing Analysis: In addition to quantifying gene expression, transcriptomics also allows for the study of alternative splicing events. Alternative splicing can generate multiple mRNA isoforms from a single gene, expanding the functional diversity of the proteome.

- Long Non-Coding RNA (lncRNA) Analysis: Transcriptomics can reveal the presence and differential expression of long non-coding RNAs, which play crucial roles in gene regulation and various cellular processes.

- Proteomics

Proteomics is a branch of bioinformatics and biology that focuses on the large-scale study of proteins. It involves the comprehensive analysis of the structure, function, and expression of all the proteins in a biological system, such as a cell, tissue, or organism. Proteins are crucial molecules in living organisms, responsible for performing various biological functions, and understanding their properties and behaviors is essential for gaining insights into complex biological processes. Here are some key aspects of proteomics in bioinformatics:

- Protein Identification and Characterization: Proteomics involves identifying and characterizing proteins. This can include determining the amino acid sequence, post-translational modifications (e.g., phosphorylation, glycosylation), and three-dimensional structures of proteins.

- Protein Expression and Quantification: Proteomic studies aim to measure the relative abundance of proteins in different biological conditions. This can help researchers understand how proteins are regulated and expressed under various circumstances, such as disease states or drug treatments.

- Protein-Protein Interactions: Proteins rarely function in isolation; they often work together in complexes. Proteomics helps in identifying protein-protein interactions, which are crucial for understanding cellular processes and signaling pathways.

- Functional Annotation: Assigning biological functions to proteins is a fundamental goal of proteomics. This may involve studying the role of proteins in specific pathways, cellular processes, and disease mechanisms.

- Biomarker Discovery: Proteomics plays a vital role in biomarker discovery for diseases. By comparing protein profiles in healthy and diseased samples, researchers can identify potential biomarkers for early disease diagnosis or monitoring treatment responses.

- Mass Spectrometry and Other Techniques: Mass spectrometry is a common technology used in proteomics. It allows the precise measurement of protein masses and has the capability to identify and quantify thousands of proteins simultaneously. Other techniques, like gel electrophoresis and antibody-based assays, are also used in proteomic studies.

- Metabolomics

Metabolomics is a subfield of bioinformatics that focuses on the comprehensive analysis of small molecules, known as metabolites, in biological systems. Metabolites include a wide range of compounds such as sugars, amino acids, lipids, organic acids, and other small molecules that play crucial roles in various biochemical processes within living organisms. Metabolomics aims to identify, quantify, and analyze these metabolites to gain insights into an organism's metabolism and understand its biochemical pathways, which can be essential for both basic research and practical applications. Following are the key concepts of metabolomics in bioinformatics:

- Data Generation: Metabolomics data is generated through various analytical techniques, such as mass spectrometry (MS), nuclear magnetic resonance spectroscopy (NMR), and liquid or gas chromatography. These techniques allow researchers to detect and quantify a wide range of metabolites present in a biological sample.

- Data Preprocessing: Metabolomics datasets can be large and complex, and preprocessing is a crucial step in data analysis. It involves data cleaning, alignment, normalization, and the removal of any technical variation or noise. This step ensures that the data is suitable for subsequent analysis.

- Metabolite Identification: One of the primary goals of metabolomics is to identify the metabolites detected in the sample. Bioinformatics tools and databases play a critical role in matching experimental data to known metabolite profiles. This process often involves spectral databases, reference libraries, and computational algorithms to make accurate identifications.

- Quantitative Analysis: Metabolomics data also provides quantitative information about the abundance of metabolites in a sample. Researchers can compare the concentration of specific metabolites across different samples or conditions to understand the metabolic changes.

- Statistical and Multivariate Analysis: Bioinformatics tools are used to analyze metabolomics data statistically. Techniques like principal component analysis (PCA), partial least squares-discriminant analysis (PLS-DA), and hierarchical clustering can reveal patterns and trends in the data, helping researchers identify biomarkers or distinguish between sample groups.

- Pathway Analysis: Metabolomics data can be integrated with other omics data, such as genomics and proteomics, to gain a more comprehensive understanding of the biological systems. Pathway analysis tools help researchers map metabolites onto known metabolic pathways, identifying key pathways and their interactions.

- Biomarker Discovery: Metabolomics is often applied to discover biomarkers, which are specific metabolites associated with a particular disease or condition. Identifying biomarkers can be valuable in disease diagnosis, prognosis, and treatment monitoring.

- System Biology

Systems biology is an interdisciplinary field in bioinformatics that focuses on understanding complex biological systems by studying how individual components, such as genes, proteins, and metabolites, interact and function as a whole. It aims to provide a comprehensive and integrated view of biological processes to better explain and predict the behavior of living organisms. Following are the key aspects of systems biology in bioinformatics:

- Holistic Approach: Systems biology takes a holistic approach to biology, looking beyond the individual components. It considers the interactions, feedback loops, and dependencies among genes, proteins, and other molecules in biological systems.

- Data Integration: It involves integrating data from various sources, such as genomics, transcriptomics, proteomics, and metabolomics, to create a comprehensive picture of biological processes. This integration is often achieved through computational methods.

- Mathematical and Computational Modeling: Systems biology heavily relies on mathematical and computational modeling techniques. These models simulate biological processes and provide a framework for understanding and predicting system behavior. Examples of modeling techniques include differential equations, agent-based models, and network analysis.

- Network Analysis: Biological networks, such as protein-protein interaction networks and metabolic pathways, are a central focus of systems biology. Network analysis helps uncover relationships and patterns within complex biological systems.

- Dynamic Processes: Systems biology often deals with dynamic processes. It explores how biological systems change over time in response to various stimuli, environmental conditions, or genetic variations.

- Hypothesis Generation and Testing: Systems biology generates hypotheses about how biological systems work. These hypotheses can then be tested through experiments, helping to refine the models and improve our understanding of the system.

- Biomedical Applications: Systems biology has practical applications in medicine and drug discovery. It can be used to study complex diseases, identify potential drug targets, and optimize treatment strategies.

- Quantitative Biology: A quantitative approach is a hallmark of systems biology. It involves measuring and quantifying various biological components and processes, often using high-throughput technologies.

- Nutritional Genomics

Nutritional genomics, often referred to as nutrigenomics, is a branch of genomics that focuses on the interaction between nutrition and genes. It aims to understand how an individual's genetic makeup influences their response to specific nutrients, foods, and dietary patterns. Nutritional genomics plays a significant role in agriculture by helping improve crop production and the nutritional quality of food. Following are the key points how it applies to agriculture in the context of bioinformatics:

- Genomic Sequencing of Crops: One of the key aspects of nutritional genomics in agriculture is the genomic sequencing of crop plants. Advances in bioinformatics and genomics have made it possible to sequence the entire genomes of various crops, such as rice, wheat, and maize. This provides a comprehensive understanding of the genes and genetic variations present in these crops.

- Identification of Nutritional Genes: Bioinformatics tools are used to identify genes related to the nutritional content of crops. This includes genes that influence the levels of essential nutrients like vitamins, minerals, and proteins. By identifying these genes, researchers can target specific genetic traits for crop improvement.

- Marker-Assisted Breeding: Nutritional genomics, coupled with bioinformatics, facilitates marker-assisted breeding programs. Researchers can identify genetic markers associated with desirable nutritional traits in crops. This helps in the selection and breeding of crop varieties with improved nutritional content.

- Customized Diets for Livestock: Nutritional genomics also plays a role in livestock agriculture. By understanding the genetic makeup of animals, farmers can tailor their diets to optimize growth, health, and the nutritional quality of animal products, such as meat and dairy.

- Optimizing Soil and Crop Interactions: Understanding the genetic factors that influence a crop's ability to absorb nutrients from the soil is crucial for sustainable agriculture. Bioinformatics helps in studying these interactions and optimizing nutrient uptake for crop growth.

- Resilience to Environmental Stress: Nutritional genomics can help in developing crop varieties that are resilient to environmental stress, such as drought or nutrient-poor soil. By understanding the genetic basis of stress responses, crops can be engineered to thrive under challenging conditions.

- Personalized Nutrition: In the context of agriculture, personalized nutrition refers to tailoring crop choices and farming practices based on the nutritional needs of specific regions or populations. Nutritional genomics can help identify which crops are best suited for a particular area, taking into account genetic factors.

- Metagenomics

Metagenomics is a powerful field within bioinformatics that has significant implications for agriculture. It involves the study of genetic material collected directly from environmental samples, such as soil, water, or plant tissues. In the context of agriculture, metagenomics has several applications:

- Soil Health and Microbiome Analysis: Metagenomics is used to analyze the soil microbiome, which includes bacteria, fungi, and other microorganisms. Understanding the diversity and functional potential of these microorganisms is crucial for assessing soil health. Healthy soils are essential for crop growth and productivity. Metagenomics helps in identifying beneficial microbes, understanding their roles in nutrient cycling and disease suppression, and designing strategies for sustainable agriculture.

- Plant-Microbe Interactions: Metagenomics enables the study of interactions between plants and the microorganisms in the rhizosphere (the soil zone around plant roots).

These interactions play a vital role in nutrient uptake, disease resistance, and overall plant health. By analyzing the metagenome of the rhizosphere, researchers can gain insights into the beneficial or pathogenic microorganisms present and their impact on crop growth.

- Crop Pathogen Detection: Metagenomics can be used to identify and characterize pathogens in agricultural environments. By analyzing metagenomic data from infected plant samples, researchers can detect the presence of harmful pathogens, such as viruses, bacteria, or fungi. This information is valuable for disease management and quarantine measures.

- Biological Control: Metagenomics can assist in identifying natural enemies of agricultural pests. Beneficial microorganisms or nematodes can be detected and used for biological pest control strategies, reducing the reliance on chemical pesticides.

- Microbial-Based Crop Enhancements: Metagenomics helps in the discovery and development of microbial-based products that can enhance crop growth, nutrient uptake, and stress resistance. These products, such as biofertilizers or biostimulants, are environmentally friendly alternatives to traditional agricultural inputs.

- Monitoring Ecosystem Changes: Metagenomics can be used to monitor changes in agricultural ecosystems over time. This includes tracking shifts in microbial populations due to changes in land use, cropping systems, or climate conditions. Understanding these changes can guide more sustainable agricultural practices.

- Resilience to Climate Change: As climate change impacts agriculture, metagenomics can provide insights into how plant-microbe interactions may be affected. This information is essential for developing crop varieties and management strategies that can adapt to changing environmental conditions.

- Waste Management: In livestock farming, metagenomics can be used to manage waste, such as manure. By understanding the microbial communities in waste, strategies for reducing environmental contamination and converting waste into bioenergy or other valuable products can be developed.

## Cheminformatics

Cheminformatics is a specialized field within bioinformatics that deals with the storage, retrieval, and analysis of chemical information and data, particularly in the context of biological and agricultural applications. In the context of agriculture, cheminformatics plays a crucial role in various aspects of crop management, agricultural research, and biotechnology. Here's how cheminformatics is applied in bioinformatics to benefit agriculture:

- Pesticide and Fertilizer Development: Cheminformatics is used to design and develop new pesticides and fertilizers. Researchers can use databases of chemical structures and properties to predict the effectiveness and safety of these agrochemicals. This helps in reducing the environmental impact and improving crop yields.

- Chemical Safety: Cheminformatics tools are used to assess the safety of chemicals used in agriculture. This includes predicting the toxicity of pesticides and assessing their impact on non-target organisms, such as beneficial insects and pollinators.

- Drug Discovery for Plant Health: Bioinformatics and cheminformatics can be used to discover compounds that protect plants from diseases. This is essential in reducing the need for chemical pesticides. Identifying compounds that enhance plant immunity or inhibit pathogens is a common application.

- Plant Breeding: In modern agriculture, cheminformatics plays a role in crop improvement. For instance, researchers can use chemical profiling to identify compounds responsible for desirable traits in crops, such as nutritional content or disease resistance. This information can guide traditional breeding programs or genetic engineering efforts.

- Metabolomics: Cheminformatics tools are crucial in metabolomics, which involves studying the chemical processes occurring within organisms, including plants. Metabolomics data can be used to understand how plants respond to environmental changes, stress, and disease, helping in crop management and breeding.

- Herbicide Design: Cheminformatics assists in designing herbicides that selectively target weeds while sparing crop plants. Understanding the chemical properties and interactions of herbicides with plant biology is key to developing effective and environmentally friendly weed control solutions.

- Molecular Docking: Cheminformatics and molecular docking techniques are used to study how chemicals interact with biological molecules like plant proteins and enzymes. This information is valuable in understanding how chemicals can influence plant processes and can be used in the development of targeted agrochemicals.

- Environmental Impact Assessment: Cheminformatics can be used to assess the environmental impact of agricultural chemicals. This includes predicting their persistence in soil and water, their potential to leach into groundwater, and their impact on non-target organisms.

**Computational resources**

Databases and algorithms are essential components of bioinformatics, a multidisciplinary field that combines biology, computer science, and data analysis. They play a crucial role in managing, analyzing, and interpreting biological data, making it easier for researchers to extract meaningful information from large datasets. An overview of databases and algorithms commonly used in bioinformatics are as follows:

- Databases in Bioinformatics:

Genomic Databases: These contain DNA and RNA sequences from various species. Examples include GenBank, Ensembl, and RefSeq. Genomic databases provide a wealth of genetic information used in sequence analysis, gene annotation, and comparative genomics.

Protein Databases: These store information about proteins, including sequences, structures, and functional annotations. Popular protein databases include UniProt, Protein Data Bank

(PDB), and Pfam. Researchers use these databases to study protein structure, function, and evolution.

Gene Expression Databases: These house data related to gene expression levels in different tissues, conditions, or experimental settings. The Gene Expression Omnibus (GEO) and ArrayExpress are examples of repositories for gene expression data.

Metabolic Pathway Databases: These contain information about biochemical pathways and the interactions between molecules in metabolic processes. KEGG and Reactome are widely used for pathway analysis.

## ✦ Algorithms

In bioinformatics, several key algorithms and methods are used for tasks related to sequence analysis and phylogenetics. These algorithms are fundamental for understanding the relationships between biological sequences, such as DNA, RNA, and proteins. Here, I'll provide an overview of pairwise and multiple sequence alignment, substitution matrices, and phylogenetic tree reconstruction algorithms:

1. Pairwise Sequence Alignment:
Pairwise sequence alignment is used to identify regions of similarity between two biological sequences. This can be helpful for comparing sequences for structural and functional analysis.
Needleman-Wunsch Algorithm: This algorithm performs global alignment, meaning it compares the entire sequences and finds the optimal alignment by maximizing a similarity score.
Smith-Waterman Algorithm: It's used for local sequence alignment, which finds the best-matching subsequence within the sequences.

2. Substitution Matrices:
Substitution matrices are used to score the substitution of one amino acid or nucleotide with another in sequence alignments. They provide a measure of evolutionary relatedness between sequences.
PAM (Point Accepted Mutation) Matrices: Developed by Margaret Dayhoff, PAM matrices describe the probability of specific amino acid substitutions over a fixed evolutionary distance.
BLOSUM (Blocks Substitution Matrix) Matrices: These matrices are used in protein sequence alignment and are based on observed substitutions within closely related sequences.

3. Phylogenetic Tree Reconstruction:
Phylogenetic tree reconstruction is used to infer evolutionary relationships and construct a tree that represents the divergence of species or sequences over time.
Neighbor-Joining (NJ): A distance-based method that constructs a tree by iteratively joining the closest neighbors.
Maximum Parsimony: This method seeks the tree that requires the fewest evolutionary changes (mutations) to explain the observed sequence data.
Maximum Likelihood: A likelihood-based approach that estimates the probability of observing the given sequences under different tree topologies.
Bayesian Inference: Uses a Bayesian framework to estimate the posterior distribution of tree topologies and model parameters.

### Challenges

- Traditional bioinformatics methods heavily rely on reference databases, limiting analysis to known sequences and structures.
- These methods struggle to predict novel patterns, making them less effective in underexplored biological areas.
- Rapid improvements in high throughput sequencing technologies have given rise to heterogeneous and enormous amounts of omics data making it a big data problem.
- Moreover, there has been a shift in data types, transitioning from conventional structured data to a more diverse range of architectures, including unstructured, semi-structured, and heterogeneous formats, each with distinct characteristics.
- There is a need for advanced computational techniques such as Artificial Intelligence (AI) to leverage various data types, including sequences, images, and unstructured text, facilitating the integration of diverse biological information.

### Big Data

In bioinformatics, as in other fields, the concept of "Big Data" is characterized by the "5 Vs," which describe key aspects of the data challenges faced. These Vs are Volume, Velocity, Variety, Veracity, and Value.

- Volume: Big data in bioinformatics originates from various sources, including genomics (DNA sequencing), transcriptomics (RNA sequencing), proteomics (protein data), metabolomics (small molecule data), structural biology (protein structures), and more. Additionally, data sources include literature, and data from high-throughput experiments. Genomic data, in particular, has seen a dramatic increase in the form of DNA and RNA sequences, with millions of sequences available in public databases. This volume continues to expand rapidly.

- Variety: Biological data comes in diverse formats, such as sequences, alignments, 3D structures, images, clinical records, and omics data. Integrating and analyzing these various data types poses challenges.

- Velocity: The speed at which new biological data is generated is incredibly high, especially with the advent of high-throughput sequencing technologies. Keeping up with the pace of data generation is a significant challenge for bioinformaticians.

- Veracity: Veracity relates to the accuracy, quality, and reliability of data. In bioinformatics, ensuring the veracity of data is crucial since errors or inaccuracies can lead to incorrect scientific conclusions.

- Value: The value of big data in bioinformatics is the benefit that can be derived from it. It involves extracting meaningful insights, making discoveries, and ultimately improving healthcare, agriculture, and various biological research fields.

### Artificial Intelligence in Bioinformatics

Artificial intelligence (AI) was formally defined at the Dartmouth conference in 1956. It quickly entered a period of rapid development and innovation, becoming known as the "golden age" of AI. The field of AI encompasses a wide array of content, and one of its crucial branches is machine learning (ML). ML is a methodology for achieving AI and includes a range of mathematical tools and algorithms. Although ML initially achieved remarkable progress, it faced a significant setback in the 1960s due to theoretical limitations.

It wasn't until the introduction of the backpropagation algorithm in the 1980s that ML experienced a resurgence in activity and widespread application. Subsequently, deep learning (DL) emerged from artificial neural networks (ANN) within the realm of machine learning and has been a driving force behind the current era of deep learning since 2006.

Over the last decade, AI has found extensive use in omics studies, thanks to the accumulation of large-scale omics data and the growing need for big data analysis. Machine learning, as a subset of AI, focuses on acquiring insights and establishing patterns from data through computational models and algorithms. Its goal is to enhance system performance through computation and learning from experiences. Machine learning has diverse applications, spanning natural language processing, computer vision, data mining, and more. Various machine learning algorithms serve distinct purposes, including clustering, classification, regression, association rule mining, dimension reduction, and others. Based on the nature of the data and training strategies, machine learning is categorized into three primary types: supervised, unsupervised, and reinforcement learning.

Supervised learning deals primarily with regression and classification problems, while unsupervised learning focuses on clustering. Reinforcement learning, on the other hand, involves learning from new experiences through trial-and-error. The field boasts a variety of traditional machine learning algorithms such as generalized regression, decision trees, naive Bayes, support vector machines (SVM), K-means clustering, and more.

Deep learning, a critical branch of machine learning, originated from artificial neural networks and was formally introduced in 2006. It has since experienced rapid and substantial development. Deep learning encompasses a multidisciplinary approach, merging elements of statistics, optimization, algorithms, programming, distributed computing, and other fields. By constructing models with multiple hidden layers, deep learning allows for the discovery of intricate relationships within data, improving the accuracy of classification and prediction. This evolution has had a significant impact on various fields, making it a fundamental component of the broader AI landscape.

In the forthcoming sections, various applications of ML in different omics have been discussed.

- Machine learning in genomics:

  In the field of genomics and genome research, machine learning has become a crucial tool for various applications. These applications encompass diverse aspects of genomics, from predicting 3D genome structures to genome annotation, transcription regulation, effects of genetic variants, and even genome editing (Figure 2).

  1. Reconstruction of 3D Genome Structure:
  Understanding the spatial organization of the eukaryotic genome is essential for elucidating chromosomal activities within the cell. Experimental techniques, such as chromosome conformation capture (3C)-based technologies, provide insights into 3D genome organization but have limitations in resolution and cost. Therefore, machine learning methods have been developed to complement experimental studies. These methods are categorized based on their training data, including genomic sequences, 3C-based interactions, chromatin states derived from epigenetic modifications, or hybrid data. They aim to predict various aspects of 3D genome structure, including reconstruction, compartmentalization, topologically associating domains (TADs), and chromatin loops.

2. Computational Modeling of Epigenomic and Chromatin States:
Epigenomic modifications play a crucial role in genome regulation. Machine learning approaches have been employed to interpret and predict the effects of epigenetic modifications, DNA methylation, histone modifications, and chromatin states. These methods generate features from epigenetic data and leverage deep learning techniques to understand and predict epigenomic changes.

3. Genome Annotation and Transcription Regulation:
Machine learning is applied to the annotation of the genome, including the identification of protein-coding genes, non-coding RNAs, microRNAs, transcript splicing isoforms, regulatory elements, protein-binding sites, and cis-regulatory binding modules. It goes beyond simple identification to elucidate their functions and interactions. This is essential for understanding the roles of different genomic elements in gene regulation.

4. Identifying the Effects of Genetic Variants:
Genetic variants, especially those in non-coding regions, can significantly impact gene expression and phenotypes. Machine learning models have been developed to classify and predict the pathogenicity of genetic variants. These models help identify functional effects of non-coding variants and their contributions to diseases.

5. Machine Learning in Genome Editing:
The advent of genome editing technologies, such as CRISPR, has opened new possibilities in genome engineering. Machine learning is applied to design guide RNAs for CRISPR-based editing, predict cleavage tendencies, evaluate off-target effects, and identify optimal editing locations. These methods contribute to more precise and efficient genome editing.

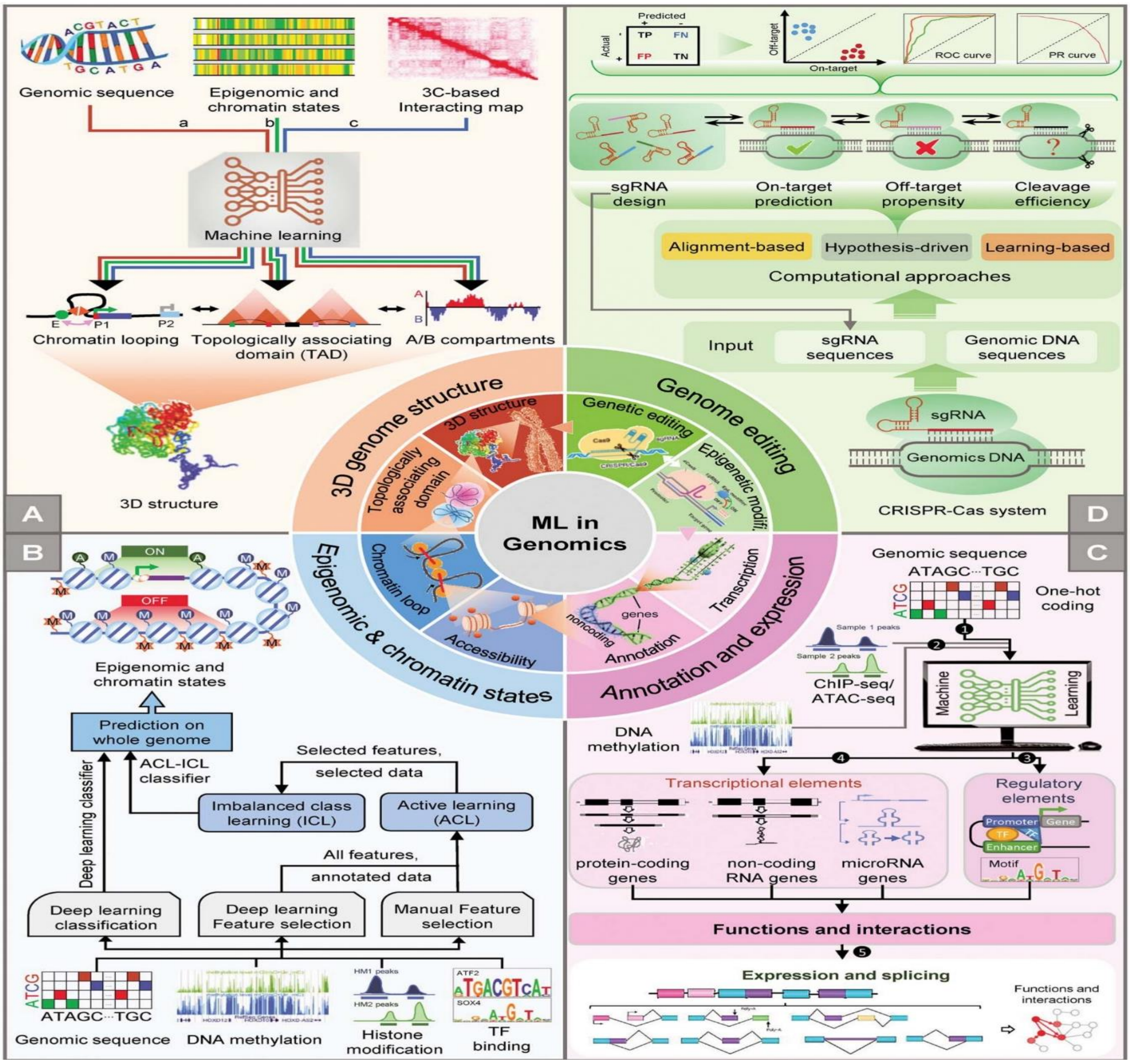


Figure 2: Schematics representation of the application of AI in genomics (Li et al., 2022)

- Machine learning in transcriptomics

Machine learning, particularly deep learning, has made significant contributions to various aspects of transcriptomics, including the prediction and understanding of gene expression, splicing patterns, and transcription factor binding sites (Figure 3).

1. Prediction of Gene Expression:

Machine learning, especially deep learning, has proven highly effective in predicting gene expression levels based on genetic and epigenetic information. For instance, deep neural networks (DNNs) have been used to build models like D-GEX, which can predict target gene expression based on landmark genes. Histone modifications, which play a vital role in gene regulation, have also been leveraged for gene expression prediction using models like DeepChrome, demonstrating the superior performance of deep learning compared to traditional machine learning methods.

2. Prediction and Classification of Splicing:

Splicing, which determines how the genome is transcribed, influences the diversity of transcriptomes and proteomes. Aberrant splicing can lead to diseases, making it a crucial area of study. Deep learning methods, including deep neural networks, are employed to predict and classify splicing patterns based on RNA-seq data, genomic sequences, and epigenetic features. These models accurately predict splicing outcomes in different biological contexts and contribute to our understanding of splicing regulation.

3. Prediction of Transcription Factor Binding Sites:
Transcription factors (TFs) are central to gene regulation, and their binding sites on DNA are essential for controlling gene expression. Machine learning, particularly deep learning, has been applied to identify TF-binding sites more accurately and efficiently. Models like PIQ and DeepBind have demonstrated the ability of deep learning to predict TF-binding sites. These models improve the accuracy of prediction, especially in comparison to traditional methods based on position weight matrices (PWMs).

4. Auxiliary Diagnosis Using Transcriptomics:
Machine learning plays a significant role in aiding disease diagnosis, particularly in the medical field. Artificial neural networks (ANNs) can analyze gene expression data to enhance the accuracy and efficiency of disease classification and diagnosis. Machine learning models combined with gene expression data are used for various medical applications, including predicting myopathy subtypes, drug-induced liver injury, and diagnoses related to mental and neurological diseases. In the context of cancer, machine learning assists in cancer classification, predicting molecular subtypes, early diagnosis, prognosis, and recurrence prediction. The integration of multiple cohort datasets is a promising avenue for improving auxiliary diagnosis, although the challenge of limited data remains.

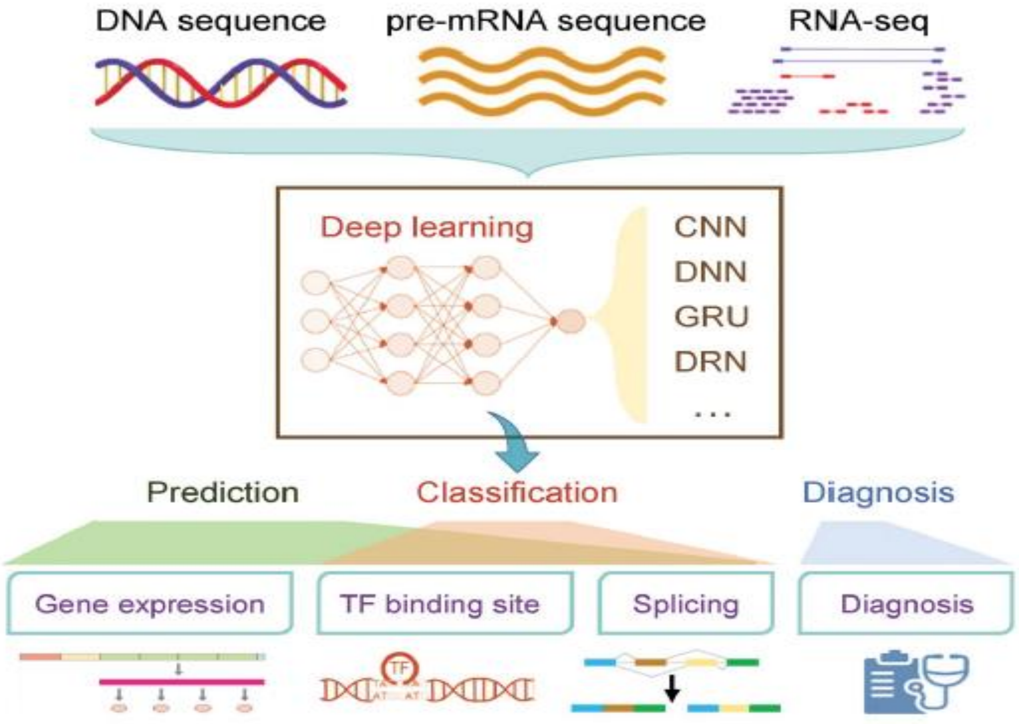


Figure 3: Schematics representation of the application of AI in transcriptomics (Li et al., 2022)

- Machine learning in proteomics

Machine learning is playing a pivotal role in the field of proteomics, where it aids in efficiently processing and analyzing vast amounts of proteomic data (Figure 4). Specifically, machine learning methods are significantly impacting proteomics in various areas, as outlined below.

1. Biomass Spectrometry

Mass spectrometry (MS) is an indispensable tool for studying protein structures and components. However, the processing of MS data has often lagged behind the development of MS instruments. Machine learning, particularly deep learning, is stepping in to address the challenges posed by high-dimensional and sparse proteomic data. Deep learning models are being harnessed for tasks like de novo sequencing, peptide property prediction, and mass spectrometry imaging analysis. For instance, DeepNovo, a deep learning-based model, is enhancing the accuracy of de novo peptide sequencing. Moreover, DeepRT employs deep learning to predict peptide retention times, a critical factor in liquid chromatography-mass spectrometry tandem analysis. Machine learning has the potential to significantly enhance the retrieval and analysis of peptide data, thereby advancing our understanding of proteome characterization.

2. Screening of Protein Biomarkers

Biomarkers are vital for disease screening, diagnosis, and therapy guidance. Traditional statistical methods often face limitations in biomarker discovery due to classification boundaries and variable correlations. Machine learning methods, both supervised and unsupervised, offer more flexibility in this context. Researchers have been combining machine learning with proteomic techniques, such as mass spectrometry, to identify disease-specific protein markers. For example, a study utilized a deep belief network (DBN) to screen for protein diagnostic markers in Alzheimer's disease, yielding a marker group with high diagnostic accuracy. While machine learning holds immense promise in biomarker discovery, challenges like overfitting and model interpretability need to be addressed.

3. Nucleic Acid–Binding Protein Prediction

Identifying proteins that bind to nucleic acids is essential for understanding various biological processes. Traditionally, this identification has been hampered by accuracy and scalability issues. However, with the availability of high-throughput measurements, such as protein binding microarrays and SELEX, machine learning has emerged as a highly accurate predictor of nucleic acid–binding properties in proteins. Tasks include DNA-binding domain recognition and predicting protein-DNA/RNA docking interactions. Despite the success, challenges remain, particularly in reducing cross-prediction between DNA and RNA-binding residues.

4. Predicting Protein–Protein Interactions (PPIs)

PPIs are a critical domain where machine learning is revolutionizing our understanding of protein functions. While public databases offer some PPI data, they often lack specificity and comprehensiveness. Combining experimental methods with machine learning is proving effective in predicting PPIs accurately. Various machine learning algorithms, including random forests, support vector machines, and Bayesian probabilistic inference, are being used for PPI prediction. Deep learning has also found application in predicting PPIs through methods like domain-based ensemble models. Accurate identification of PPIs is instrumental in comprehending a wide range of physiological activities.

5. Protein Post-Translational Modification (PTM)

PTM prediction is yet another area significantly benefiting from machine learning methods. PTMs, such as phosphorylation and glycosylation, play vital roles in regulating protein function. Machine learning models have been developed to predict PTM sites with high accuracy. For example, Musite predicts phosphorylation sites, while GlycoEP identifies N-, O-, and C-linked glycosylation sites. Additionally, web servers like MusiteDeep employ convolutional neural networks (CNNs) for predicting multiple PTM sites simultaneously, offering advantages in accuracy and speed. Furthermore, tools like SAPH-ire TFx assist in the identification of functional PTM sites from large-scale datasets.

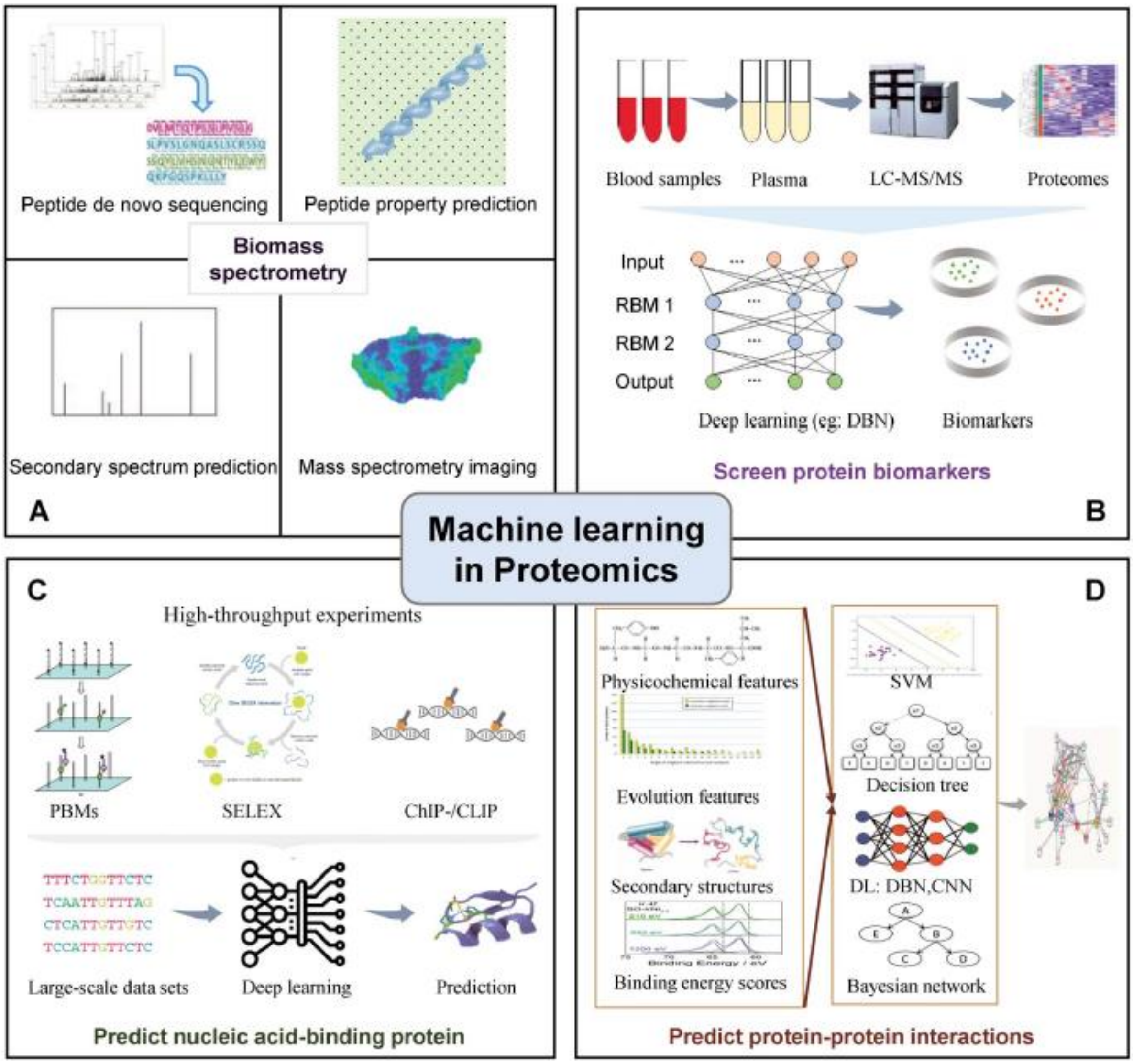


Figure 4: Schematics representation of the application of AI in proteomics (Li et al., 2022)

- Machine learning in metabolomics

Metabolomics, akin to genomics and proteomics, focuses on quantitatively analyzing all metabolites in organisms to uncover their relationships with physiological and pathological changes. It's a valuable technology for diagnosing diverse diseases characterized by metabolic variations. Traditional methods often struggle with the sparsity of large-scale metabolomic data obtained through mass spectrometry, chromatography, and nuclear magnetic resonance. This challenge has led to an increased interest in machine learning algorithms. In the field of metabolomics, various machine learning techniques are being employed, contributing to advancements in data processing, metabolic phenotype stratification, and metabolic modeling (Figure 5).

1. Data Processing and Analysis:

Machine learning has significantly enhanced the processing and analysis of metabolomic data. These algorithms excel in pattern recognition and multivariate classification, assisting in classifying data based on complex patterns. Traditional methods like partial least squares discriminant analysis (PLS-DA), as well as support vector machines (SVM), have been employed for this purpose. SVM has gained prominence in metabolomics due to its high prediction and classification accuracy. Deep learning, a subset of machine learning, has also been applied in metabolomics for processes like estimating the detection probability of specific peaks. Deep learning, through methods like deep neural networks (DNNs), aids in eliminating false-positive peaks, enhancing the quality of metabolomic data. Tandem mass spectrometry (MS/MS) is used to identify unknown metabolites. The application of deep learning, such as the DeepMASS framework, helps effectively identify these unknown metabolites. Additionally, machine learning methods are being used to automate quality control and quality assurance processes in data processing.

2. Stratification of Metabolic Phenotypes:

Machine learning, particularly deep learning, is revolutionizing the stratification of metabolic phenotypes. This approach characterizes the metabolic profiles and processes of individuals based on the presence, content, and ratios of specific metabolites. Deep learning techniques have demonstrated success in capturing the intricate metabolic characteristics present in the data. For example, deep neural networks combined with t-distribution random neighborhood embedding have revealed the metabolic heterogeneity in human colorectal cancer. Deep learning frameworks are also employed in classifying the estrogen receptor status of breast cancer, surpassing other machine learning methods in prediction accuracy. Novel methods combining deep neural networks enhance metabolic phenotype stratification and metabolite selection, offering high classification accuracy.

3. Genome-Scale Construction of Metabolic Models:

Machine learning is also playing a vital role in constructing genome-scale metabolic models (GEMs). GEMs encompass the metabolic reactions of a specific organism's genome and serve as a platform for metabolic flux modeling. The modeling process involves constraint-based quantitative modeling, integrating biochemical and genetic information. Machine learning optimizes model parameters, tests various input conditions, and enhances biomarker recognition, quantifying metabolite flux, and predicting metabolic genes. Applications extend to determining predictors of metabolic-related drug side effects, generating collision cross-section values of small molecules, and identifying early metabolic disease markers. Despite these advancements, challenges persist, including experimental limitations, small sample sizes, interpretability issues, and a lack of comprehensive reference data.

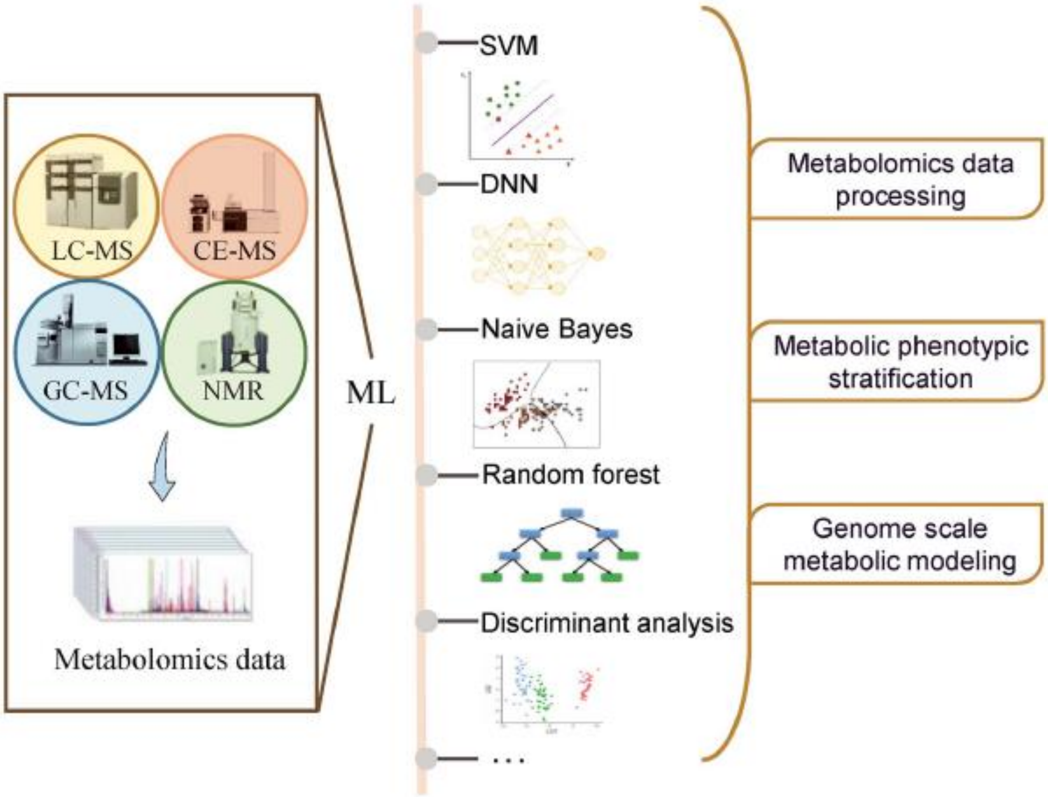

LC-MS
CE-MS
GC-MS
NMR
Metabolomics data
ML
SVM
DNN
Naive Bayes
Random forest
Discriminant analysis
...
Metabolomics data processing
Metabolic phenotypic stratification
Genome scale metabolic modeling

Figure 5: Schematics representation of the application of AI in metabolomics (Li et al., 2022)

## References

Gauthier, J., Vincent, A. T., Charette, S. J., & Derome, N. (2019). A brief history of bioinformatics. Briefings in bioinformatics, 20(6), 1981-1996.

Jawdat, D. (2006, April). The era of bioinformatics. In 2006 2nd International Conference on Information & Communication Technologies (Vol. 1, pp. 1860-1865). IEEE.

Li, R., Li, L., Xu, Y., & Yang, J. (2022). Machine learning meets omics: applications and perspectives. Briefings in Bioinformatics, 23(1), bbab460.

Mochida, K., & Shinozaki, K. (2011). Advances in omics and bioinformatics tools for systems analyses of plant functions. Plant and Cell Physiology, 52(12), 2017-2038.

# ASHOKA: Functioning and Activities

**K.K. Chaturvedi, U.B. Angadi and Jai Bhagwan**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

First HPC systems were vector-based systems (e.g. Cray) named ‘supercomputers’ because they were an order of magnitude more powerful than commercial systems. The ‘supercomputer’, a large systems are just scaled up versions of smaller systems. High performance computing can mean high flop count per processor and totalled over many processors working on the same or related problems. This can have faster turnaround time, more powerful system, scheduled to first available system(s) and using multiple systems simultaneously. The HPC is any computational technique that solves a large problem faster than possible using single, commodity systems, Custom-designed, high-performance processors, Parallel computing, Distributed computing and Grid computing.

Parallel computing is a single system with many processors working on the common task. The Distributed computing is configured as many systems loosely coupled by a scheduler to work on related problems and Grid Computing is defined as many systems tightly coupled by software and networks to work together on single problems or on related problems.

Parallel computer is a computer that contains multiple processors where each processor works on its section of the problem and allowed to exchange information with other processors.

Two big advantages of parallel computers are performance and memory. Parallel computers enable us to solve problems that benefit from or require, fast solution, require large amounts of memory and both.

As per the Moore’s Law ‘predicts’ that single processor performance doubles every 18 months, eventually physical limits on manufacturing technology will be reached as in figure 1.

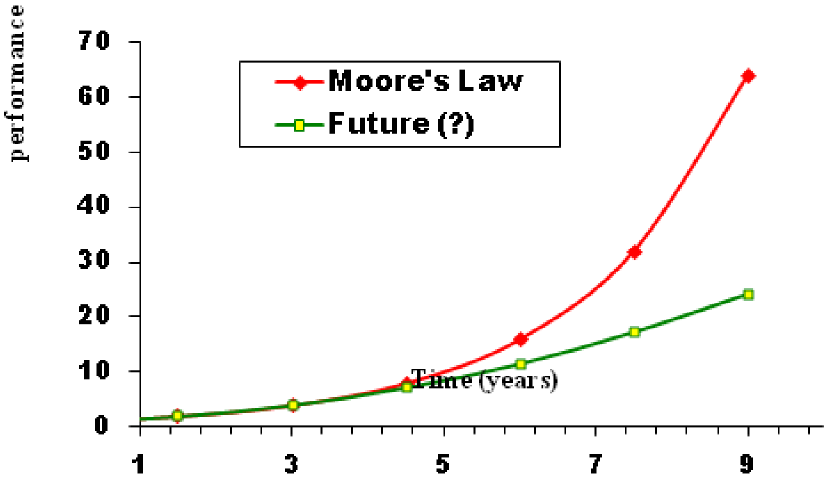


**Fig. 1: Moore’s Law towards performance of the system**

There are two types of parallel computers by their memory model namely shared memory and distributed memory. All processors have access to a pool of shared memory (Figure 2-A) while each processor has its own local memory in distributed memory (Figure 2-B).

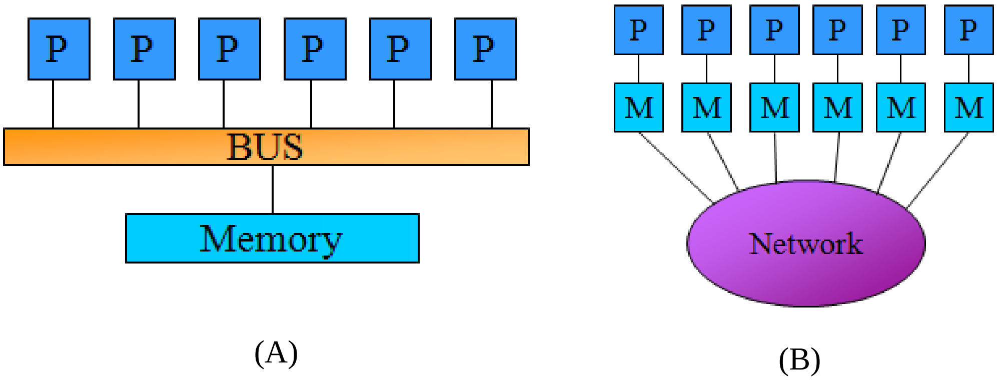


**Fig. 2: Shared Memory and distributed memory system**

Shared memory have two types of architecture i.e., Uniform memory access (UMA) and Non-uniform memory access (NUMA). Each processor has uniform access to memory in UMA and also called as symmetric multiprocessors, or SMPs (Figure 3-A). Time for memory access depends on location of data in NUMA as local access is faster than non-local access but it is easy to scale up than SMPs (Figure 3-B).

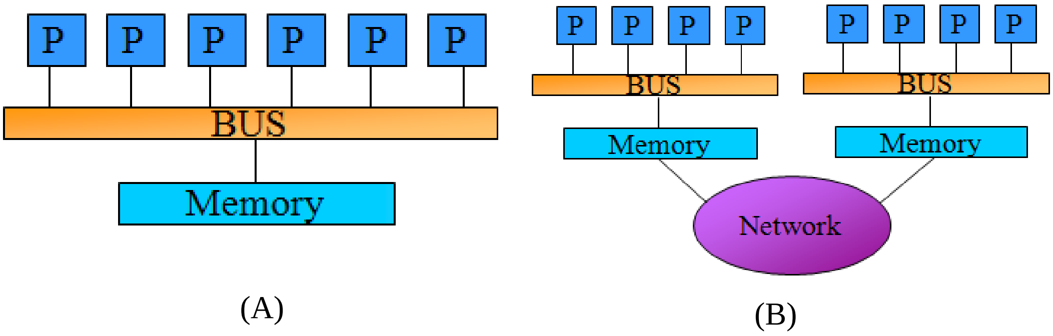


**Fig. 3: Shared Memory with UMA and NUMA**

The distributed memory is two types namely Massively Parallel Processor (MPP) and cluster. MPP is tightly integrated, single system image and cluster is an individual computers connected by specialized software and connected using interconnect network. Distributed memory is shown in figure 4.

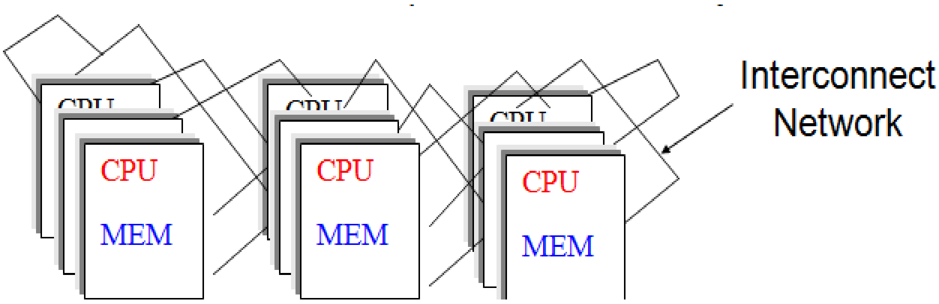


**Fig. 4: Distributed Memory**

Both types of memory systems have processors, memory and network/interconnect.

## Terminology

Clock period (cp): The minimum time interval between successive actions in the processor. It is measured in nanoseconds (~1-5 for fastest processors) which is inverse of frequency (MHz).

Instruction: An action executed by a processor, such as a mathematical operation or a memory operation.

Register: A small and extremely fast location for storing data or instructions in the processor.

Functional Unit (FU): A hardware element that performs an operation on an operand or pair of operations. Common FUs are ADD, MULT, INV, SQRT, etc.

Pipeline: A Technique enables multiple instructions to be overlapped during execution.

Superscalar: Multiple instructions are possible per clock period.

Flops: Floating point operations per second.

Cache: A Fast memory in the processor which keep instructions and data close to functional units so processor can execute more instructions more rapidly.

SRAM: Static Random Access Memory (RAM). Very fast (~10 nanoseconds), made using the same kind of circuitry as the processors, so speed is comparable.

DRAM: Dynamic RAM. Longer access times (~100 nanoseconds), but hold more bits and are much less expensive (10x cheaper).

Memory hierarchy: The hierarchy of memory in a parallel system, from registers to cache to local memory to remote memory.

Networks Latency: How long does it take to start sending a "message"? Measured in microseconds.

Networks Processors: How long does it take to output results of some operations, such as floating point add, divide etc., which are pipelined?)

Networks Bandwidth: What data rate can be sustained once the message is started? Measured in Mbytes/sec or Gbytes/sec

## Types of Clusters/Processors

Symmetric Multiprocessors (SMPs) connect processors to global shared memory using either bus or crossbar. It provides simple programming model, but has problems with buses can become saturated and crossbar size must increase with number of processors. Problem grows with number of processors, limiting maximum size of SMPs. Programming models are easier since message passing is not necessary. The techniques are auto-parallelization via compiler options, loop-level parallelism via compiler directives, OpenMP, and pthreads.

In MPP, each processor has its own memory and is not shared globally but the processors adds another layer to memory hierarchy (remote memory). The processor/memory nodes are connected by interconnect network using many possible topologies. The processors must pass data via messages so the communication overhead can be minimized. Many vendors have custom interconnects that provide high performance for their MPP system such as Gigabit Ethernet, Fast Ethernet, etc.

Clusters are similar to MPPs with processors and memory. The processor performance must be maximized and memory hierarchy needs remote memory as no shared memory for message passing to avoid the communication overhead.

Clusters are different from MPPs as commodity processors including interconnect and OS with multiple independent systems and separate I/O systems. The advantages of clusters are inexpensive, fastest processors first, potential for true parallel I/O and high availability while the disadvantages are less mature software (programming and system), more difficult to manage (changing slowly), lower performance interconnects (not as scalable to large number).

Distributed Memory Programming provides message passing using MPI, MPI-2 and active/one-sided messages.

There are two types of parallelism i.e., data and task. Each processor performs the same task on different sets or sub-regions of data in data parallelism. Each processor performs a different task in task parallelism. Most parallel applications fall somewhere on the continuum between these two extremes.

Example of data parallelism in a bottling plant, there are several ‘processors’, or bottle cappers, applying bottle caps concurrently on rows of bottles.

Example of task parallelism in a restaurant kitchen, there are several chefs, or ‘processors’, working simultaneously on different parts of different meals. A good restaurant kitchen also demonstrates load balancing and synchronization--more on those topics later.

A common form of parallelism used in developing applications was Master-Worker parallelism where a single processor is responsible for distributing data and collecting results (task parallelism) and all other processors perform same task on their portion of data (data parallelism).

According to Flynn’s Taxonomy, the computing systems are classified into the following broad categories:

- SISD: Single Instruction and Single Data
- SIMD: Single Instruction and Multiple Data
- MISD: Multiple Instruction and Single Data
- MIMD: Multiple Instruction and Multiple Data

The purpose of High-performance computing (HPC) platform is to provide the access to the compute resources remotely. The user can login remotely and submit compute their jobs either from the command line or through the GUI based interface provided to them. The computing systems are connected together through a high bandwidth data transfer and made available to the users in a queue-based job submission system. There are many open-source and commercial software packages installed.

### At IASRI, New Delhi

The National Agricultural Bioinformatics Grid in ICAR consists of an advanced HPC infrastructure at IASRI, New Delhi and moderate HPC facilities at the domain centres for undertaking research in the field of agricultural bioinformatics. Clusters are collections of computers that are connected together. The special sets of software are used to configure HPC environment. This set up has been named as Advanced Supercomputing Hub for Omics Knowledge in Agriculture (ASHOKA). The importance of HPC is rapidly growing because more and more scientific and technical problems are being studied on the huge data sets which require very high computational power as well. HPC offers environment for biologists, scientists, analysts, engineers and students to utilize the computing resources in making vital decisions, to speed up research and development, by reducing the execution time.

The following HPC infrastructure are set up under NAIP project NABG which are as follows in the form of clusters, network and storage.

**Types of Clusters**

a. 256 Nodes Linux Based Cluster with two masters
b. 16 Nodes Windows Based Cluster with one master
c. 16 Nodes GPGPU Based Linux Cluster with one master
d. 16 Nodes Linux based SMP system
e. 16 Nodes Linux Based Cluster at each of the five domains with one master

**Types of Networks**

a. High bandwidth network with low latency (Q-logic QDR InfiniBand switch)
b. Gigabit network for cluster administration and management
c. ILO3 Management Network

**Types of Storage**

a. Parallel File System (PFS) for computational purpose
b. Network Attached Storage (NAS) for user Home Directory
c. Archival Storage for back up.

The hardware configuration of the Head/Master node is as follows

| | | |
|---|---|---|
| Server Name | : | HP ProLiant DL380-G7 Server |
| Type of Processor | : | Intel Xeon X5675 3.07Ghz |
| Number of Processors | : | 2 |
| Core per Processor | : | 6 |
| Total memory (RAM) | : | 96GB |
| Memory per Core | : | 8GB |
| Hard Disk | : | 6*600GB SAS |
| OS | : | RHEL 6.2 (Linux) |

The hardware configuration of each compute node is as follows

| | | |
|---|---|---|
| Server Name | : | HP ProLiant SL390-G7 Server |
| Type of Processor | : | Intel Xeon X5675 3.07 Ghz |
| Number of Processors | : | 2 |
| Core per Processor | : | 6 |
| Total memory (RAM) | : | 96G |
| Memory per Core | : | 8GB |
| Hard Disk | : | 300GB SAS |
| OS | : | RHEL 6.2 (Linux) |

**Measuring Performance**

The memory is measured in terms of bytes i.e., Kilo ($2^{10}$ or $10^3$), Mega ($2^{20}$ or $10^6$) , Giga ($2^{30}$ or $10^9$) – Tera ($2^{40}$ or $10^{12}$), Peta ($2^{50}$ or $10^{15}$) , Exa ($2^{60}$ or $10^{18}$)

The computational performance is measured in Flop/s (Flop/s = floating point operations per second) i.e., Mega Flops, Tera Flops, Peta Flops etc.

One can calculate peak performance of the cluster using standard formula i.e. Cluster Performance = (Number of nodes) * (number of CPUs per node) * (number of cores per CPU) * (CPU speed in GHz) * (CPU instruction per cycle)

The grid has been established using the following network diagram as in figure 5.

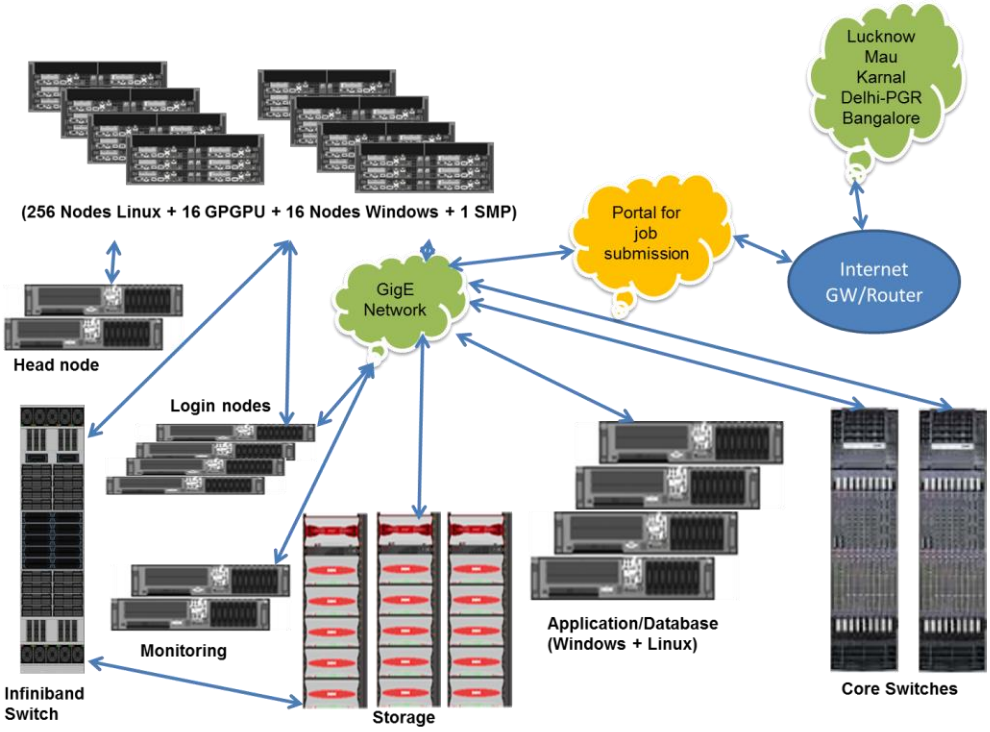


**Fig. 5: Network diagram of NABG Grid**

The hardware and software specifications of the SMP is as follows

| | | |
|---|---|---|
| Server Name | : | HP ProLiant DL 980 G7 |
| Type of Processor | : | Intel Xeon E7- 2830 2.13GHz |
| Number of Processors | : | 8 |
| Core per Processor | : | 8 |
| Total memory (RAM) | : | 1.5 TB |
| Hard Disk | : | 396 GB |
| OS | : | RHEL 6.2 |

A switched fabric computer network communications link, is being used in HPC and enterprise data centre with InfiniBand interconnect switch. The InfiniBand architecture specification defines a connection between processor nodes and high performance I/O nodes such as storage devices as in figure 6.

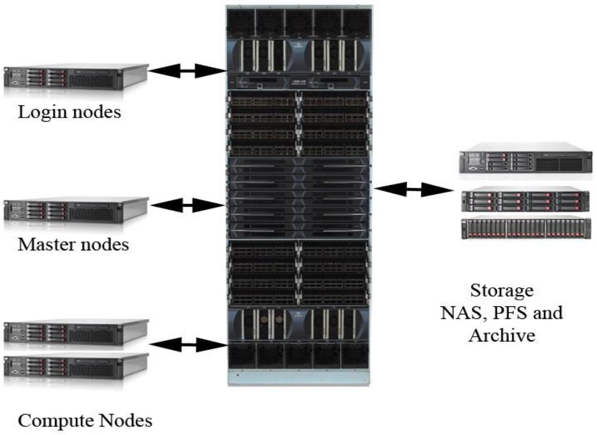


**Fig. 6: InfiniBand interconnect switch**

Main purpose of Ethernet network in the cluster is to provide services like cluster management, cluster monitoring, compute node deployment and many other things in figure 7.

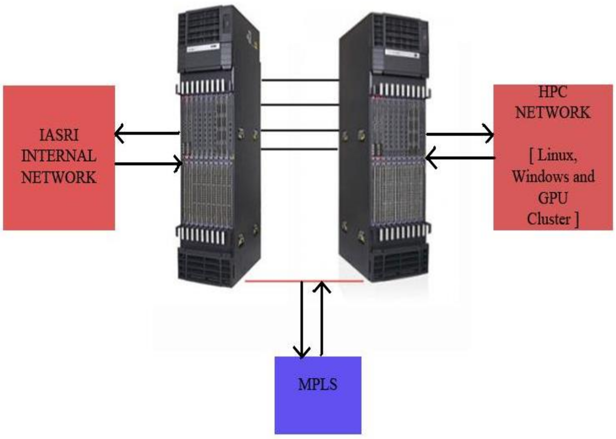


**Fig. 7: InfiniBand interconnect switch**

Different types of file system are configured for storing user’s data, running parallel jobs and archiving the important data. There are three types of storage (i) Network Attached Storage (NAS), (ii) Parallel File System (PFS) and (iii) Archival Storage.

The following challenges in bioinformatics are exists which essentially require the grid based architecture.

- Protein folding & structure prediction
- Homology search
- Multiple alignment
- Genomic sequence analysis
- Gene finding
- Gene expression data analysis
- Drug discovery
- Phylogenetic inference
- Computational genomics, proteomics
- Computational evolutionary biology

# Introduction to Linux Basics

**S. B. Lal**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

The Linux operating system is basically a variant of the UNIX operating system, and Linux has probably all that UNIX offers and more. It is a multi-user, multitasking, network operating system which also has a user friendly Graphical User Interface (GUI).

Every desktop computer uses an operating system. The most popular operating systems are Windows, Mac OS, UNIX, Linux.

## What is an Operating System?

An operating system is the first piece of software that the computer executes when a system is turned on. The operating system loads itself into memory and begins managing the resources available in the computer. It provides those resources to other applications that the user wants to run. Typical services that an operating system provides include:

A task scheduler - The task scheduler is able to allocate the execution of the CPU to a number of different tasks. Some of those tasks are the different applications that the user is running, and some of them are operating system tasks.

A memory manager - The memory manager controls the system's RAM and normally creates a larger virtual memory space using a file on the hard disk.

A disk manager - The disk manager creates and maintains the directories and files on the disk. When a file is needed, the disk manager makes it available from the disk.

A network manager - The network manager controls all data moving between the computer and the network.

Other I/O services manager - The OS manages the keyboard, mouse, video display, printers, etc.

Security manager - The OS maintains the security of the information in the computer's files and controls who can access the computer.

An operating system normally also provides the default user interface for the system. The standard "look" of Windows 98 includes the Start button, the task bar, etc. The Mac OS provides a completely different look and feel for Macintosh computers.

To understand why Linux has become so popular, it is helpful to know a little bit about its history.

## Background on Linux

Linux, a UNIX-like operating system, is based on Minix and has been invented by Linus Benedict Torvalds in 1991. The following is an excerpt of a newsgroup, called "comp.os.minix" where Linus posted this text on 08/01/91: "...As I mentioned a month ago, I'm working on a free version of a Minix-look-alike for AT-386 computers. It has finally reached the stage where it's even usable (though may not be, depending on what you want), and I am willing to put out the sources for wider distribution. It is just version 0.02... but I've successfully run bash, gcc, gnu-make, gnu-sed, compress, etc. under it."

Linux is a free version of UNIX that continues to be developed by the cooperative efforts of volunteer groups of programmers, primarily on the Internet, who exchange code, report bug, and fix problems in an open-ended environment. As a result, the world now has a powerful, robust, and full-featured operating system that continues to change and grow.

In other words, Linux is little bit harder to manage than something like Windows, but offers more flexibility and configuration options.

Linux is licensed under the GPL (General Public license) from the GNU organization, under which the kernel is provided with the source code, and is available for free. As a result, Linux is considered to be more secure and stable than closed source or proprietary systems like Windows because anyone can analyse the source code written in the C language and find bugs or add new features. One important point that should be noted is that even though the source is free, anyone is allowed to sell it for profit.

Linux is known as an *open source* operating system and also called *free software* because everything about Linux is accessible to the public and is freely available to anyone. Since the Linux source code is available, anyone can copy, modify, and distribute this software. This allows for various companies such as SuSE, Red Hat, Caldera and others to sell and distribute Linux; however, at the same time, these companies must keep their Linux distribution code open for public inspection, comment, and changes. Despite of the command-line origins of Linux, these distributing companies are working to make the Graphical User Interface (GUI).

### The GNU General Public License

To make software free, you need a license that defines the rights and the limits, that have to be regarded by the open source developer that wants to obtain, edit and eventually redistribute your source code. Because of that exists the GNU GPL (General Public License). Of course, there are also other licenses, but today’s most open source programs are distributed under this popular license.

The GNU project was started in 1984 and “GNU is recursive acronym for “GNU’s Not Unix”; The Free Software Foundation, which stands for the freedom, the security and the protection of free source code therefore founded this kind of license, designed to protect open source code. GNU is also founder and maintainer of many software packages for the Linux operating system, such as basic tools and file system software.

### Is Linux Right for you?

It depends on you and what you would like to do. Linux is not an all-purpose operating system and it would probably be more suited for some people and not so pleasing for others. If you are a person using your computer for some entertainment at home and are satisfied with your Windows system there are no compelling reasons for switching over to Linux, but you do have a choice now. There are several other reasons to consider Linux. Linux is not just a simple operating system. It is an entire server and desktop environment, equipped with add-ons, GUI tools and interfaces, and supplementary programs.

You can use Linux at home and even in college to understand the commands and even the internal workings of UNIX systems.

### Distributions

When people use the name Linux they are probably referring to a particular distribution of Linux. There are several software packages provided for Linux over the Internet but selecting and downloading one is a complicated task not necessarily manageable for new users who want to try out Linux. This is exactly where a distribution kicks in.

A distribution is a set of software packages that are tested and provided on CD by a company for a small fee just like Windows. The advantages of using distributions are the support and manuals, as well as the fact that Linux can be specialized for use in a particular area. For example, if you would like using Linux for embedded systems a distribution may offer just the right amount of required software, leaving out optional things like the graphical user interface. So you get what you want instead of a general package for all users.

The mainstream distributions, which are seemingly popular, are RedHat, SuSE, Caldera and Debian. Among these distributions RedHat seems to be most widespread.

Caldera is probably more suited for those who are already using Windows. SuSE is a German based distribution known for its large number of bundled packages and support. Debian is unique because its not owned by a company and it’s a non-profit volunteer-based distribution developed solely by users.

### Getting Started with Linux

Once the installation is complete, the system will reboot and start up with Linux. There are a series of messages on the screen while booting of the system regarding the hardware enabled, services started etc. After a while, the system will display a login: prompt. You can now log in.

Some systems are configured to start graphical mode with a box in the middle containing both login: and Password: prompts. Press *[CTRL]-[ALT]-[F1]* to switch to the virtual console (text login screen), where you can log in to the system in the usual way.

### Accounts and Privileges

Linux is a multi-user system, meaning that many users can use one Linux system simultaneously, from different terminals. So to avoid confusion, each user's workspace must be kept separate from the others.

Even if a particular Linux system is a stand-alone personal computer with no other terminals physically connected to it, it can be shared by different people at different times, making the separation of user workspace is important.

This separation is accomplished by giving each individual user an *account* on the system. You need an account in order to use the system; with an account you are issued an individual workspace to use, and a unique *username* that identifies you to the system and to other users. It is the name along with the password by which the system will recognize the user.

### Logging into the System

To begin a session on a Linux system, you need to *log in*. Do this by entering your username at the login: prompt on your terminal, and then entering your password when asked.

Every Linux system has its own name, called the system's *hostname*; a Linux system is sometimes called a *host*, and it identifies itself with its hostname at the login: prompt. It's important to name your system -- like a username for a user account, a hostname gives name to the system you are using (and it becomes especially important when putting the system on a network). The system administrator usually names the system when it is being initially configured (the hostname can be changed later; its name is kept in the file `/etc/hostname'). The name of the terminal you are connecting from is displayed just after the hostname.

To log in to the system, type your username (followed by) at the login: prompt, and then type your password when asked (also followed by); for security purposes, your password is not displayed on the screen when you type it.

Once you've entered your username and password, you are "logged in" to the system. You can then use the system and run commands.

As soon as you log in, the system displays the contents of `/etc/motd', the "Message of the Day" file. The system then displays the time and date of your last login, and reports whether or not you have electronic mail waiting for you. Finally, the system puts you in a *shell*---the environment in which you interact with the system and give it commands. Bash is the default shell on most Linux systems.

The dollar sign (`$') displayed to the left of the cursor is called the *shell prompt*; it means that the system is ready and waiting for input. By default, the shell prompt includes the name of the current directory.

### Logging Out of the System

To end your session on the system, type *logout* at the shell prompt. This command logs you out of the system, and a new login: prompt appears on your terminal.

- To log out of the system

  $ *logout*

You can also logout by just pressing *Ctrl+d*.

Logging out of the system frees the terminal you were using and ensures that nobody can access your account from this terminal.

### Console Basics

A Linux *terminal* is a place to put input and get output from the system, and usually has at least a keyboard and monitor.

When you access a Linux system by the keyboard and monitor that are directly connected to it, you are said to be using the *console* terminal.

Linux systems feature *virtual consoles*, which act as separate console displays that can run separate login sessions, but are accessed from the same physical console terminal. Linux systems are configured to have seven virtual consoles by default. When you are at the console terminal, you can switch between virtual consoles at any time, and you can log in and use the system from several virtual consoles at once.

### Switching Between Consoles

To switch to a different virtual console, press **[ALT]-[F*n*],** where *n* is the number of the console to switch to.

- To switch to the fourth virtual console, press **[ALT]-[F4].**

You can also cycle through the different virtual consoles with the left and right arrow keys. To switch to the next-lowest virtual console, press [ALT]-[←]and to the next-highest virtual console, press **[ALT]-[→].**

- To switch from the fourth to the third virtual console, press **[ALT]-[←]**

The seventh virtual console is reserved for the X Window System. If X is installed, this virtual terminal will never show a login: prompt, but when you are using X, this is where your X session appears. If your system is configured to start X immediately, this virtual console will show an X login screen.

You can switch to a virtual console from the X Window System using [CTRL] in conjunction with the usual [ALT] and function keys. This is the only console manipulation keystroke that works in X.

- To switch from X to the first virtual console, press: [CTRL]-[ALT]-[F1]

### Running a Command

A *command* is the name of a tool that performs a certain function along with the options and arguments. Commands are case sensitive.

To run the hostname command just type the command in front of prompt ($)

```
$ hostname
```

Options always begin with a hyphen character, `-', which is usually followed by one alphanumeric character. Always separate the command, each option, and each argument with a space character.

*Long-style* options begin with two hyphen characters (`--').

For example, many commands have an option, `--version', to output the version number of the hostname.

```
$ hostname --version
```

Sometimes, an option itself may take an argument. For example, hostname has an option for specifying a file name to use to read the hostname from, `-F'; it takes as an argument the name of the file that hostname should read from. To run hostname and specify that the file `host.info' is the file to read from

```
$ hostname -F host.info
```

### Changing Your Password

To change your password, use the passwd command. It prompts you for your current password and a new password to replace it with. You must type it exactly the same way both times, or passwd will not change your password.

```
$ passwd  username
```

### Listing Your Username

Use whoami to output the username of the user that is logged in at your terminal.

```
$ whoami
```

### Listing Who Is on the System

Use who to output a list of all the users currently logged in to the system. It outputs a minimum of three columns, listing the username, terminal location, and time of login for all users on the system. A fourth column is displayed if a user is using the X Window System.

```
$ who
abc    tty1     Oct 20 20:09
def     tty2     Oct 21 14:37
def     ttyp1    Oct 21 15:04 (:0.0)
$
```

The output in this example shows that the user abc is logged in on tty1 (the first virtual console on the system), and has been on since 20:09 on 20 October. The user def is logged in twice -- on tty2 (the second virtual console), and ttyp1, which is an X session with a window location of `(:0.0)'.

### Listing the Last Times a User Logged In

Use last to find out who has recently used the system, which terminals they used, and when they logged in and out.

```
$ last abc
```

### Listing System Activity

When you run a command, you are starting a *process* on the system, which is a program that is currently executing. Every process is given a unique number, called its *process ID*, or "PID."

Use ps to list processes on the system. By default, ps outputs 5 columns: process ID, the name of the terminal from which the process was started, the current status of the process (including `S' for *sleeping*, meaning that it is on hold at the moment, `R' meaning that it is running, and `Z' meaning that it is a process that has already died), the total amount of time the CPU has spent on the process since the process started, and finally the name of the command being run.

### Listing Your Current Processes

Type *ps* with no arguments to list the processes you have running in your current shell session.

```
$ ps
  PID TTY STAT TIME COMMAND
  193   1 S    0:01 -bash
  204   1 S    0:00 ps
$
```

### Listing All of a User's Processes

To list all the running processes of a specific user, use ps and give the username to list as an argument with the `-u' option.

```
$ ps -u abc
```

### Listing All Processes on the System

To list all processes running by all users on the system, use the `aux' options.

```
$ ps aux
```

### Listing Processes by Name or Number

To list processes whose output contains a name or other text to match, list all processes and pipe the output to grep. This is useful for when you want to see which users are running a particular program or command.

To list all the processes whose commands contain reference to an `sbin' directory in them

```
$ ps aux | grep sbin
```

To list any processes whose process IDs contain a 13 in them

```
$ ps aux | grep 13
```

To list the process, which corresponds to a process ID, give that PID as an argument to the `-p' option (PID is 344 )

```
$ ps -p 344
```

### Finding the System Manual of a Command

Use the man command to view a page in the system manual. As an argument to man, give the name of the program whose manual page you want to view.

```
$ man ps
```

Use the up and down arrow keys to move through the text. Press [Q] to stop viewing the manual page and exit man.

### Working with Shell

S*hell* is a program that reads your command input and runs the specified commands. The shell environment is the most fundamental way to interact with the system -- you are said to be in a shell from the very moment you've successfully logged in to the system.

The `$' character preceding the cursor is called the *shell prompt*; it tells you that the system is ready and waiting for input.

If your shell prompt shows a number sign (`#') instead of a `$', this means that you're logged in with the superuser, or root, account. Beware: the root account has complete control over the system; one wrong keystroke and you might accidentally break it something awful. You need to have a different user account for yourself, and use that account for your regular use.

Every Linux system has at least one shell program, and most have several. The standard shell on most Linux systems is bash( "Bourne again shell").

### Running a List of Commands

To run more than one command on the input line, type each command in the order you want them to run, separating each command from the next with a semicolon (`;'). For example, to clear the screen and then log out of the system

```
$ clear; logout
```

### Redirecting Input and Output

The shell moves text in designated "streams." The *standard output* is where the shell streams the text output of commands -- the screen on your terminal, by default. The *standard input*, typically the keyboard, is where you input data for commands. You can redirect these streams -- to a file, or even another command -- with *redirection*.

**Redirecting Input to a File**

To redirect standard input to a file, use the `<' operator. To do so, follow a command with < and the name of the file it should take input from. For example, to redirect standard input for ls -l to file `listing'

```
$ ls -l < listing
```

**Redirecting Output to a File**

Use the `>' operator to redirect standard output to a file. If you redirect standard output to an existing file, it will overwrite the file, unless you use the `>>' operator to *append* the standard output to the contents of the existing file. For example, to append the standard output of *ls -l* to an existing file `commands'

```
$ ls -l>> commands
```

**Redirecting Output to another Command's Input**

*Piping* is to connect the standard output of one command to the standard input of another. You do this by specifying the two commands in order, separated by a vertical bar character, `|' (also called as a "pipe"). Commands built in this fashion are called *pipelines*.

For example, it's often useful to pipe commands that display a lot of text output to more for perusing text.To pipe the output of apropos bash shell shells to less

```
$ ls –l  | more
```

**Managing Jobs**

The processes you have running in a particular shell are called your *jobs*. You can have more than one job running from a shell at once, but only one job can be active at the terminal, reading standard input and writing standard output. This job is the *foreground* job, while any other jobs are said to be running in the *background*.

The shell assigns each job a unique *job number*. Use the job number as an argument to specify the job to commands. Do this by giving the job number preceded by a `%' character.

**Suspending a Job**

Type *Ctrl+z* to suspend or stop the foreground job. This is useful when you want to do something else in the shell and return to the current job later. The job stops until you either bring it back to the foreground or make it run in the background.

For example, if you are finding a file at Linux partition from root (/), typing *Ctrl+z* will suspend the find program and return you to a shell prompt where you can do something else. The shell outputs a line giving the job number (in brackets) of the suspended job, the text `Stopped' to indicate that the job has stopped, and the command line itself, as shown here:

```
[1]+  Stopped                find / -name abc
```

In this example, the job number is 1 and the command that has stopped is `find / -name abc'. The `+' character next to the job number indicates that this is the most recent job.

If you have any stopped jobs when you log out, the shell will tell you this instead of logging you out:

```
$ logout
There are stopped jobs.
$
```

At this point you can list your jobs, stop any jobs you have running and then log out.

**Putting a Job in the Background**

New jobs run in the foreground unless you specify otherwise. To run a job in the background, end the input line with an ampersand (`&'). This is useful for running non-interactive programs that perform a lot of calculations. To run the command find / -name abc > shell-commands as a background job

```
$ find / -name abc > shell-commands &
[1] 6575
$
```

The shell outputs the job number (in this case, 1) and process ID (in this case, 6575), and then returns to a shell prompt. When the background job finishes, the shell will list the job number, the command, and the text `Done', indicating that the job has completed successfully:

```
[1]+  Done                  find / -name abc >shell-commands
```

To move a job from the foreground to the background, first suspend it  and then type *bg* (for "background").

- For example, to start the command *find / -name abc > shell-commands* in the foreground, suspend it, and then specify that it finish in the background, you would type:

```
$ find / -name abc > shell-commands
Ctrl+z


[1]+  Stopped               find / -name abc >shell-commands
$ bg
[1]+ find / -name abc &
$
```

If you have suspended multiple jobs, specify the job to be put in the background by giving its job number as an argument. TFor example, to run job 4 in the background

```
$ bg %4
```

**Putting a Job in the Foreground**

Type *fg* to move a background job to the foreground. By default, fg works on the most recent background job. For example, to bring the most recent background job to the foreground

```
$ fg
```

To move a specific job to the foreground when you have multiple jobs in the background, specify the job number as an option to fg. To bring job 3 to the foreground

```
$ fg %3
```

### Listing Your Jobs

To list the jobs running in the current shell, type *jobs*.

```
$ jobs
[1]-  Stopped            find / -name abc >shell-commands
[2]+  Stopped            find / -name abc >bash-commands
$
```

This example shows two jobs--- *find / -name abc > shell-commands* and *find / -name abc > bash-commands*. The `+' character next to a job number indicates that it's the most recent job, and the `-' character indicates that it's the job *previous* to the most recent job. If you have no current jobs, jobs returns nothing.

### Stopping a Job

Typing *Ctrl+c* interrupts the foreground job before it completes, exiting the program. To interrupt *cat*, a job running in the foreground

```
$ cat
Ctrl+c
$
```

Use kill to interrupt ("kill") a background job, specifying the job number as an argument. To kill job number 2

```
$ kill %2
```

### Command History

Your command *history* is the sequential list of commands you have typed, in the current or previous shell sessions. The commands in this history list are called *events*.

By default, bash remembers the last 500 events, but this number is configurable.

Your command history is stored in a text file in your home directory called `.bash_history'; you can view this file or edit it like you would any other text file.

### Viewing Your Command History

Use history to view your command history.  To view your command history

```
$ history
1 who
2 apropos shell >shell-commands
3 apropos bash >bash-commands
4 history
$
```

This command shows the contents of your command history file, listing one command per line prefaced by its *event number*. Use an event number to specify that event in your history. To search your history for the text `find'

```
$ history | grep find
```

### Specifying a Command from Your History

You can specify a past event from your history on the input line, in order to run it again.

The simplest way to specify a history event is to use the up and down arrow keys at the shell prompt to browse your history. The up arrow key takes you back through past events, and the down arrow key moves you forward into recent history. When a history event is on the input line, you can edit it as normal, and type to run it as a command; it will then become the newest event in your history.

To run a history event by its event number, enter an exclamation mark (`!') followed by the event number (1).

```
$ !1
```

# Biological Databases: An Overview

**K. K. Chaturvedi**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

Bioinformatics is the field of science in which biology, physics, chemistry, mathematics. Statistical and computer science, information and communication technology become a single discipline. It is emerging field that application of computer to collection, organization, storing, maintaining, accessing, sharing, analysis, interpretation and presentation of biological data (nucleotide and amino acids sequences, protein domains, protein structures) which helps to accomplishing life science research.

The potential flood of sequence data and the rapidly evolving database technologies empowered researchers to establish international DNA data banks in the early 1980s. Today, we have massive sequence data in the public biological databases due to concerted effort at a number of molecular biology laboratories throughout the world, and the internet and computer technologies. At the beginning, the main concern of bioinformatics was the creation and maintenance of database to store nucleotide and amino acid sequences with wen based interfaces user can access existing data and submitting new data to the database. Hence, database creation and maintenance is major components in bioinformatics. Now, emphasis has shifted to decipher the functional, structural and evolutionary clues encoded in the languages of biology, in which sequences is represented by as sentence, motifs and patterns are by words and nucleotides and amino acids are by letters. However, database design and management is core area in bioinformatics.

Data represents facts or value of results and relations between them have the capacity to represent information (Figure 1). Patterns of relationship between information have the capacity to represent knowledge. Each data is assigned to one data type, which indicates possible relationship with other data. For example; text, integer, float/double, character, time, date and binary.

A **database** is a collection of data organized in the way which can be easily, stored, accessed and managed. Database system is amalgamation of database, database management system and users. (Fig. 1)

## Types of Database models

In mid of 1960 the “database” word was first introduced with direct-access-storage. Charles Bachman has introduced Integrated Data Store (IDS), founded, the group “Database Task Group” responsible for the creation and standardization of COBOL. In 1971 the DTG within CODASYL (Conference on Data Systems Languages) delivered standard for database, which generally became known as the "Codasyl approach”, this led to network database. Same period IBM was developed IMS (Information Management System), which is similar to Codasyl approach and used hierarchical model of data. Edgar Codd worked at IBM in San Jose, California and he was unhappy with the above two models. He wrote a number of papers those illustrated a new approach based on relational algebra for construction of database that led to a well accepted Relational Model of Data for Large Shared Data Banks. This based on concept relational algebra. There are three main types of database models; 1) Network Model, 2) Hierarchy Model, and 3) Relational Model. Main objective of these models is integration of data, which is process of combining data of different sources under single query interface.

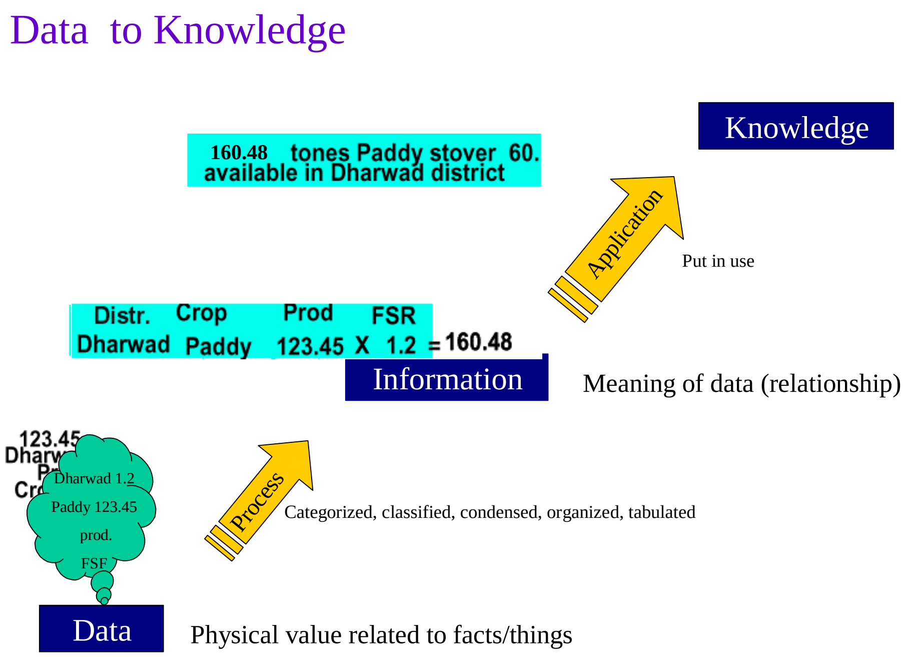

Data to Knowledge
Knowledge
160.48 tones Paddy stover 60.
available in Dharwad district
Application
Put in use
Distr. Crop Prod FSR
Dharwad Paddy 123.45 X 1.2 = 160.48
Information
Meaning of data (relationship)
123.45
Dharw
P
Cr
Dharwad 1.2
Paddy 123.45
prod.
FSF
Process
Categorized, classified, condensed, organized, tabulated
Data
Physical value related to facts/things

**Fig. 1: Data to knowledge**

### Network Database Model

This model visualizes data in a flexible way of representing objects and their relationships. Its distinguishing feature is that the schema, viewed as a graph in which object types are nodes and relationship types are arcs, is not restricted to being a hierarchy or lattice.

### Hierarchical database model

This model is a data model in which the data is organized into a reverse tree-like structure. In this data can be represented as parent and child relationships by 1 to many relationships that each parent can have many children, but each child has only one parent. All attributes of a specific record are listed under an entity type.

### Relational Database Model

In this model, database structure is represented in terms of tuples (rows), grouped into relations (tables) and values in each columns of tuple are represented as attributes values (data) and identified solely by the attribute name (Field).

### Major Components and Architecture of Database System

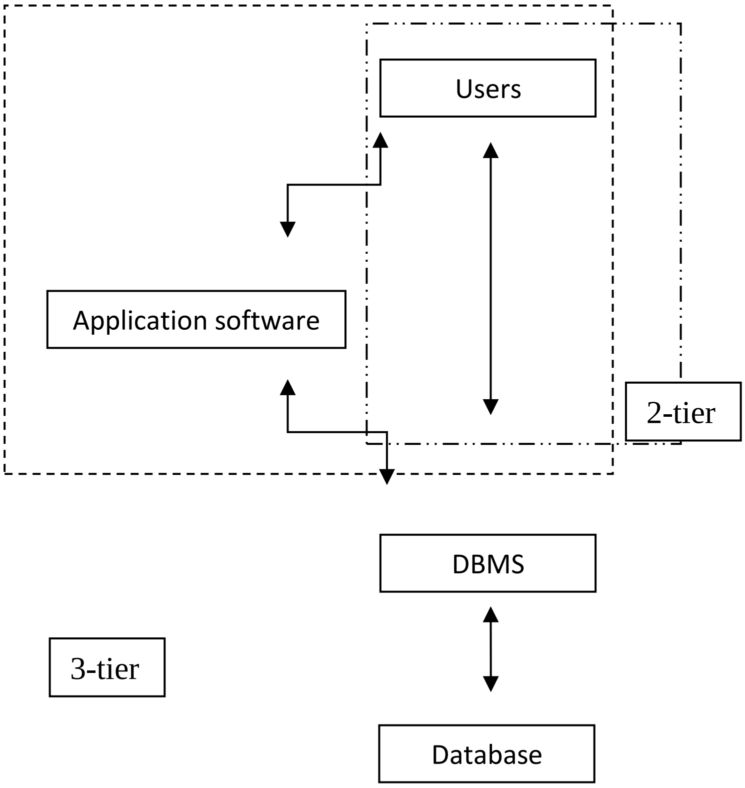


**Fig. 2: Architecture of Database**

- Users: DB Administrator, Developer and end-user.
- Application: Application software to any specific domain.
- DBMS: Software for creation, insertion, deletion and modification.
- Database: Collection of data

Database architecture logically divided in to two types

- 2 - tier: End-user < -- > DBMS; Here end-user/client can directly communicate with database server.
- 3- tier: End-user < -- > Application Software < -- > DBMS; Here end-user/client will communicate with database server through application tools.

### Basic Concept of DataBase Management System (DBMS)

Database Management Systems (DBMS) is specially designed applications software that designed to interact with the user, other applications and database(s) to capture and analyse data. The DBMS have facilities to allow the definition, creation, querying, update, and administration of databases. Well-known DBMSs include MySQL, PostgreSQL, Microsoft SQL Server, Oracle, SAP, MS Access, FoxPro, IBM DB2/TeraByte, etc. Now database have generally portable across different DBMS by using standards such as SQL and ODBC or JDBC to allow a single application to work with more than one database.

### Major functions of DBMS

- Data definition: Defining new data structures, removing and modifying the existing structure.
- Update: Inserting, modifying, and deleting data.
- Retrieval: Obtaining information for end-user queries or for applications.
- Administration: Registering and monitoring users, enforcing data security, monitoring performance, maintaining data integrity, dealing with concurrency control, and recovering information if the system fails.

**Benefits of DBMS**

- Segregation of work to end-users
- Easy editing, maintenance and retrieval
- Minimizing data duplication
- Reducing time in development and maintenance
- Data security
- Multiple user accessing
- Backup and recovery

**Relational Database Management System (RDBMS)**

A Relational database Management System (RDBMS) is a database management system to manage relational database based on relation database model as discussed above, which is introduced by E. F. Codd. In this data is represented in terms of tuples (rows) Relational database is collection of tables, table is consist of rows usually called as records and columns called as field or attributes, and columns are identified by unique name. Table is most simplest and fundamental unit of data storage. Each table has its own primary key (one or more fields), which ensures that uniqueness of each record with set of fields. The keys are very important part of relational database. They are used to establish and identify relationship between tables. The RDBMS supports Structured Query Language (SQL).

**Normalization**

Normalization is a systematics pre-process of decomposing tables to eliminate data redundancy. This will help to easy insertion, updation and deletion. Normalization rule are divided into following form

- First Normal Form: Row cannot contain repeating group of data.
- Second Normal Form: Remove partial dependency between columns
- Remove transitive functional dependency
- Boyce and Codd Normal Form: This deals with certain anomaly that is not handled by3NF.

**Entity-Relationship (E-R) Diagram**

ER diagram is visual diagrammatic representation of data with standard symbols and notation, which describes how data is related to each other (Fig. 3).

Major symbols and notations

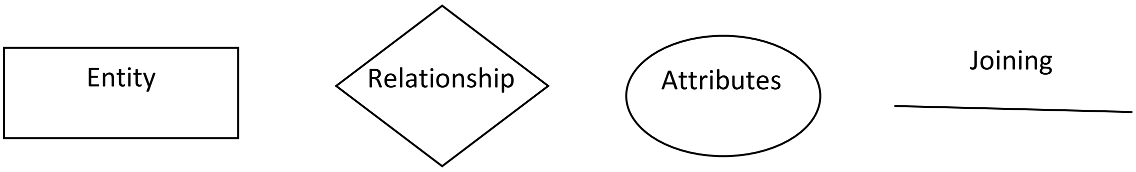


**Fig. 3: Symbols and Notations**

Entity may be any object, person, place and etc. Attributes are features or characteristics. For Example livestock census statistics is shown in table 1.

**Table 1: Livestock data before normalization**

| State | State Capital | Dist | Dist Head Qrts | Year | Animal | Category | Population | Population (000) |
|---|---|---|---|---|---|---|---|---|
| Karnataka | Bangalore | Dharwad | Dharwad | 2007 | Cattle | < 1 year | 14355 | 14.356 |
| Karnataka | Bangalore | Dharwad | Dharwad | 2007 | Cattle | 1-2.5 year | 24675 | 24.675 |
| Karnataka | Bangalore | Dharwad | Dharwad | 2007 | Cattle | >2.5 year | 44355 | 44.355 |
| Karnataka | Bangalore | Uttar Kannada | Karwar | 2007 | Cattle | < 1 year | 45255 | 45.255 |
| Karnataka | Bangalore | Uttar Kannada | Karwar | 2007 | Cattle | 1-2.5 year | 56555 | 56.555 |
| Karnataka | Bangalore | Uttar Kannada | Karwar | 2007 | Cattle | >2.5 year | 1836 | 1.836 |

The ER diagram for the table 1 is shown in Fig. 4.

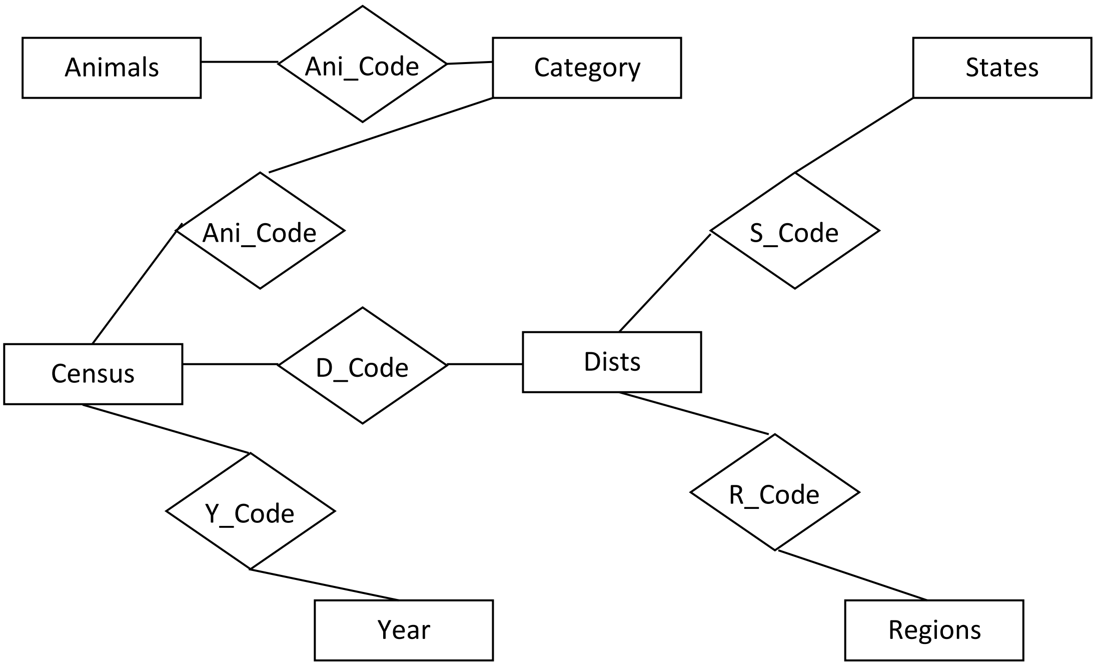


**Fig. 4: ER-Diagram**

The relationships of the tables are shown in Fig. 5.

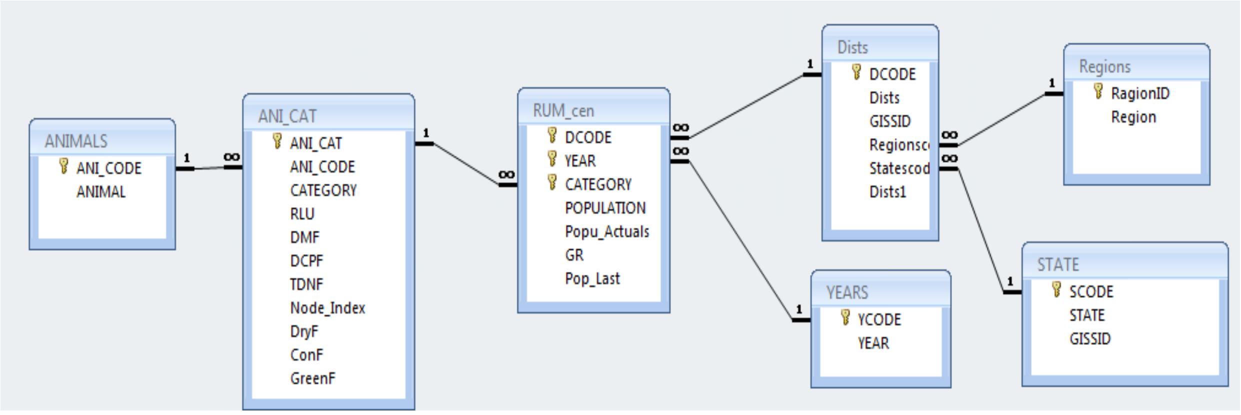


**Fig. 5: Relationship diagram from MS Access**

## Structured Query Language (SQL)

SQL is a tool for communicate with database. SQL is a plat form independent common language is used to perform all types of data operation such as data defining, storing and managing in RDBMS database concept. Now, all RDBMS software employs this language as standard database language. Some of the sample commands are mentioned in table 2.

**Table 2: Sample of SQL commands**

| Command | Description | Syntax |
|---|---|---|
| **Data Definition** | | |
| create | To create new table or database | CREATE TABLE "tablename" ("column1_name" "data type",<br>"column2_name" "data type", “. . . ") |
| alter | For alteration | ALTER TABLE table_name ADD column_name datatype;<br>ALTER TABLE table_name DROP COLUMN column_name;<br>ALTER TABLE table_name MODIFY COLUMN column_name datatype; |
| drop | To drop a table | DROP TABLE "tablename" |
| rename | To rename a table | RENAME TABLE tbl_name TO new_tbl_name; |
| **Data Manipulation** | | |
| Insert | To insert a new row | INSERT INTO tablename" (column1,... column_last) VALUES (value1, ... value_last); |
| update | To update existing row | UPDATE "tablename" SET "columnname" = "newvalue" [,"nextcolumn" = "newvalue2"...] WHERE "columnname" OPERATOR "value" [AND\|OR "column" OPERATOR "value"]; |
| delete | To delete a row | DELETE FROM "tablename" WHERE "columnname" OPERATOR "value" [AND\|OR "column" OPERATOR "value"]; |
| **Transaction control** | | |
| commit | To permanently save | COMMIT; |
| rollback | To undo change | ROLLBACK; |
| savepoint | To save temporarly | SAVEPOINT SAVEPOINT_NAME; |
| **Data query** | | |
| select | | SELECT[ALL\| DISTINCT] *column1* [,*column2*] FROM *table1* [,*table2*] [WHERE "conditions"] [GROUP BY "column-list"] [HAVING "conditions] [ORDER BY "column-list" [ASC \| DESC] ] |

**Biological Database**

Life science is a field which generates an enormous amount of un-integrated data. Biological databases are collection of life sciences data, information and knowledge collected from different sources such as scientific experiments, published literature, high-throughput experiment, and computational & statistical analyses in form text, numbers, videos, images and diagrams. These data are broadly classified into four categories based type of data such as literature, sequences, structures and micro-array data. Also area wise classified into Genomics, Proteomics, Metabolomics, and Micro-array (gene expression) and Phylogenetics.

**Primary Genomic Databases**

- GenBank (National Center for Biotechnology Information) url: http://www.ncbi.nlm.nih.gov/genome
- DNA Data Bank of Japan (National Institute of Genetics) url: http://www.ddbj.nig.ac.jp/
- European Nucleotide Archive (European Bioinformatics Institute) url: http://www.ebi.ac.uk/ena/

**Primary Protein Databases**

- Uniprot (Universal Protein Resources) url:www.uniprot.org
- PDB url: www.rcsb.org/pdb/

**Metabolomics databases**

- META Cyc url: http://metacyc.org/
- KEGG: url : http://www.genome.jp/kegg/pathway.html
- Plant Metabolic Network (PMN) url: http://www.plantcyc.org/

**Phylogenetics databases**

- PhylomeDB url: http://phylomedb.org
- TreeBASE url: http://treebase.org

**Microarray Database**

- EMBL-EBI microarray database array express url: http://www.ebi.ac.uk/arrayexpress/
- Stanford University database url: http://smd.princeton.edu/
- Gene expression Omnibus (GEO) (NLM) url: http://www.ncbi.nlm.nih.gov/geo/
- ExpressDB - Harvard url: http://arep.med.harvard.edu/ExpressDB/

Similarly many bioinformatics databases such as Compound-Specific Databases, Comprehensive Metabolomic Database, drug database, RNA database, SNP database, Microsatellites, Literature database, Crystallographic database, NMR spectra database, Carbohydrate structure databases, Protein-protein interactions database, Signal transduction pathway databases, primer databases, Taxonomic databases and etc.

# Sequence Analysis

**S. B. Lal**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## 1. Introduction

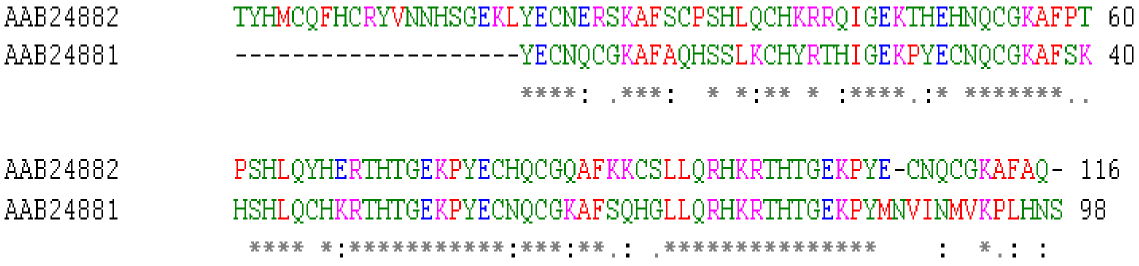
Since the development of high-throughput methods for production of gene and protein sequences during 90s, the rate of addition of new sequences to the databases increases very rapidly. However, comparing sequences with known functions with these new sequences is one way of understanding the biology of that organism from which the new sequence comes. Thus, sequence analysis can be used to study of the similarities between the compared sequences. Now a days, there are many tools and techniques that provide the sequence comparisons (sequence alignment) and analyze the alignment to understand the biology.

Sequence analysis in molecular biology and bioinformatics is an automated, computer-based examination of characteristic fragments, e.g. of a DNA strand. It basically includes relevant topics:

1. The comparison of sequences in order to find similarity and dissimilarity in compared sequences (sequence alignment)
2. Identification of gene-structures, reading frames, distributions of introns, exons and regulatory elements
3. Finding and comparing point mutations or the single nucleotide polymorphism (SNP) in organism in order to get the genetic marker.
4. Revealing the evolution and genetic diversity of organisms.
5. Functional annotation of genes.

Sequence alignment is a way to identify regions of similarity in DNA, RNA, or protein sequences that may be a consequence of functional, structural, or evolutionary relationships between the sequences. Aligned sequences of nucleotide or amino acid residues are typically represented as rows within a matrix. If two sequences share a common ancestor for the alignment, mismatches can be interpreted as point mutations and gaps as indels (that is, insertion or deletion mutations). Thus, a letter or a stretch of letters may be paired up with dashes in the other sequence to signify such an insertion or deletion. Homologous sequences may have different length, which is generally explained through insertions or deletions in sequences. Since an insertion in one sequence can always be seen as a deletion in the other one frequently uses the term "indel". In sequence alignments of proteins, the degree of similarity between amino acids sequence can be interpreted as a rough measure of how conserved a particular region or sequence motif is among lineages. The absence of substitutions, or the presence of only very conservative substitutions (that is, the substitution of amino acids whose side chains have similar biochemical properties) in a particular region of the sequence, suggest that this region has structural or functional importance. Although DNA and RNA nucleotide bases are more similar to each other than are amino acids, the conservation of base pairs can indicate a similar functional or structural role.

Very short or very similar sequences can be aligned by hand. However, most interesting problems require the alignment of lengthy, highly variable or extremely numerous sequences that cannot be aligned solely by human effort. Computational methods need to be developed for the alignment of a large pair of sequences. Computational approaches are of two categories: *global alignments* and *local alignments*. Global alignment is a form of global optimization that "forces" the alignment to span the entire length of all query sequences. Global alignment will be applied when the sequences are of similar lengths. Local alignments identify regions of similarity within long sequences. Local alignments are often preferable, but it consumes more time to calculate because of the additional challenge of identifying the regions of similarity in the local regions. Number of algorithms is being applied for the sequence alignment, including optimizing methods like dynamic programming, and heuristic algorithms or probabilistic methods designed for large-scale database search.



```
AAB24882        TYHMCQFHCRYVNNHSGEKLYECNERSKAFSCPSHLQCHKRRQIGEKTHEHNQCGKAFPT 60
AAB24881        --------------------YECNQCGKAFAQHSSLKCHYRTHIGEKPYECNQCGKAFSK 40
                                    ****: .***:  * *:** * :****.:* *******..

AAB24882        PSHLQYHERTHTGEKPYECHQCGQAFKKCSLLQRHKRTHTGEKPYE-CNQCGKAFAQ- 116
AAB24881        HSHLQCHKRTHTGEKPYECNQCGKAFSQHGLLQRHKRTHTGEKPYMNVINMVKPLHNS 98
                 **** *:***********:***:**.: .***************      :  *.: :
```

**Fig. 1 Sample of sequence Alignment text based representations**

In sequence alignment of graphical representations, sequences are written in rows so that aligned residues appear in successive columns. While in text formats, aligned columns containing identical or similar characters are indicated with a system of conservation symbols. An asterisk or pipe symbol is used to represent the similarity of these two columns, a colon for conservative substitutions and a period for semi-conservative substitutions.

Many sequence visualization techniques use a color coding scheme to display information about the properties of the individual sequence elements. In DNA and RNA sequences, each nucleotide is represented by a specific color. In protein alignments, color is used to indicate amino acid properties in determining the conservation of a given amino acid substitution.

## 2. Pair-wise Alignment

Pair-wise sequence alignment methods are used to find the best-matching pairs of two sequences. The three primary methods of pair-wise alignments are dot-matrix, dynamic programming and word methods. One way of quantifying the utility of a pair-wise alignment is the 'maximum unique match', or the longest subsequence that occurs in both query sequence.

***a) Dot-Matrix Method:*** The two sequences are written along the top row and leftmost column of a two-dimensional matrix and a dot is placed at any point where the characters in the appropriate columns match. We try to draw lines diagonally. The dot plots of very closely related sequences will appear as a single line along the matrix's main diagonal (Fig. 2). The dot-matrix approach produces a simple way of alignments for small sequences with the similar regions but time-consuming to analyze large sequences.

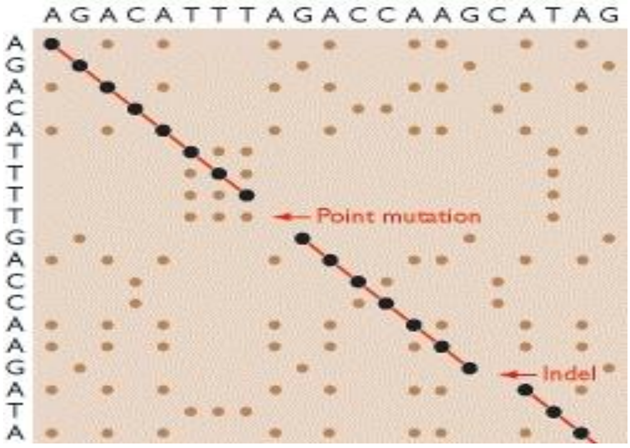


**Fig. 2: The dot matrix technique for sequence alignment**

There are many problems with dot plots such as noise, lack of clarity, difficulty extracting match summary statistics. Dot-plots are limited to two sequences only.

***b) Dynamic Programming:*** Dynamic programming can be applied to produce global and local alignments. This can be done by applying Needleman-Wunsch algorithm for global alignment and Smith-Waterman algorithm for the local alignments. In general, alignments use a substitution matrix to assign scores for matches or mismatches, and a gap penalty for matching an in one sequence with a gap in the other.

DNA and RNA alignments may use a different scoring matrix, but in practice often simply assign a positive match score, a negative mismatch score, and a negative gap penalty. Dynamic programming can be useful in aligning nucleotide to protein sequences. The framesearch method produces a series of global or local pair-wise alignments between a query nucleotide sequence and a search set of protein sequences, or vice versa. The BLAST and EMBOSS provide basic tools for creating alignments of the sequences.

***c) Word Method:*** Word or *k*-tuple methods are heuristic methods but are not guaranteed to find an optimal alignment solution. These methods are especially useful in large-scale database searches Word methods are best known for their implementation in the database search tools FASTA and BLAST family. Word methods identify a series of short, non-overlapping subsequences ("words") that are matched to candidate database sequences. The relative positions of the word in the two sequences being compared are subtracted to obtain an offset; this will indicate a region of alignment if multiple distinct words produce the same offset.

In the FASTA method, the user defines a value *k* to use as the word length with which to search the database. The method is slower but more sensitive for lower values of *k*, which are preferred for searching a very short query sequence. The BLAST family of search methods provides a number of algorithms optimized for particular types of queries. BLAST was developed to provide a faster alternative to FASTA without sacrificing accuracy. BLAST uses a word search of length *k*, but evaluates only the most significant word matches. Most BLAST implementations use a fixed default word length that is optimized for the query and database. Web based implementations are available such as EMBL FASTA and NCBI BLAST.

## 3. Global and Local Alignment

### *Global Alignment*

Global alignments, which attempt to align every residue of each sequence, when the size of the sequences are similar or of equal size. A general global alignment technique is based on dynamic programming i.e., Needleman-Wunsch algorithm. This can be easily understood with the following two sequences aligned globally as follows

G A A T T C A G T T A (sequence #1)
G G A T C G A (sequence #2)

In simple dynamic programming principle, we construct a matrix. The matrix will be filled by inserting 0 or 1 where ever there is a mismatch or match. We also penalize the gaps with 0 as a simple case. Following steps are needed for construction of the matrix

i. Initialization

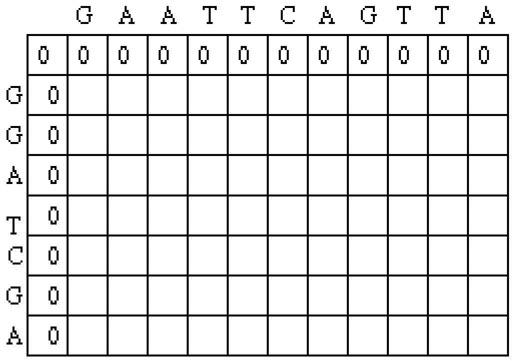
ii. Matrix fill (scoring)

iii. Traceback (alignment)

i. Initialization

The first step is to create a matrix with M + 1 columns and N + 1 rows where M and N are the sizes of the sequences to be aligned.

With the given sequences, length of sequence #1 = 11 and length of sequence #2 is 7. The size of the matrix will be 12*8 (11+1 * 7+1). The first row and first column of the matrix can be initially filled with 0 because we assume assumes there is no gap opening or gap extension penalty as shown in fig. 3.



| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | | | | | | | | | | | |
| G | 0 | | | | | | | | | | | |
| A | 0 | | | | | | | | | | | |
| T | 0 | | | | | | | | | | | |
| C | 0 | | | | | | | | | | | |
| G | 0 | | | | | | | | | | | |
| A | 0 | | | | | | | | | | | |

**Fig. 3. Initial matrix with two sequences**

ii. Matrix Fill

One possible way of filling the matrix is to find the maximum global alignment score by starting from the upper left hand corner of the matrix and find the maximal score $M_{i,j}$ for each position in the matrix.

For each position, $M_{i,j}$ is defined to be the maximum score at position i,j i.e.,

**$M_{i,j}$ = MAXIMUM[**

**$M_{i-1, j-1}$ + $S_{i,j}$** (match/mismatch in the diagonal),

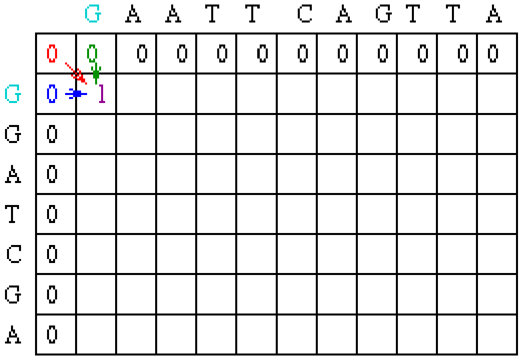
**$M_{i,j-1}$ + w** (gap in sequence #1),

**$M_{i-1,j}$ + w** (gap in sequence #2)**]**

In fig. 4, $M_{i-1,j-1}$ will be red, $M_{i,j-1}$ will be blue and $M_{i-1,j}$ will be green. The score at position 1,1 in the matrix can be calculated. Since the first residue in both sequences is a G i.e., a match, so score $S_{1,1} = 1$. We assumed the gap penalty as 0.

Thus, $M_{1,1} = \text{MAX}[M_{0,0} + 1, M_{1,0} + 0, M_{0,1} + 0] = \text{MAX}[1, 0, 0] = 1$.

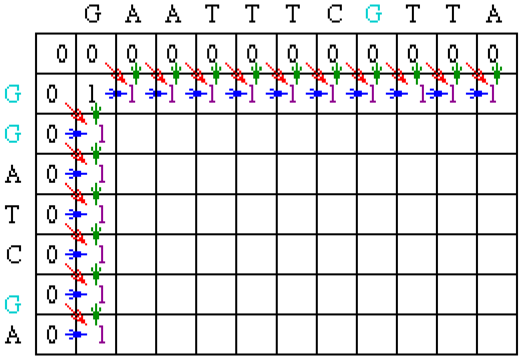
A value of 1 is then placed in position 1,1 of the scoring matrix.



| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 1 | | | | | | | | | | |
| G | 0 | | | | | | | | | | | |
| A | 0 | | | | | | | | | | | |
| T | 0 | | | | | | | | | | | |
| C | 0 | | | | | | | | | | | |
| G | 0 | | | | | | | | | | | |
| A | 0 | | | | | | | | | | | |

**Fig. 4. Sample fill of the entry $M_{1,1}$**

Now the element $M_{1,2}$, the value is the max of 0 (for a mismatch), 0 (for a vertical gap) or 1 (horizontal gap). The rest of element of first row can be filled up similarly. At this point, there is a G in both sequences (light blue). Thus, the value for the cell at row 1 column 8 is the maximum of 1 (for a match), 0 (for a vertical gap) or 1 (horizontal gap). The value will again be 1 as in fig. 5

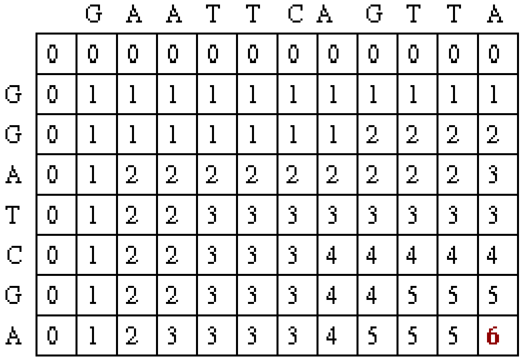


| | | G | A | A | T | T | T | C | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| G | 0 | 1 | | | | | | | | | | |
| A | 0 | 1 | | | | | | | | | | |
| T | 0 | 1 | | | | | | | | | | |
| C | 0 | 1 | | | | | | | | | | |
| G | 0 | 1 | | | | | | | | | | |
| A | 0 | 1 | | | | | | | | | | |

**Fig. 5. Sample fill of the entry whene there is a collosion of two cells for $M_{1,8}$**

Now similarly at column 2. The location at row 2 will be assigned the value of the maximum of 1(mismatch), 1(horizontal gap) or 1 (vertical gap). So its value is 1.

After filling in all of the values the score matrix is shown in fig. 6:



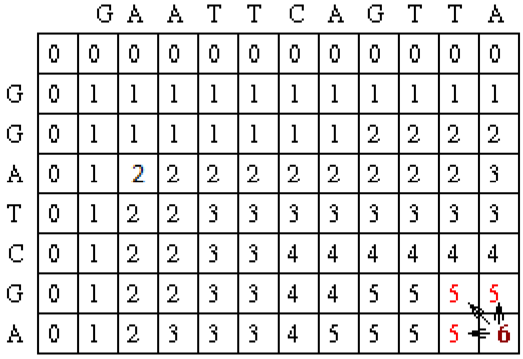
| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| A | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 |
| T | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| C | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 |
| G | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 4 | 4 | 5 | 5 | 5 |
| A | 0 | 1 | 2 | 3 | 3 | 3 | 3 | 4 | 5 | 5 | 5 | 6 |

**Fig. 6. Final filled matrix**

### iii. Traceback Step

After the matrix fill step, find the the maximum alignment score for the two test sequences. The traceback step determines the actual alignment(s) that result in the maximum score. Note that with a simple scoring algorithm such as one that is used here, there are likely to be multiple maximal alignments.

The traceback step begins in the matrix that leads to the maximal score. In this case, there is a 6 in that location. Traceback takes the current cell and looks to the neighbor cells that could be direct predecessors. This means that it looks to the neighbor to the left (gap in sequence #2), the diagonal neighbor (match/mismatch), and the neighbor above it (gap in sequence #1). The algorithm for traceback chooses as the next cell in the sequence one of the possible predacessors. In this case, the neighbors are marked in red. They are all also equal to 5 as in fig 7.

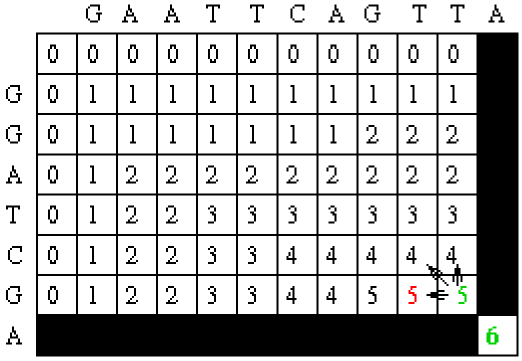


| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| A | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 |
| T | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| C | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 |
| G | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 5 | 5 |
| A | 0 | 1 | 2 | 3 | 3 | 3 | 4 | 5 | 5 | 5 | 5 | 6 |

**Fig. 7. Traceback process start where the score is maximum**

Since the current cell has a value of 6 and the scores are 1 for a match and 0 for anything else, the only possible predecessor is the diagonal match/mismatch neighbor. If more than one possible predecessor exists, any can be chosen. The corresponding row and column can be crossed out as in fig. 8. This gives us a current alignment of

(Seq #1) A
|
(Seq #2) A



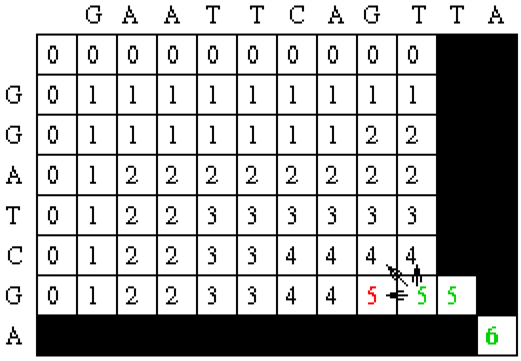
| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | |
| A | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | |
| T | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | |
| C | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | |
| G | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 5 | |
| A | | | | | | | | | | | | 6 |

**Fig. 8. Traceback steps and crossing of the row and column**

Now, look at the current cell and determine which cell is its direct predecessor. In this case, it is the cell with the red 5 as in fig. 9. The alignment as described in the above step adds a gap to sequence #2 , so the current alignment is

```
(Seq #1)    T A
              |
(Seq #2)    _ A
```

Once again, the direct predecessor produces a gap in sequence #2.



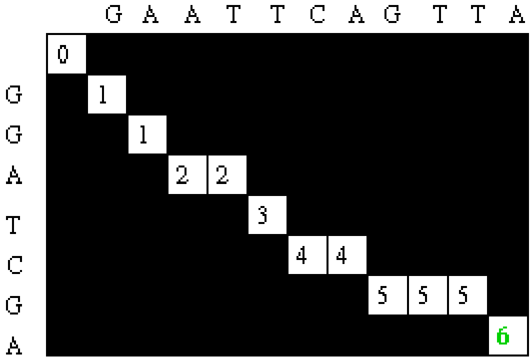
| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | | |
| A | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | | |
| T | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | | |
| C | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 4 | | |
| G | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 5 | |
| A | | | | | | | | | | | | 6 |

**Fig. 9. Traceback steps and crossing of the row and column**

After this step, the current alignment is

```
(Seq #1)    T T A
                |
            _ _ A
```

Continuing on with the traceback step, we eventually get to a position in row 0 and column 0, which tells us that traceback is completed as in fig. 10.



| | | G | A | A | T | T | C | A | G | T | T | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | | | | | | | | | | | |
| G | | 1 | | | | | | | | | | |
| G | | | 1 | | | | | | | | | |
| A | | | | 2 | 2 | | | | | | | |
| T | | | | | | 3 | | | | | | |
| C | | | | | | | 4 | 4 | | | | |
| G | | | | | | | | | 5 | 5 | 5 | |
| A | | | | | | | | | | | | 6 |

**Fig. 10. Final matrix with the traceback steps**

One possible maximum alignment is

```
G A A T T C A G T T A
|   |   | |   |     |
G G A _ T C _ G _ _ A
```

### *Local Alignment*

Local alignments are more useful for dissimilar sequences that may contains regions of similarity or similar sequence motifs within their larger sequence context. The Smith-Waterman algorithm is a general local alignment method based on dynamic programming. A local alignment searches for regions of local similarity between two sequences and need not include the entire length of the sequences. This can be done by reading a scoring matrix that contains values for every possible residue or nucleotide match or mismatch. The Smith-Waterman algorithm is a member of the class of algorithms that can calculate the best score and local alignment in the order of m*n steps, where 'm' and 'n' are the lengths of the two sequences. Local alignment methods only report the best matching areas between two sequences while there may be a large number of alternative local alignments which do not score as highly as the best alignment done by this algorithm.

Consider the two DNA sequences to be globally aligned are:

ACACACT (x=7, length of sequence 1)

AGCACAC (y=7, length of sequence 2)

It also follows three steps

i. Initialization

ii. Matrix fill (scoring)

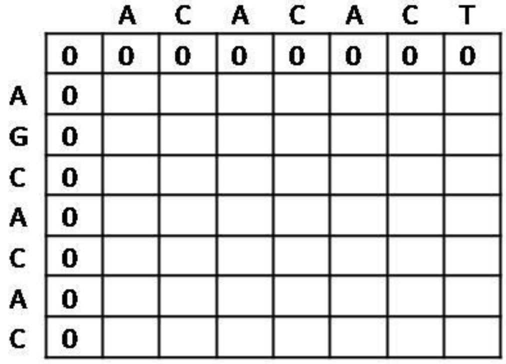
iii. Traceback (alignment)

Let us assume the simple scoring scheme as

- $S_{i,j} = 2$ if there is a match
- $S_{i,j} = -1$ if there is a mismatch
- w = -1 as gap penalty

i. Initialization

The first step in the global alignment dynamic programming approach is to create a matrix with M + 1 columns and N + 1 rows where M and N correspond to the size of the sequences to be aligned. In this example, we assume that there is no gap opening or gap extension penalty. The first row and first column of the matrix can be initially filled with 0 as in fig. 11.



| | | A | C | A | C | A | C | T |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | | | | | | | |
| G | 0 | | | | | | | |
| C | 0 | | | | | | | |
| A | 0 | | | | | | | |
| C | 0 | | | | | | | |
| A | 0 | | | | | | | |
| C | 0 | | | | | | | |

**Fig. 11. Initial matrix with first row and first column element as 0**

ii. Matrix Fill

One way to fill the matrix is to find the maximum global alignment score by starting from the upper left hand corner in the matrix and get the maximal score $M_{i,j}$ for each position in the matrix. In order to find $M_{i,j}$ for any i,j it is minimal to know the score for the matrix positions to the left, above and diagonal to i, j. In terms of matrix positions, it is necessary to know $M_{i-1,j}$, $M_{i,j-1}$ and $M_{i-1, j-1}$.

For each position, $M_{i,j}$ is defined to be the maximum score at position i,j; i.e.

**$M_{i,j}$ = MAXIMUM[**

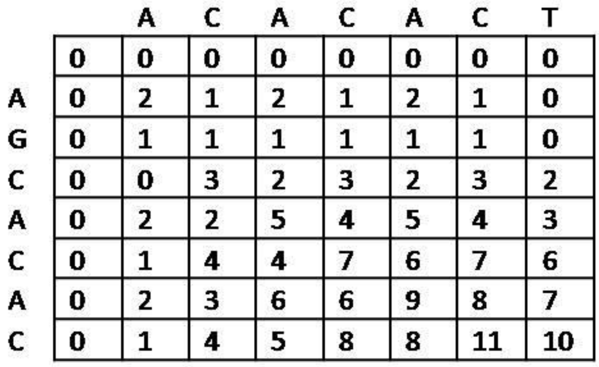
**$M_{i-1, j-1}$ + $S_{i,j}$** (match/mismatch in the diagonal),

**$M_{i,j-1}$ + w** (gap in sequence #1),

**$M_{i-1,j}$ + w** (gap in sequence #2)**]**

Using this information, the score at position 1,1 in the matrix can be calculated. Since the first residue in both sequences is A, $S_{1,1} = 2$, and by the assumptions stated at the beginning, w = 0. Thus, $M_{1,1} = MAX[M_{0,0} + 2, M_{1,0} -1, M_{0,1} -1] = MAX [2, -1, -1] = 2$.

A value of 2 is then placed in position 1,1 of the scoring matrix as in fig. 12. And subsequently the whole matrix is filled in the same way.



| | | A | C | A | C | A | C | T |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 2 | 1 | 2 | 1 | 2 | 1 | 0 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| C | 0 | 0 | 3 | 2 | 3 | 2 | 3 | 2 |
| A | 0 | 2 | 2 | 5 | 4 | 5 | 4 | 3 |
| C | 0 | 1 | 4 | 4 | 7 | 6 | 7 | 6 |
| A | 0 | 2 | 3 | 6 | 6 | 9 | 8 | 7 |
| C | 0 | 1 | 4 | 5 | 8 | 8 | 11 | 10 |

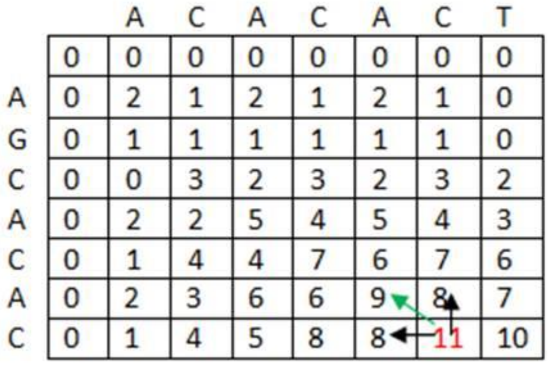
**Fig. 12. Final filled matrix**

iii. Traceback

After the matrix fill step, the maximum alignment score for these two test sequences is 11. The traceback step determines the actual alignment(s) for the maximum score. It is not mandatory that the last cell has the maximum alignment score.

The traceback step begins with the position that leads to the maximal score. In this case, there is 11 in that location.

Trace back takes the current cell and looks to the neighbor cells that could be direct predecessors. This means it looks to the neighbor to the left (gap in sequence #2), the diagonal neighbor (match/mismatch), and the neighbor above it (gap in sequence #1) as in fig. 13. The algorithm for trace back chooses as the next cell in the sequence one of the possible predecessors. This continues till cell with value 0 is reached.



| | | A | C | A | C | A | C | T |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 2 | 1 | 2 | 1 | 2 | 1 | 0 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| C | 0 | 0 | 3 | 2 | 3 | 2 | 3 | 2 |
| A | 0 | 2 | 2 | 5 | 4 | 5 | 4 | 3 |
| C | 0 | 1 | 4 | 4 | 7 | 6 | 7 | 6 |
| A | 0 | 2 | 3 | 6 | 6 | 9 | 8 | 7 |
| C | 0 | 1 | 4 | 5 | 8 | 8 | 11 | 10 |

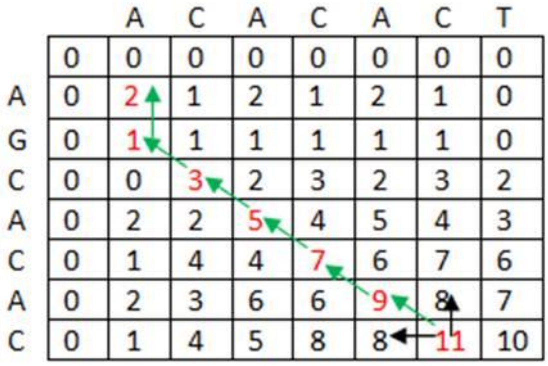
**Fig. 13. Traceback Step**

The only possible predecessor is the diagonal match/mismatch neighbor. If more than one possible predecessor exists, any can be chosen. This gives us a current alignment of

(Seq #1)    C
            |
(Seq #2)    C

So now we look at the current cell and determine which cell is its direct predecessor. In this case, it is the cell with the red 9 as in fig. 14.

(Seq #1)    C A
            |  |
(Seq #2)    C A



| | | A | C | A | C | A | C | T |
|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 2 | 1 | 2 | 1 | 2 | 1 | 0 |
| G | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| C | 0 | 0 | 3 | 2 | 3 | 2 | 3 | 2 |
| A | 0 | 2 | 2 | 5 | 4 | 5 | 4 | 3 |
| C | 0 | 1 | 4 | 4 | 7 | 6 | 7 | 6 |
| A | 0 | 2 | 3 | 6 | 6 | 9 | 8 | 7 |
| C | 0 | 1 | 4 | 5 | 8 | 8 | 11 | 10 |

**Fig. 14. Traceback step with the correct arrows**

Continuing with the traceback step, we eventually get a position in column 0 or row 0 which tells us that traceback is completed as in fig. 15.

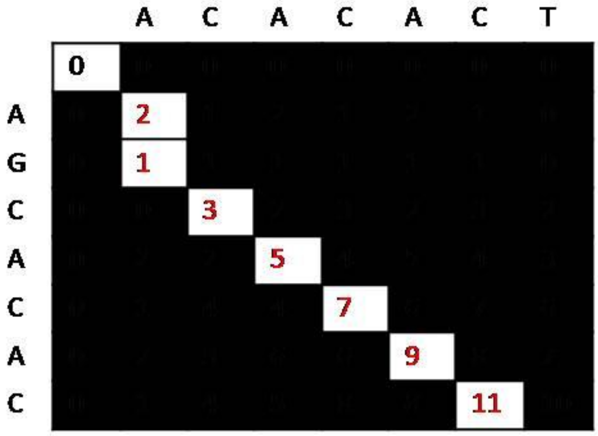


**Fig. 15. Final Traceback Matrix**

The possible maximum alignment is:

AG C A C A C

|   | | | | |

A _ C A C A C

There is a combination of these two methods which is called hybrid methods, also known as semiglobal or "glocal" methods. This method attempts to find the best possible alignment that includes the start and end of one or the other sequence. This can be especially useful when the downstream part of one sequence overlaps with the upstream part of the other sequence. In this case, neither global nor local alignment is entirely appropriate.

## 4. Significance of Sequence Alignment

Sequence alignments are useful in bioinformatics for identifying sequence similarity, producing phylogenetic trees, and developing homology models of protein structures. However, the biological relevance of sequence alignments is not always clear. Alignments are often assumed to reflect a degree of evolutionary change between sequences descended from a common ancestor; however, it is formally possible that convergent evolution can occur to produce apparent similarity between proteins that are evolutionarily unrelated but perform similar functions and have similar structures.

In database searches such as BLAST, statistical methods can determine the likelihood of a particular alignment between sequences or sequence regions arising by chance with the given the size and composition of the database being searched. These values can vary significantly depending on the search space. In particular, the likelihood of finding a given alignment by chance increases, if the database consists only of sequences from the same organism as the query sequence. Repetitive sequences in the database or query can also distort both the search results and the assessment of statistical significance. BLAST automatically filters such repetitive sequences in the query to avoid apparent hits that are statistical artifacts.

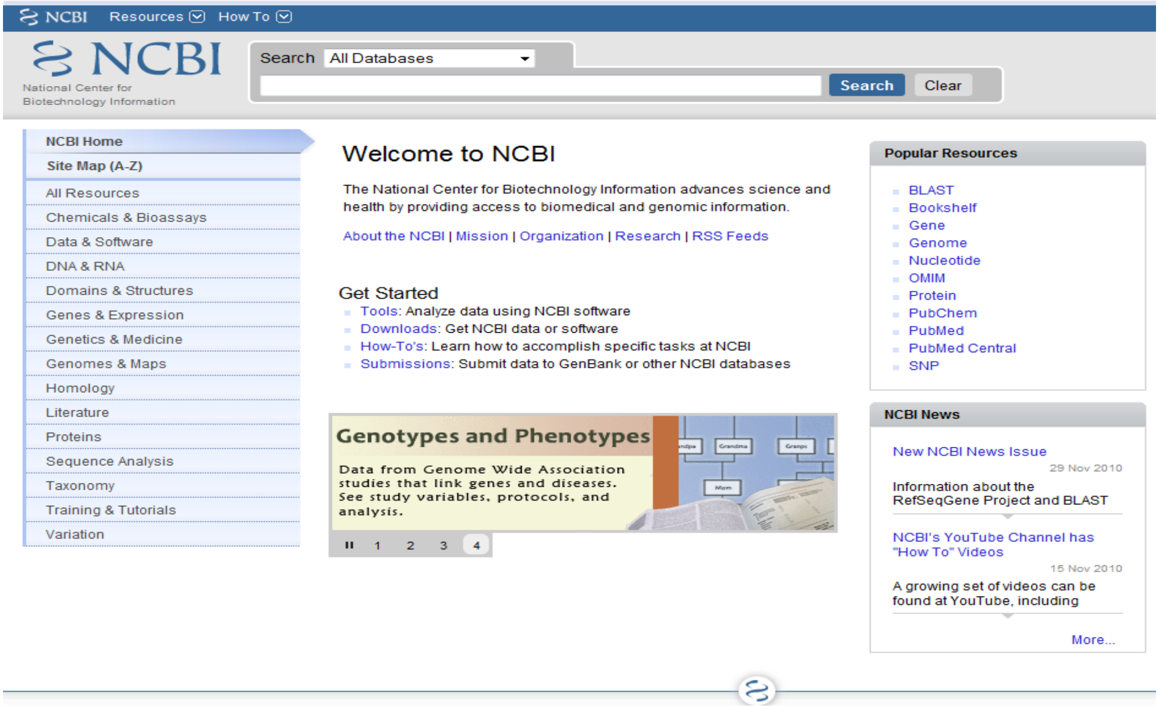
The choice of a **scoring function** that reflects biological or statistical observations about known sequences is important to producing good alignments. Protein sequences are frequently aligned using substitution matrices that reflect the probabilities of given character-to-character substitutions. A series of matrices called PAM matrices (Point

Accepted Mutation matrices, originally defined by Margaret Dayhoff and sometimes referred to as "Dayhoff matrices") explicitly encode evolutionary approximations regarding the rates and probabilities of particular amino acid mutations. Another common series of scoring matrices, known as BLOSUM (Blocks Substitution Matrix), encodes empirically derived substitution probabilities. Variants of both types of matrices are used to detect sequences with differing levels of divergence, thus allowing users of BLAST or FASTA to restrict searches to more closely related matches or expand to detect more divergent sequences. Gap penalties account for the introduction of a gap - on the evolutionary model, an insertion or deletion mutation - in both nucleotide and protein sequences, and therefore the penalty values should be proportional to the expected rate of such mutations. The quality of the alignments produced therefore depends on the quality of the scoring function.

## 5. Sequence Databases

The repositories for the genomic sequences are

**National Center for Biotechnology Information** (**NCBI**) is part of the United States National Library of Medicine (NLM), a branch of the National Institutes of Health. The NCBI is located in Bethesda, Maryland and was founded in 1988 through legislation sponsored by Senator Claude Pepper. The NCBI houses genome sequencing data in GenBank and an index of biomedical research articles in PubMed Central and PubMed, as well as other information relevant to biotechnology. All these databases are available online through the Entrez search engine. The NCBI is directed by David Lipman, one of the original authors of the BLAST sequence alignment program and a widely respected figure in Bioinformatics. The NCBI has had responsibility for making available the GenBank DNA sequence database since 1992 as shown in fig. 16. GenBank coordinates with individual laboratories and other sequence databases such as those of the European Molecular Biology Laboratory (EMBL) and the DNA Data Bank of Japan (DDBJ). Since 1992, NCBI has grown to provide other databases in addition to GenBank. NCBI provides Online Mendelian Inheritance in Man, the Molecular Modeling Database (3D protein structures), dbSNP a database of single-nucleotide polymorphisms, the Unique Human Gene Sequence Collection, a Gene Map of the human genome, a Taxonomy Browser, and coordinates with the National Cancer Institute to provide the Cancer Genome Anatomy Project.



NCBI Resources How To

NCBI
National Center for Biotechnology Information

Search All Databases

Search Clear

- NCBI Home
- Site Map (A-Z)
- All Resources
- Chemicals & Bioassays
- Data & Software
- DNA & RNA
- Domains & Structures
- Genes & Expression
- Genetics & Medicine
- Genomes & Maps
- Homology
- Literature
- Proteins
- Sequence Analysis
- Taxonomy
- Training & Tutorials
- Variation

Welcome to NCBI

The National Center for Biotechnology Information advances science and health by providing access to biomedical and genomic information.

About the NCBI | Mission | Organization | Research | RSS Feeds

Get Started

- Tools: Analyze data using NCBI software
- Downloads: Get NCBI data or software
- How-To's: Learn how to accomplish specific tasks at NCBI
- Submissions: Submit data to GenBank or other NCBI databases

Genotypes and Phenotypes

Data from Genome Wide Association studies that link genes and diseases. See study variables, protocols, and analysis.

1 2 3 4

Popular Resources

- BLAST
- Bookshelf
- Gene
- Genome
- Nucleotide
- OMIM
- Protein
- PubChem
- PubMed
- PubMed Central
- SNP

NCBI News

New NCBI News Issue
29 Nov 2010
Information about the RefSeqGene Project and BLAST

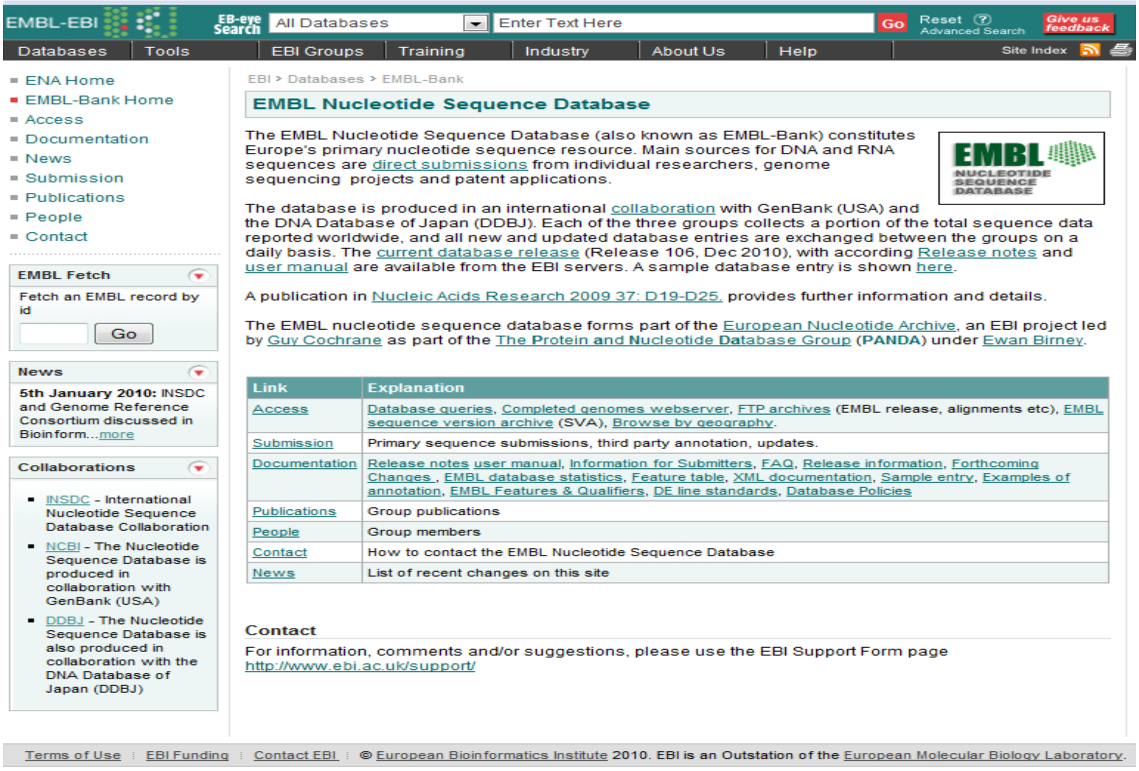
NCBI's YouTube Channel has "How To" Videos
15 Nov 2010
A growing set of videos can be found at YouTube, including

More...

**Fig. 16. NCBI portal**

The NCBI assigns a unique identifier (Taxonomy ID number) to each species of organism. The NCBI has software tools that are available by WWW browsing or by FTP. For example, BLAST is a sequence similarity searching program. BLAST can do sequence comparisons against the GenBank DNA database in less than 15 seconds. The **NCBI Bookshelf** is a collection of freely available, downloadable, on-line versions of selected biomedical books. The Bookshelf has various titles covering aspects of molecular biology, biochemistry, cell biology, genetics, microbiology, a couple of disease states from a molecular and cellular point of view, research methods, and virology. Some of the books are online versions of previously published books, while others, such as Coffee Break (book), are written and edited by NCBI staff. The Bookshelf is a complement to the Entrez PubMed repository of peer-reviewed publication abstracts in that Bookshelf contents provide established perspectives on evolving areas of study and a context in which many disparate individual pieces of reported research can be organized.

**European Molecular Biology Laboratory** (**EMBL**) is a molecular biology research institution supported by 20 European countries and Australia as associate member state. The EMBL was created in 1974 and is a non-profit organisation funded by public research money from its member states. Research at EMBL is conducted by approximately 85 independent groups covering the spectrum of molecular biology. The Laboratory operates from five sites: the main Laboratory in Heidelberg, and Outstations in Hinxton (the European Bioinformatics Institute (**EBI**)), Grenoble, Hamburg, and Monterotondo near Rome as in fig. 17. Each of the sites has a research specific field. At EBI, the research is oriented towards computational biology and bioinformatics. At Grenoble and Hamburg the research is in the field of structural biology. At Monterotondo the research is focused mainly on mouse models for clinical research. At the headquarters in Heidelberg, there are big departments in Cell Biology and Gene Expression as well as smaller complementing the aforementioned research fields.



EBI > Databases > EMBL-Bank

**EMBL Nucleotide Sequence Database**

The EMBL Nucleotide Sequence Database (also known as EMBL-Bank) constitutes Europe's primary nucleotide sequence resource. Main sources for DNA and RNA sequences are direct submissions from individual researchers, genome sequencing projects and patent applications.

The database is produced in an international collaboration with GenBank (USA) and the DNA Database of Japan (DDBJ). Each of the three groups collects a portion of the total sequence data reported worldwide, and all new and updated database entries are exchanged between the groups on a daily basis. The current database release (Release 106, Dec 2010), with according Release notes and user manual are available from the EBI servers. A sample database entry is shown here.

A publication in Nucleic Acids Research 2009 37: D19-D25. provides further information and details.

The EMBL nucleotide sequence database forms part of the European Nucleotide Archive, an EBI project led by Guy Cochrane as part of the The Protein and Nucleotide Database Group (PANDA) under Ewan Birney.

| Link | Explanation |
|---|---|
| Access | Database queries, Completed genomes webserver, FTP archives (EMBL release, alignments etc), EMBL sequence version archive (SVA), Browse by geography. |
| Submission | Primary sequence submissions, third party annotation, updates. |
| Documentation | Release notes user manual, Information for Submitters, FAQ, Release information, Forthcoming Changes , EMBL database statistics, Feature table, XML documentation, Sample entry, Examples of annotation, EMBL Features & Qualifiers, DE line standards, Database Policies |
| Publications | Group publications |
| People | Group members |
| Contact | How to contact the EMBL Nucleotide Sequence Database |
| News | List of recent changes on this site |

**Contact**

For information, comments and/or suggestions, please use the EBI Support Form page http://www.ebi.ac.uk/support/

**News**

**5th January 2010:** INSDC and Genome Reference Consortium discussed in Bioinform...more

**Collaborations**

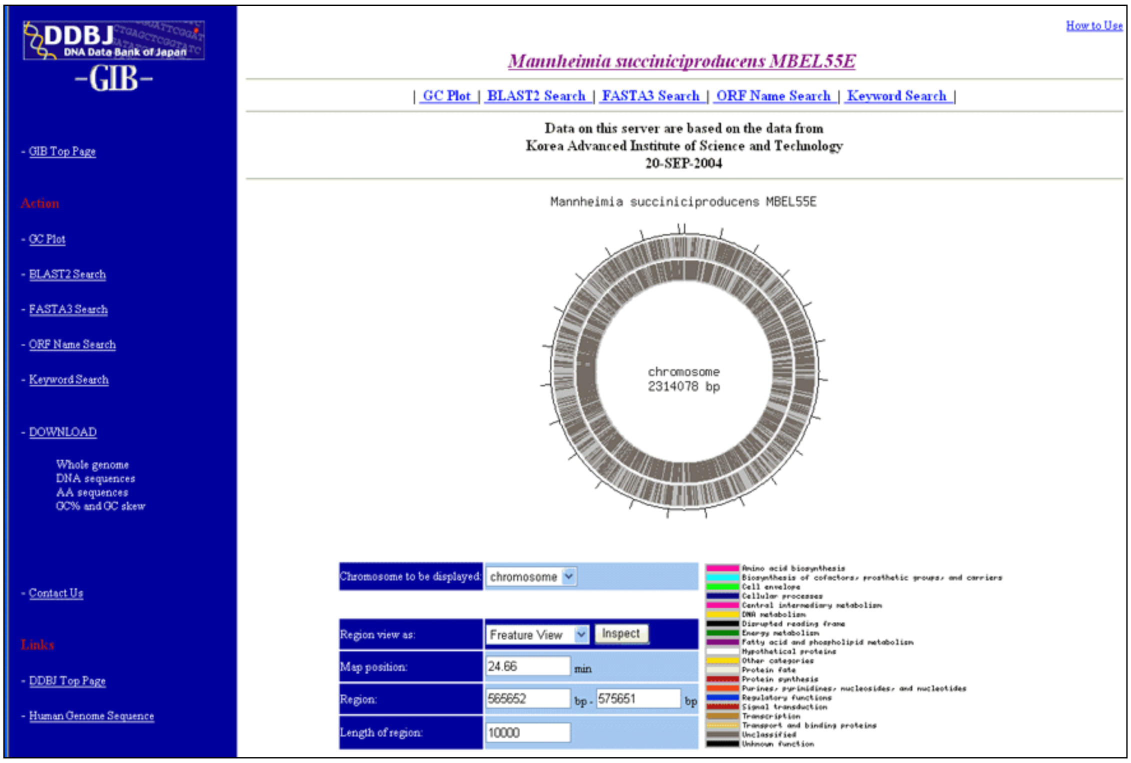
- INSDC - International Nucleotide Sequence Database Collaboration
- NCBI - The Nucleotide Sequence Database is produced in collaboration with GenBank (USA)
- DDBJ - The Nucleotide Sequence Database is also produced in collaboration with the DNA Database of Japan (DDBJ)

Terms of Use | EBI Funding | Contact EBI | © European Bioinformatics Institute 2010. EBI is an Outstation of the European Molecular Biology Laboratory.

**Fig. 17. EMBL portal**

The cornerstones of EMBL's mission are: to perform basic research in molecular biology and molecular medicine, to train scientists, students and visitors at all levels, to offer vital services to scientists in the member states, to develop new instruments and methods in the life sciences, and to actively engage in technology transfer. EMBL's international PhD Programme has a student body of about 170. The Laboratory also sponsors an active Science and Society programme. Many scientific breakthroughs have been made at EMBL, most notably the first systematic genetic analysis of embryonic development in the fruit fly by Christiane Nüsslein-Volhard and Eric Wieschaus, for which they were awarded the Nobel Prize for Medicine in 1995.

**DNA Data Bank of Japan (DDBJ)** is a DNA data bank. It is located at the National Institute of Genetics (NIG) in the Shizuoka prefecture of Japan. It is also a member of the International Nucleotide Sequence Database Collaboration or INSDC. It exchanges its data with European Molecular Biology Laboratory at the European Bioinformatics Institute and with GenBank at the National Center for Biotechnology Information on a daily basis. Thus these three databanks contents the same data at any given time. DDBJ began data bank activities since 1986 at NIG and it boasts to be the only nucleotide sequence data bank in Asia. Although DDBJ mainly receives its data from Japanese researchers, however it can accept data from a contributor belonging to any other country as in fig. 18.



- GIB Top Page

Action

- GC Plot
- BLAST2 Search
- FASTA3 Search
- ORF Name Search
- Keyword Search
- DOWNLOAD

Whole genome
DNA sequences
AA sequences
GC% and GC skew

- Contact Us

Links

- DDBJ Top Page
- Human Genome Sequence

How to Use

*Mannheimia succiniciproducens MBEL55E*

| GC Plot | BLAST2 Search | FASTA3 Search | ORF Name Search | Keyword Search |

Data on this server are based on the data from
Korea Advanced Institute of Science and Technology
20-SEP-2004

Mannheimia succiniciproducens MBEL55E

Chromosome to be displayed: chromosome

Region view as: Feature View Inspect

Map position: 24.66 min

Region: 565652 bp - 575651 bp

Length of region: 10000

Amino acid biosynthesis
Biosynthesis of cofactors, prosthetic groups, and carriers
Cell envelope
Cellular processes
Central intermediary metabolism
DNA metabolism
Disrupted reading frame
Energy metabolism
Fatty acid and phospholipid metabolism
Hypothetical proteins
Other categories
Protein fate
Protein synthesis
Purines, pyrimidines, nucleosides, and nucleotides
Regulatory functions
Signal transduction
Transcription
Transport and binding proteins
Unclassified
Unknown function

**Fig. 18. DDBJ Portal**

DDBJ is primarily funded by the Japanese Ministry of Education, Culture, Sports, Science and Technology (MEXT). DDBJ has an international advisory committee which consists of nine members, 3 members each from Europe, US, and Japan. This committee advice DDBJ about its maintenance, management and future plans once a year. Apart from this DDBJ also has an international collaborative committee which advises on various technical issues related to international collaboration and consists of working-level participants.

## 6. Softwares Used in Sequence Alignment

| S. No. | Name | Function | Website Link |
|---|---|---|---|
| 1 | ALIGN | Sequence Analysis | http://www.ebi.ac.uk/Tools/emboss/align |
| 2 | CENSOR | Sequence Analysis | http://www.ebi.ac.uk/Tools/censor/ |
| 3 | CLUSTALW2 | Sequence Analysis | http://www.ebi.ac.uk/Tools/clustalw2/ |
| 4 | CpG Plot/ CpGreport | Sequence Analysis | http://www.ebi.ac.uk/Tools/emboss/cpgplot/ |
| 5 | Genewise | Sequence Analysis | http://www.ebi.ac.uk/Tools/Wise2/ |
| 6 | Kalign | Sequence Analysis | http://www.ebi.ac.uk/Tools/kalign |
| 7 | MAFFT | Sequence Analysis | http://www.ebi.ac.uk/Tools/mafft/ |
| 8 | MUSCLE | Sequence Analysis | http://www.ebi.ac.uk/Tools/muscle/ |

| | | | |
|---|---|---|---|
| 9 | Pepstats/ Pepwindow/Pepinfo | Sequence Analysis | http://www.ebi.ac.uk/Tools/emboss/ pepinfo/ |
| 10 | PromoterWise | Sequence Analysis | http://www.ebi.ac.uk/Tools/Wise2/ promoterwise.html |
| 11 | SAPS | Sequence Analysis | http://www.ebi.ac.uk/Tools/saps/ |
| 12 | T-coffee | Sequence Analysis | http://www.ebi.ac.uk/Tools/t-coffee/ |
| 13 | Transeq | Sequence Analysis | http://www.ebi.ac.uk/Tools/emboss/transeq/ |
| 14 | COBALT | Sequence Analysis | http://www.ncbi.nlm.nih.gov/tools/ cobalt/ |
| 15 | Genome Workbench | Sequence Analysis | http://www.ncbi.nlm.nih.gov/projects/ gbench/ |
| 16 | ORF Finder | Sequence Analysis | http://www.ncbi.nlm.nih.gov/gorf/gorf/ html |
| 17 | Primer - BLAST | Sequence Analysis | http://www.ncbi.nlm.nih.gov/tools/ primer-blast |
| 18 | ProSplign | Sequence Analysis | http://www.ncbi.nlm.nih.gov/sutils/static/prosplin/prosplign.html |
| 19 | Splign | Sequence Analysis | http://www.ncbi.nlm.nih.gov/sutils/ splign/ |
| 20 | VecScreen | Sequence Analysis | http://www.ncbi.nlm.nih.gov/VecScreen/VecScreen.html |
| 21 | Sequence Analysis | Sequence analysis | http://www.informagen.com/SA/ |
| 22 | SeWeR | Sequence analysis | http://www.bioinformatics.org/sewer/ |
| 23 | Motif Search | Sequence analysis | http://nbc11.biologie.uni-kl.de/framed/left/menu/auto/right/ motifsearch2/ index.pl |
| 24 | DNA Translator | Sequence analysis | http://nbc11.biologie.uni-kl.de/framed/left/menu/auto/right/JDT/ |
| 25 | Non coding RNA Gene Finder | Sequence analysis | http://nbc11.biologie.uni-kl.de/framed/left/menu/auto/right/ ncRnaGeneFinder/index.pl |
| 26 | TransTerm | Sequence analysis | http://nbc11.biologie.uni-kl.de/framed/left/menu/auto/right/ transterm/ |

|  |  |  |  |
|---|---|---|---|
| 27 | QRNA | Sequence analysis | http://nbc11.biologie.unikl.de/framed/left/menu/auto/right/qrna/ |
| 28 | Clustalformatter 5 | Sequence analysis | http://nbc11.biologie.uni-kl.de/framed/left/menu/auto/right/ClustalFormatter/ |
| 29 | BioEdit | Sequence Alignment Editor | http://www.mbio.ncsu.edu/BioEdit/bioedit.html |
| 30 | FASTA | Sequence Similarity Search | http://www.ebi.ac.uk/Tools/fasta/ |
| 31 | HMMER | Homology of protein | http://hmmer.janelia.org/ |
| 32 | JAligner | Pairwise seq. alignment | http://jaligner.sourceforge.net/ |
| 33 | JSTRING | Java Search for Tandem Repeats IN Genomes | http://bioinf.dms.med.uniroma1.it/JSTRING/ |
| 34 | NCBI BLAST | Aligning Sequences | http://blast.ncbi.nlm.nih.gov/Blast.cgi |
| 35 | Gene Runner/ Motif Runner | Motif based sequence analysis | http://www.generunner.net/ |
| 36 | GoCore | Protein Seq. Alignment & Analysis | http://www.helsinki.fi/project/ritvos/GoCore/ |
| 37 | MAFFT | Multiple alignment | http://mafft.cbrc.jp/alignment/server/index.html |
| 38 | MAUVE | Multiple alignment | http://gel.ahabs.wisc.edu/mauve/ |
| 39 | MEME Suite | Motif based sequence analysis | http://meme.nbcr.net/ |
| 40 | CORAL (CDTree) | Aligning Core Conserved Regions | http://www.ncbi.nlm.nih.gov/Structure/cdtree/cdtree.shtml |
| 41 | BlastAlign | Align N Seq. with large INDELs | http://www.bioafrica.net/blast/BlastAlign.html |
| 42 | ARB software | Sequence DB Handling and Data Analysis | http://www.arb-home.de/ |

| | | | |
|---|---|---|---|
| 43 | Automated Codon Usage Analysis Software - ACUA | Nucleotide Analysis | http://www.bioinsilico.com/acua |
| 44 | AnnHyb | Nucleotide Analysis | http://www.bioinformatics.org/annhyb/ |
| 45 | SOAP2 | Short read Alignment | http://soap.genomics.org.cn/ |
| 46 | ACT (Artemis Comparison Tool) | DNA Sequence Comparison | http://www.sanger.ac.uk/resources/software/act/ |
| 47 | WU-BLAST | Multiple Sequence Alignment | www.ebi.ac.uk/Tools/blast2/ |
| 48 | CLUSTALW2 | multiple sequence alignment | http://www.ebi.ac.uk/Tools/clustalw2/ |

## References

www.wikipedia.org/

cnx.org/content/m11026/latest/

www.ncbi.nlm.nih.gov/

www.ebi.ac.uk/embl/

www.ddbj.nig.ac.jp/

# Phylogenetic Analysis

**Sarika[1], M. A. Iquebal[1], Anil Rai[2] and Dinesh Kumar[1]**

**[1]ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

**[2]Indian Council of Agricultural Research, New Delhi**

## INTRODUCTION

Phylogenetics is the study of evolutionary relationships. Biological sequences (amino acids and nucleotides) are the product of evolutionary history and phylogenies are graphical summaries of this history. Phylogenetic analysis of a family of related nucleic acids and protein sequences is the determination of how the family might have been derived during evolution. Phylogenetic analysis is the means of inferring or estimating the relationships. The evolutionary history from phylogenetic analysis is generally depicted as branching or treelike diagrams. Traditionally morphological features were used to derive relationships but now a days molecular information is used to derive relationships, which are more informative than the traditional anatomic or morphological characters. Molecular phylogeny provides new, powerful and independent tests of the theory of evolution. Evolution supported molecular phylogeny to be consistent with classical phylogeny. It also predicted that all parts of the genome should evolve in parallel and exhibit the same taxonomic pattern. The recent development of techniques to analyze and sequence proteins and nucleic acids has allowed biologists to determine relatedness of organisms and to construct phylogenetic sequences. Molecular phylogenetics attempts to determine the rates and patterns of change occurring in DNA and proteins and to construct the evolutionary history of genes and organisms.

## WHY DO WE BUILD PHYLOGENETIC TREES

The main aim of phylogenetics is to discover rates of evolutionary change, find origin of diseases, prediction of sequence function and population history. In addition to analyzing changes that have occurred in the evolution of different organisms, the evolution of a family of sequences may be studied. On the basis of analysis, sequences that are most closely related can be identified by their occupying neighboring branches on a tree. When a gene family is found in an organism and group of organisms, phylogenetic relationships among the genes can help to predict which ones might have an equivalent function. These functional predictions can be tested by genetic experiments.Phylogenetic analysis can be used to study the changes occurring in the rapidly changing species like virus. Analysis of types of changes within a population can reveal whether or not a particular gene is under selection.

## TERMINOLOGIES

A phylogeny or evolutionary tree, represents evolutionary relationships among a set of organisms or groups of organisms, called taxa (Fig. 1). Understanding phylogeny is like reading a family tree. The root of tree represents the ancestral lineage and the tips of branches represent the descendants of that ancestor. Moving from root to tip means moving forward in time. When a lineage splits (speciation), it represents a branching on a phylogeny.Whenever speciation occurs, a single ancestral lineage give rise to two or more daughter lineages.Two descendants that split from the same node are called sister groups. Branches connect nodes uniquely and define the relationship between the taxonomic units in terms of descent and ancestry. Only one branch can connect any two adjacent nodes. The branching pattern of the tree is called topology, and the branch length usually represents the number of changes that have occurred in the branch. Branches on phylogenetic trees may be scaled representing the amount of evolutionary change, time or both, under the assumption of molecular clock or they may be unscaled with no correspondence with either time or amount of evolutionary change. Phylogenies trace patterns of shared ancestry between lineages. Each lineage has a part of its history that is unique to it alone and parts that are shared with other lineages. Similarly, each lineage has ancestors that are unique to that lineage and ancestors that are with other lineages-common ancestors (Fig. 2).Clade includes a common ancestor and all the descendants of that ancestor. When clades are nested within one another, they form a nested hierarchy.

Phylogenetic trees may be rooted or un-rooted (Fig. 3). In rooted trees, a particular node is called the root, representing a common ancestor from which a unique path leads to any other node. In case of un-rooted trees, branching relationship between taxa are specified by the way they are connected to each other but the position of common ancestor is not. For example, on an unrooted tree with five species, there are five branches on which tree can be rooted. Rooting on each of the five branches has different implications for evolutionary relationships.

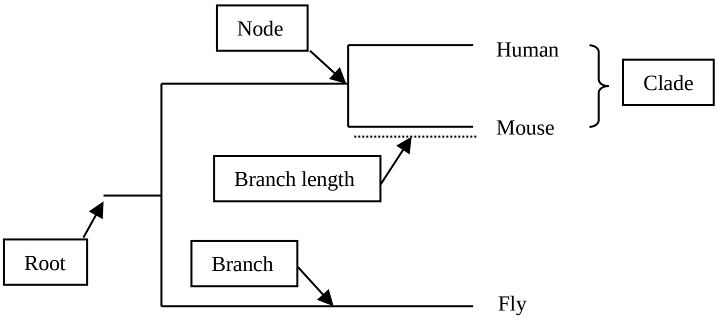


**Fig. 1: Parts of a phylogenetic tree**

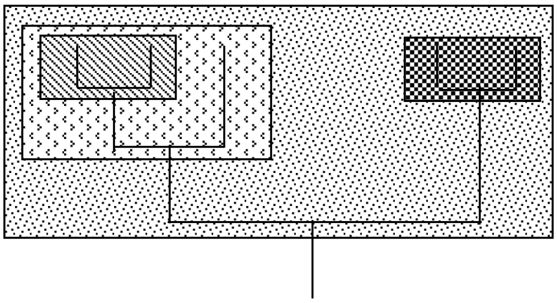
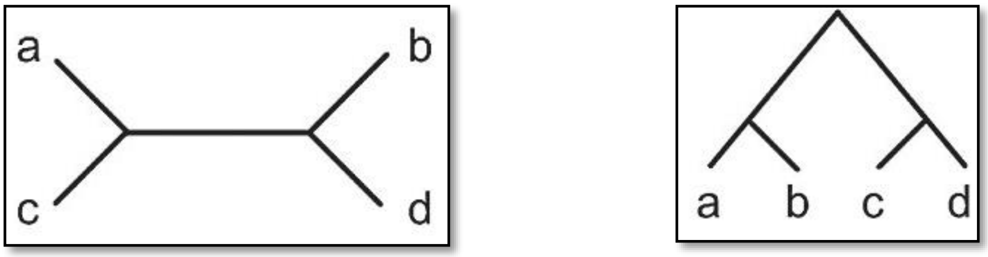


**Fig. 2: Each box represents a clade**



(A) (B)

**Fig. 3. Rooted and rooted phylogenetic tress**

## ADVANTAGES OF PHYLOGENETIC CLASSIFICATION

Phylogenetic classification has two main advantages over the Linnaean system. First, phylogenetic classification tells you something important about the organism: its evolutionary history. Second, phylogenetic classification does not attempt to "rank" organisms. Linnaean classification "ranks" groups of organisms artificially into kingdoms, phyla, orders, etc. This can be misleading as it seems to suggest that different groupings with the same rank are equivalent.

There is just no reason to think that any two identically ranked groups are comparable and by suggesting that they are, the Linnaean system is misleading. So it seems that there are many good reasons to switch to phylogenetic classification. However, organisms have been named using the Linnaean system for many hundreds of years. How are biologists making the transition to phylogenetic classification?

## CONSTRUCTION OF PHYLOGENETIC TREE

Molecular phylogenetic tree construction can be divided into four steps (Felsenstein, 2004):

A. Choosing sequences
B. Multiple sequence alignment
C. Determining a tree building method and
D. Assessing tree reliability

## A. CHOICE OF SEQUENCE

For constructing molecular phylogenetic trees, one can use either nucleotide or protein sequence data. The choice of molecular markers is an important matter because it can make a major difference in obtaining a correct tree. The decision to use nucleotide or protein sequences depends on the properties of the sequences and the purpose of study. For studying very closely related organisms nucleotide sequences can be used. For studying the evolution of more widely divergent groups of organisms, one may choose either slowly evolving nucleotide sequences, such as ribosomal RNA or protein sequences. If the phylogenetic relationships to be delineated are at the deepest level, such as between bacteria and eukaryotes, using conserved protein sequences makes more sense than using nucleotide sequences. DNA sequences are sometimes more biased than protein sequences because of preferential codon usage in different organisms. In this case, different codons for the same amino acid are used at different frequencies, leading to sequence variations not attributable to evolution. In addition, the genetic code of mitochondria varies from the standard genetic code. Therefore, for comparison of mitochondria protein-coding genes, it is necessary to translate the DNA sequences into protein sequences. Protein sequences allow more sensitive alignment than DNA sequences because the former has twenty characters versus four in the latter. For moderately divergent sequences, it is almost impossible to use DNA sequences to obtain correct alignment. In addition, to align protein-coding DNA sequences, when gaps are introduced to maximize alignment scores, they almost always cause frame-shift errors, making the alignment biologically meaningless. Synonymous substitutions are nucleotide changes in the coding sequence that do not result in amino acid sequence changes for the encoded protein. Non synonymous substitutions are nucleotide changes that result in alterations in the amino acid sequences. Comparing the two types of substitution rates helps to understand an evolutionary process of a sequence. For example, if the non-synonymous substitution rate is found to be significantly greater than the synonymous substitution rate, this means that certain parts of the protein are undergoing active mutations that may contribute to the evolution of new functions. This is described as positive selection or adaptive evolution. On the other hand, if the synonymous substitution rate is greater than the non-synonymous substitution rate, this causes only neutral changes at the amino acid level, suggesting that the protein sequence is critical enough that changes at the amino acid sequence level are not tolerated. In this case, the sequence is said to be under negative or purifying selection.

## B. MULTIPLE SEQUENCE ALIGNMENT

The second step in making phylogenetic tree is sequence alignment. This is the most critical step in the procedure because it establishes positional correspondence in evolution. Only the correct alignment produces correct phylogenetic inference because aligned positions are assumed to be genealogically related. Incorrect alignment leads to systematic errors in the final tree or even a completely wrong tree. Therefore it is essential that the sequences are correctly aligned. Two approaches are used for aligning sequence: Global alignment (similarity across the full stretch of sequences) and a Local alignment (similarity in parts of the sequences).

Although many programs exist that can generate a multiple alignment from unaligned sequences, extreme care must be taken when interpreting the results. An alignment may show perfect matching of a known active-site residue with an identical residue in a well characterized protein family, but, if the alignment is incorrect, any inference about function will also be incorrect. A clustal program such as ClustalX which aligns sequences according to an explicitly phylogenetic criterion, is the most commonly used program for the multiple alignment of biochemical sequences. The multiple alignment is inefficient with sequences if INDELs are common and substitution rates are high, most studies restrict comparisons to regions in which alignments are relatively obvious. The substitution model should be given the same emphasis as alignment and tree building. The simplest nucleotide substitution model is the Jukes–Cantor model, which assumes that all nucleotides are substituted with equal probability. A formula for deriving evolutionary distances that include hidden changes is introduced by using a logarithmic function.

$$d_{AB} = -(3/4)ln[1-(4/3)p_{AB}]$$

where $d_{AB}$ is the evolutionary distance between sequences A & B and $p_{AB}$ is the observed sequence distance measured by the proportion of substitutions over the entire length of the alignment. Another model is the Kimura two-parameter model. This is a more sophisticated model in which mutation rates for transitions and transversion are assumed to be different, which is more realistic. According to this model, transitions occur more frequently than transversions, which, therefore, provides a more realistic estimate of evolutionary distances.

The Kimura model uses the following formula:

$$d_{AB} = -(1/2)\ ln(1 - p_{ti} - p_{tv}) - (1/4)ln(1 - 2p_{tv})$$

where $d_{AB}$ is the evolutionary distance between sequences A and B, $p_{ti}$ is the observed frequency for transition, and $p_{tv}$ the frequency of transversion. The substitution model influences both alignment and tree building. For protein sequences, the evolutionary distances from an alignment can be corrected using a Protein Accepted Mutation (PAM) or Jones, Taylor, Thornton (JTT) amino acid substitution matrix whose construction already takes into account the multiple substitutions.

Alternatively, protein equivalents of Jukes–Cantor and Kimura models can be used to correct evolutionary distances. For example, the Kimura model for correcting multiple substitutions in protein distances is:

$$d = -ln(1 - p - 0.2p^2)$$

where $p$ is the observed pairwise distance between two sequences.

At the present time, two elements of the substitution model can be computationally assessed for nucleotide data but not for amino acid or codon data. One element is the model of substitution between particular bases; the other is the relative rate of overall substitution among different sites in the sequence. Substitutions are more frequent between bases that are biochemically more similar. In the case of DNA, the transitions between purine to purine and pyrimidine to pyrimidine are usually more frequent than the transversion between purine to pyrimidine and pyrimidine to purine. Such biases will affect the estimated divergence between two sequences. Specification of the relative rates of substitution among particular residues usually takes the form of a square matrix. The most widely used models of amino acid substitution include distance based methods, which are based on matrixes such as PAM and BLOSUM. Dayhoff's PAM 001 matrix is an empirical model that scales probabilities of change from one amino acid to another in terms of an expected 1% change between two amino acid sequences. Phylogenetic distances are calculated with the assumption that the probabilities in the matrix are correct. There are currently two main categories of tree-building methods. Although any of the parameters in a substitution model might prove critical for a given data set, the best model is not always the one with the most parameters. For a given DNA sequence comparison, a two-parameter model will require that the summed base differences be sorted into two categories and into six for a six parameter model. The number of sites sampled in each of the six categories would be much smaller to give a reliable estimate. For protein sequences, the model used is often dependent on the degree of sequence similarity. For more divergent sequences, the BLOSUM matrices are often better, whereas the PAM matrix is suited for more highly similar sequences.

## C. TREE BUILDING METHOD

Tree building method is one of the steps of construction of phylogenetic trees. These may be divided into Distance based method and character based method.

### a) DISTANCE BASED METHODS

These methods employ the number of changes between each pair in a group of sequences to produce a phylogenetic tree. These methods use the amount of dissimilarity (the distance) between two aligned sequences to derive trees. The distance method was pioneered by Feng and Doolittle. The algorithms for the distance based tree building method can be subdivided into either clustering based or optimality based. The clustering type algorithms compute a tree based on a distance matrix starting from the most similar sequence pairs. These algorithms include an unweighted pair group method using arithmetic average (UPGMA) and neighbour joining (NJ). The optimality based algorithms compare many alternative tree topologies and select one that has the best fit between estimated distances in the tree and the actual evolutionary distances. This category includes the Fitch-Margoliash and minimum evolutionary algorithms.

### 1. *Unweighted Pair Group Method with Arithmetic Mean (UPGMA)*

The UPGMA method is the simplest method of tree construction. It joins tree branches based on the criterion of greatest similarity. It is not strictly an evolutionary distance method. It employs a sequential clustering algorithm, in which local topological relationship are identified in the order of similarity, and the phylogenetic tree is built in a stepwise manner. Firstly, two nodes which are most similar to each other is identified among all nodes and treat these as new single node. Such a node is referred to as a composite node. Subsequently, among the new group of nodes, the pair with highest similarity is identified and so on. UPGMA often produces erroneous tree topologies.

### 2. *Neighbor-Joining (NJ)*

The UPGMA method uses unweighted distances and assumes that all taxa have constant evolutionary rates. Since the molecular clock assumption is often not met in biological sequences, so NJ method can be used, which is somewhat similar to UPGMA in that it builds a tree by using stepwise reduced distance matrices. It does not require that all lineages have diverged by equal amounts. The method is especially suited for datasets comprising lineages with largely varying rates of evolution (Saitou, 1987). The NJ method is a special case of the star decomposition method. The fully resolved tree is decomposed from a fully unresolved star tree by successively inserting branches between a pair of closest neighbours and the remaining terminals in the tree. The raw data are provided as distance matrix and the initial tree is a star tree. Then a modified distance matrix is constructed in which the separation between each pair of nodes is adjusted on the basis of their divergence from all other nodes. The tree is constructed by linking the least-distant pair of nodes in this modified matrix. When two nodes are linked, their common ancestral node is added to the tree and the terminal nodes with their respective branches are removed from the tree. This pruning process converts the newly added common ancestor into a terminal node on a tree of reduced size. At each stage in the process two terminal nodes are replaced by one new node. The process is complete when two nodes remain, separated by a single branch. The NJ method produces an unrooted tree. It is fast and thus suited for large datasets. Sequence information is reduced. The methods is comparatively very fast. Algorithm for finding NJ tree is:

$$d_{AB'} = d_{AB} - 1/2\,x(r_A + r_B)$$

where$d_{AB'}$ is the converted distance between A and B and $d_{AB}$ is the actual evolutionary distance between A and B. The value of $r_A$ (or $r_B$) is the sum of distances of A (or B) to all other taxa.

### 3. *Fitch-Margoliash Least Square Method (FM)*

Optimality based methods have a well-defined algorithm to compare all possible tree topologies and select a tree that best fits the actual evolutionary distance matrix. Based on the differences in optimality criteria, there are two types of algorithms, Fitch–Margoliash and minimum evolution (Fitch, 1967**)**. The Fitch–Margoliash (FM) method selects a best tree among all possible trees based on minimal deviation between the distances calculated in the overall branches in the tree and the distances in the original dataset. It starts by randomly clustering two taxa in a node and creating three equations to describe the distances, and then solving the three algebraic equations for unknown branch lengths. The clustering of the two taxa helps to create a newly reduced matrix. This process is repeated until a tree is completely resolved. The method searches for all tree topologies and selects the one that has the lowest squared deviation of actual distances and calculated tree branch lengths. The optimality criterion is expressed in the following formula:

$$E = \sum_{t=1}^{T} \sum_{j=j+1}^{T} \frac{\left(d_{ij} - p_{ij}\right)^2}{d_{ij}^2}$$

### 4. *Minimum Evolution (ME)*

In the ME method, distance measures that correct for multiple hits at the same sites are used. The construction of a minimum evolution tree is time-consuming because, in principle, the values for all topologies must be evaluated. The number of possible topologies (unrooted trees) rapidly increases with the number of taxa so it becomes very difficult to examine all topologies. While the NJ tree is usually the same as the ME tree, when the number of taxa is small the difference between the NJ and ME trees can be substantial. If a long DNA or amino acid sequence is used, the ME tree is preferable. When the number of nucleotides or amino acids used is relatively small, the NJ method generates the correct topology more often than does the ME method. It constructs a tree with a similar procedure, but uses a different optimality criterion that finds a tree among all possible trees with a minimum overall branch length. The optimality criterion relies on the formula:

$$S = \sum b_i$$

where $b_i$ is the *i*th branch length. Searching for the minimum total branch length is an indirect approach to achieving the best fit of the branch lengths with the original dataset.

## b) CHARACTER BASED METHODS

Character-based methods are based directly on the sequence characters rather than on pairwise distances. A character is a heritable trait possessed by an organism. When amino acid are used we have 20 possible states per position (character), when DNA is used there are 4 states. The actual nucleotide or amino acid occupying a site is the character state. The character-based approaches treat each substitution separately rather than reducing all of the individual variation to a single divergence value. Ancestral sequence can also be inferred. The two most popular character-based approaches are maximum parsimony (MP) and maximum likelihood (ML) methods.

### 1. *Maximum Parsimony (MP)*

The parsimony method chooses a tree that has the fewest evolutionary changes or shortest overall branch lengths. The MP approach is in principal similar to ME approach but the latter is distance based instead of character based. Parsimony tree building works by searching for all possible tree topologies and reconstructing ancestral sequences that require the minimum number of changes to evolve to the current sequences. To save computing time, only a small number of sites that have richest phylogenetic information are used in tree determination. These sites are called informative sites, which are defined as sites that have at least two different kinds of characters, each occurring at least twice. Informative sites are the ones that can often be explained by a unique tree topology. Other sites are non-informative, which are constant sites or sites that have changes occurring only once. Constant sites have the same state in all taxa and are obviously useless in evaluating the various topologies. The sites that have changes occurring only once are not very useful either for constructing parsimony trees because they can be explained by multiple tree topologies. The non-informative sites are thus discarded in parsimony tree construction. Once the informative sites are identified and non-informative sites are discarded, the minimum, number of substitutions at each informative site is computed for a given tree topology. The total number of changes at all informative sites is summed up for each possible tree topology. The tree that has smallest number of changes is chosen as the best tree (Kitching, 1998). The key to counting a minimum number of substitutions for a particular site is to determine the ancestral character states at internal nodes. Because these ancestral character states are not known directly, multiple possible solutions may exist. In this case, the parsimony principal applies to choose the character states that result in a minimum number of substitutions. The inference of an ancestral sequence is made by first going from the leaves to internal nodes and to the common root to determine all possible ancestral character states and then going back from the common root to the leaves to assign sequences that require the minimum number of substitutions.

### *2. Maximum Likelihood (ML)*

Another character-based approach is ML, which uses probabilistic models to choose a best tree that has the highest probability or likelihood of reproducing the observed data (Felsenstein, 1973). It finds a tree that most likely reflects the actual evolutionary process. ML is an exhaustive method that searches every possible tree topology and considers every position in an alignment, not just informative sites. It sometimes also incorporates parameters that account for rate variations across sites. This method uses probability calculations to find a tree that best accounts for the variation in a set of sequences. The likelihood becomes the sum of the probabilities of each possible reconstruction of substitutions under a particular substitution process. The likelihoods for all the sites are multiplied to give an overall "likelihood of the tree" (i.e., the probability of the data given the tree and the substitution process). As one can imagine, for one particular tree, the likelihood of the data is low at some sites and high at others. For a "good" tree, many sites will have higher likelihood, so the product of likelihoods is high. For a "poor" tree, the reverse will be true. The method is similar to the maximum parsimony method in that the analysis is performed on each column of a multiple sequence alignment. All possible trees are considered. Hence, the method is only feasible for a small number of sequences. The number of sequence changes or mutations that may have occurred to give the sequence variation is considered for each tree. Because the rate of appearance of new mutations is very small, the more mutations needed to fit a tree to the data, the less likely that tree. Thus, the method can be used to explore relationships among more diverse sequences, conditions that are not well handled by maximum parsimony methods. The main disadvantage of maximum likelihood methods is this method uses great amounts of computational time, it is usually impractical to perform a complete search that simultaneously optimizes the substitution model and the tree for a given data set. However, with faster computers, the maximum likelihood method is seeing wider use and is being used for more complex models of evolution. ML works by calculating the probability of a given evolutionary path for a particular extant sequence. The probability values are determined by a substitution model (either for nucleotides or amino acids). For example, for DNA sequences using the Jukes–Cantor model, the probability ($P$) that a nucleotide remains the same after time $t$ is:

$$P(t) = 1/4 + 3/4\, e^{-\alpha t}$$

where $\alpha$ is the nucleotide substitution rate in the Jukes–Cantor model, which is either empirically assigned or estimated from the raw datasets. The most commonly used heuristic ML method is called quartet puzzling, which uses a divide-and-conquer approach.

## PHYLOGENETIC ANALYSIS USING BIOINFORMATICS TOOLS

Bioinformatics has transformed the discipline of biology from a purely lab-based science to an information science as well. Now it becomes easier to do phylogenetic analysis by using different softwares. Some of the softwares are free (PHYLIP) and some are not free (PAUP). To do phylogeny with the help of bioinformatics tools it is easier to get results.

**PHYLIP** (the *PHYL*ogeny *I*nference *P*ackage)

PHYLIP is the most widely-distributed phylogeny package. It is a package of programs for inferring phylogenies (evolutionary trees) freely available on web. Methods that are available in the package include parsimony, distance matrix, and likelihood methods and bootstrapping. Data types that can be handled include molecular sequences, gene frequencies, restriction sites and fragments, distance matrices, and discrete characters. The data are read into the program from a text file, which the user can prepare using any word processor.

Programs of the PHYLIP package that make distance matrix include the following programs DNADIST computes distances among input nucleic acid sequences. PROTDIST computes a distance measure for protein sequences, based on the Dayhoff PAM model. Distance analysis programs in PHYLIP includes FITCH which estimates a phylogenetic tree assuming additivity of branch lengths using the Fitch-Margoliash method and does not assume a molecular clock. KITSCH estimates a phylogenetic tree using the Fitch-Margoliash method but under the assumption of a molecular clock. NEIGHBOR estimates phylogenies using the neighbor-joining or UPGMA method.

The main programs for maximum parsimony analysis in the PHYLIP package are DNAPARS which treats gaps as a fifth nucleotide state. DNAPENNY which performs parsimonious phylogenies by branch-and-bound search that can analyze more sequences. DNACOMP, which performs phylogenetic analysis using the compatibility criterion. Rather than searching for overall parsimony at all sites in the multiple sequence alignment, this method finds the tree that supports the largest number of sites. This method is recommended when the rate of evolution varies among sites. DNAMOVE which performs parsimony and compatibility analysis interactively. For analysis of protein sequences, the program is: PROTPARS which counts the minimum number of mutations to change a codon for the first amino acid into a codon for the second amino acid, but only scores those mutations in the mutational path that actually change the amino acid.

PHYLIP includes two programs for maximum likelihood analysis DNAML estimates phylogenies from nucleotide sequences by the maximum likelihood method, allowing for variable frequencies of the four nucleotides, for unequal rates of transitions and transversions. DNAMLK estimates phylogenies from nucleotide sequences by the maximum likelihood method in the same manner as DNAML, but assumes a molecular clock. One starts with an evolutionary model of sequence change that provides estimates of rates of substitution of one base for another in a set of nucleic acid sequences. Once the analysis have done then we have to see the phylogenetic tree by choosing the program DRAWGRAM which made rooted tree and DRAWTREE which made unrooted tree.

## D) TREE RELIABILITY

Although various methods have been developed for reconstructing phylogenetic trees, there exist few methods for evaluating the statistical confidence of an inferred phylogeny or for testing whether one phylogeny is significantly better than another. There are two questions that need to be answered in assessing reliability. One is how reliable the tree or a portion of the tree is; and the second is whether this tree is significantly better than another tree. To answer the first question, we need to use analytical resampling strategies such as bootstrapping and jackknifing, which repeatedly resample data from the original dataset. For the second question, conventional statistical tests are needed. Bootstrapping is a statistical technique that tests the sampling errors of a phylogenetic tree. It does so by repeatedly sampling trees through slightly changed datasets. The robustness of the original tree can be assessed by this way. The rationale for bootstrapping is that a newly constructed tree is possibly biased owing to incorrect alignment or chance fluctuations of distance measurements. To determine the robustness or reproducibility of the current tree, trees are repeatedly constructed with slightly disturbed alignments that have some random fluctuations introduced. A truly robust phylogenetic relationship should have enough characters to support the relationship even if the dataset is disturbed in such a way. Otherwise, the noise introduced in the resampling process is sufficient to generate different trees, indicating that the original topology may be derived from weak phylogenetic signals. Thus, this type of analysis gives an idea of the statistical confidence of the tree topology. Bootstrap resampling relies on redistribution of original sequence datasets. There are two redistribution strategies. One way to produce disturbances by random replacement of sites. This is referred to as Nonparametric bootstrapping. Another disturbance is by making new datasets based on a particular sequence distribution, which is Parametric bootstrapping. Both types of bootstrapping can be applied to the distance, parsimony, and likelihood tree construction methods. A large number of bootstrap resampling steps are needed to achieve meaningful results. It is generally recommended that a phylogenetic tree should be bootstrapped 500 to 1,000 times. On the basis of simulation studies, it has been suggested that, under favorable conditions bootstrap values greater than 70% correspond to a probability of greater than 95% that the true phylogeny has been found. Under less favorable conditions, bootstrap values greater than 50% will be overestimates of accuracy. Simply put under certain conditions high bootstrap values can make the wrong phylogeny look good; therefore, the conditions of the analysis must be considered. Bootstrapping can be used in experiments in which trees are recomputed after internal branches are deleted one at a time. Bootstrapping does not assess the accuracy of a tree, but only indicates consistency and stability of individual clades of the tree. This means that, because of systematic errors, wrong trees can still be obtained with high bootstrap values. Therefore, bootstrap results should be interpreted with caution. Unusually high GC content in the original dataset, unusually accelerated evolutionary rates and unrealistic evolutionary models are the potential causes for generating biased trees, as well as biased bootstrap estimates, which come after the tree generation. In jackknifing, one half of the sites in a dataset are randomly deleted, creating datasets half as long as the original. Each new dataset is subjected to phylogenetic tree construction using the same method as the original. The advantage of jackknifing is that sites are not duplicated relative to the original dataset and that computing time is much shortened because of shorter sequences. One disadvantage of this approach is that the size of datasets has been changed into one half and that the datasets are no longer considered replicates. The statistical methodology for testing phylogenies is in a primitive state. This is because of two reasons. First, phylogenetic reconstruction has long been recognized as a problem in statistical inference few authors have formulated the problem in a statistical framework. Most current methods give one or a few trees and do not provide information concerning the confidence level of estimated phylogenies. Second, the problem is complex, because the number of possible alternative trees is large even when only a moderate number of taxa are involved. For this reason, most current statistical tests are heuristic when the number of taxa involved is five or larger. The Bayesian method is probably the most efficient statistical tests; it does not require bootstrapping because the Markov chain Monte Carlo (MCMC) procedure itself involves thousands or millions of steps of resampling. As a result of Bayesian tree construction, posterior probabilities are assigned at each node of a best Bayesian tree as statistical support. Because of fast computational speed of MCMC tree searching, the Bayesian method offers a practical advantage over regular maximum likelihood (ML) and makes the statistical evaluation of ML trees more feasible. Unlike bootstrap values, Bayesian probabilities are normally higher because most trees are sampled near a small number of optimal trees. Therefore, they have a different statistical meaning from bootstrap. The Kishino–Hasegawa (KH) test The KH test sets out to test the null hypothesis that the two competing tree topologies are not significantly different. A paired student $t$-test is used to assess whether the null hypothesis can be rejected at a statistically significant level. In this test, the difference of branch lengths at each informative site between the two trees is calculated. The standard deviation of the difference values can then be calculated. This in turn allows derivation of a $t$-value which is used for evaluation against the $t$-distribution to see whether the value falls within the significant range to warrant the rejection of the null hypothesis

$$t = \frac{D_a - D_t}{Sd/\sqrt{n}} \sim t_{n-1}$$

where *n* is the number of informative sites, *t* is the test statistic value, $D_a$ is the average site-to-site difference between the two trees, $Sd$ is the standard deviation, and $D_t$ is the total difference of branch lengths of the two trees.

**References**

1. Felsenstein, J. (2004). Inferring Phylogenies. Sunderland, MA: Sinauer Associates.
2. Felsenstein, J. (1973). Maximum likelihood estimation of evolutionary trees from continuous characters. *Am. J. Hum. Gen.,* 25: 471-492.
3. Fitch, W. and Margoliash, E. (1967). The construction of phylogenetic trees. *Science,* 155: 279-284.
4. Kitching, I. J., Forey, P. L., Humphries, C. J., and Williams, D. M. (1998). Cladistics: The Theory and Practice of Parsimony Analysis.Second Edition.The Systematics Association Publication No. 11. Oxford: Oxford University Press.
5. Saitou, N. and Nei, M. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.,* 4: 406-425.

*****

# DNA Signature based SNP and SSR Mining

**M. A. Iquebal[1], Sarika[1], Anil Rai[2] and Dinesh Kumar[1]**

**[1]ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

**[2]Indian Council of Agricultural Research, New Delhi**

## 1. Introduction

Molecular characterisation of genetic resources has been adding objectivity and rationality in decision making for conservation. Plant, animal, fish and microbial genetic resources are being characterised by various molecular markers, predominantly by microsatellite, AFLP and SNP covering both nuclear genome as well as mitochondrial genome. These molecular markers have inbuilt "molecular clock" entrained with evolutionary time scale having "pictures" or "signatures" of speciation and differentiation of dynamic germplasm in evolutionary pace and scale. Bioinformatics has not only revolutionised the germplasm characterisation, but had been proven as indispensable tool for molecular identification of species. Bioinformatics has become most powerful tool of taxonomy right from microbial meta-genome analysis of hitherto uncultured microbes, plant, animal and fish species identification. Advances in genome analysis technology are providing an unprecedented amount of information about animals, bacterial and viral organisms, and hold great potential for pathogen detection and identification. Here, a rational approach to the development and application of nucleic acid signatures is described based on SNP and STR nucleotides. Other bioinformatics tools for classification and prediction of such molecular data has also been discussed.

## 2. DNA barcoding of species and its origin

DNA barcoding is a taxonomic method that uses a short genetic marker in an organism's mitochondrial DNA to identify it as belonging to a particular species. It is based on a relatively simple concept: most eukaryote cells contain mitochondria and mitochondrial DNA (mtDNA) has a relatively fast mutation rate, which results in significant variance in mtDNA sequences between species and, in principle, a comparatively small variance within species. A 648-bp region of the cytochrome c oxidase subunit I gene (COI) was initially proposed as a potential 'barcode'.

The use of nucleotide sequence variations to investigate evolutionary relationships is not a new concept. Carl Woese used sequence differences in ribosomal RNA (rRNA) to discover archaea, which in turn led to the redrawing of the evolutionary tree, and molecular markers (e.g., allozymes, rDNA, and mtDNAvage ). DNA barcoding provides a standardised method for this process via the use of a short DNA sequence from a particular region of the genome to provide a 'barcode' for identifying species. In 2003, Paul D.N. Hebert from the University of Guelph, Ontario, Canada, proposed the compilation of a public library of DNA barcodes that may be linked to named specimens. This library would "provide a new master key for identifying species, one whose power will rise with increased taxon coverage and with faster, cheaper sequencing".

### *2.1 Identification of birds by species bar code*

In an effort to find a correspondence between traditional species boundaries established by taxonomy and those inferred by DNA barcoding, Hebert and co-workers sequenced DNA barcodes of 260 of the 667 bird species that breed in North America (Hebert et al. 2004a). It was found that every single one of the 260 species had a different COI sequence. 130 species were represented by two or more specimens. In all of these species, COI sequences were either identical or were most similar to sequences of the same species. COI variations between species averaged 7.93%, whereas variation within species averaged 0.43%. In four cases, there were deep intraspecific divergences, indicating possible new species. Three out of these four polytypic species are already split into two by some taxonomists. Hebert et al.'s (2004a) results reinforce these views and strengthen the case for DNA barcoding. They also proposed a standard sequence threshold to define new species, this threshold, the so-called "barcoding gap", was defined as 10 times the mean intraspecific variation for the group under study.

### *2.2 Delimiting cryptic species by DNA bar code*

The next major study into the efficacy of DNA barcoding was focused on the neotropical skipper butterfly, *Astraptesfulgerator* at the Area Conservacion de Guanacaste (ACG) in north-western Costa Rica. This species was already known as a cryptic species complex, due to subtle morphological differences, as well as an unusually large variety of caterpillar food plants. However, several years would have been required for taxonomists to completely delimit species. Hebert et al. (2004b) sequenced the COI gene of 484 specimens from the ACG. This sample included “at least 20 individuals reared from each species of food plant, extremes and intermediates of adult and caterpillar color variation, and representatives” from the three major ecosystems where *Astraptesfulgerator*was found. Hebert et al. (2004b) concluded that *Astraptesfulgerator* consists of 10 different species in north-western Costa Rica. This highlights that the results of DNA barcoding analyses can be dependent upon the choice of analytical methods used by the investigators, so the process of delimiting cryptic species using DNA barcodes can be as subjective as any other form of taxonomy.

### *2.3 Identifying flowering plants by species DNA bar code*

Kress et al. (2005) suggest that the use of the COI sequence “is not appropriate for most species of plants because of a much slower rate of cytochrome c oxidase I gene evolution in higher plants than in animals”. A series of experiments was then conducted to find a more suitable region of the genome for use in the DNA barcoding of flowering plants.

Three criteria were set for the appropriate genetic loci:

i. Significant species-level genetic variability and divergence
ii. An appropriately short sequence length so as to facilitate DNA extraction and amplification, and
iii. The presence of conserved flanking sites for developing universal primers.

At the conclusion of these experiments, Kress et al. (2005) proposed the nuclear internal transcribed spacer region and the plastid trnH-psbAintergenic spacer as a potential DNA barcode for flowering plants. These results suggest that DNA barcoding, rather than being a 'master key' may be a 'master keyring', with different kingdoms of life requiring different keys.

### *2.4 Strain identification of fungi*

*Pucciniagraminis*, the causal agent of stem rust, has caused serious disease of small cereal grains (wheat, barley, oat, and rye) worldwide. *P. graminis* is the first sequenced representative of the rust fungi (Uredinales), which are obligate plant pathogens. The rust fungi comprise more than 7000 species and are one of the most destructive groups of plant pathogens. Stem rust of wheat has been a serious problem wherever wheat is grown and has caused major epidemics in North America. In 1999, a new highly virulent race TTKS (Ug99) of *P. graminis* was identified in Uganda, and since then has spread, causing a widening epidemic in Kenya and Ethiopia.

Bioinformatics can play very critical role in identification of species as well as strains and also its dynamics across globe. The plethora of data both from host and parasite generated by using latest molecular or biotechnological tools can easily be analysed by bioinformatics tools. The talk will focus on Ug99 race of *P. graminis*. How the genome of it can be used to track the movement of this fungal species and how the bioinformatics tools can be helpful in strain identifcation*P. graminis* including Ug99 identification.

## 3. DNA based signature of domestic species and animal breeds

Mitochondrial DNA markers have been proved to be successful in many species of domestic animals, being used especially for meat identification, poaching of wild animals, adulteration of dairy milk, dairy products(like cheese) of various domestic animal species.

The prevalent markers used for the breeds are almost STR but very recently the SNP based chip has proven its accuracy for breed signature along with details of admixture as well as very powerful for parentage and pedigree.

### *3.1 STR based signatures of breeds*

A question has generally been asked at various scientific fora with regard to molecular characterization of breeds as to whether a livestock breed can be identified from a sample of blood, semen, hair, blood spot, carcass etc. Various attempts have been made in the last couple of years by the molecular geneticists of the world to answer this question. Some studies have succeeded in developing a technology for breed certification and breed-specific genetic/DNA signature in different breeds of cattle in Spain, Portugal and France; horses in Norway, sheep in Spain, and camel in Kenya. The degree of accuracy of certification of a breed in these studies was very high ranging between 95% to 99%.

Three methods viz (i) Frequency method (Paetkau et al., 1995), (ii) Bayesian method (Rannala et al, 1997) and (iii) Distance methods (Goldstein et al 1995) have been used for developing breed specific signatures. The Bayesian method has been reported to be more accurate with microsatellite data to the extent of > 99% confidence limits (Corander et al., 2003, Bustamante et al., 2003).

In the foreign countries, few attempts have been made to develop genetic signatures of some breeds of livestock in the recent past. For cases of doubtful breed identity where it becomes difficult to assign an individual to a particular breed due to individual being an admixture of breeds, the studies have been made to develop breed hybrid index. The review of literature has therefore been made under two headings: (i) Development of breed-specific signatures/profiles and (ii) Development of breed hybrid index.

*3.2 SNP chip based DNA signature of breeds*

In Japan, Japanese Black and Holstein cattle are appreciated as popular sources of meat, and imported beef from Australia and the United States is also in demand in the meat industry. Since the BSE outbreak, the problem of false sales has arisen: imported beef has sometimes been mislabelled as domestic beef due to consumer concerns. A method is needed to correctly discriminate between Japanese and imported cattle for food safety. The SNP 50K based chip can discrimination markers between Japanese and US cattle. There is a report where five US-specific markers (BISNP7, BISNP15, BISNP21, BISNP23, and BISNP26) has been developed with allelic frequencies that ranged from 0.102 (BISNP15) to 0.250 (BISNP7) and averaged 0.184. The combined use of the five markers would permit discrimination between Japanese and US cattle with a probability of identification of 0.858. This result indicates the potential of the bovine 50K SNP array as a powerful tool for developing breed identification markers. These markers would contribute to the prevention of falsified beef displays in Japan (Suekawa*et al* 2010, Sasazaki*et al* 2011).

## 4. DNA based signature of plant variety, example-Basmati rice

Basmati rice has a typical pandan-like (*Pandanusamaryllifolius* leaf) flavour caused by the aroma compound 2-acetyl-1-pyrroline.Difficulty in differentiating genuine traditional basmati from pretenders and the significant price difference between them has led fraudulent traders to adulterate traditional basmati. To protect the interests of consumers and trade, a PCR-based assay similar to DNA fingerprinting in humans allows for the detection of adulterated and non-basmati strains. Its detection limit for adulteration is from 1% upwards with an error rate of ±1.5%. Exporters of basmati rice use 'purity certificates' based on DNA tests for their basmati rice consignments.It was developed at the Centre for DNA Fingerprinting and Diagnostics, Labindia, an Indian company has released kits to detect basmati adulteration. World's First Single-tube, Multiplex(co-amplify eight microsatellite loci) Microsatellite Assay-based Kit for Basmati Authentication.

The Basmati Verifiler™ Kit is the world's first product for establishing the authenticity of Basmati rice samples via a molecular assay. The kit uses a PCR amplification technique based on Simple Sequence Repeats (SSR) that provides the single most discriminating assay for Basmati genotyping.

## 5. DNA based bar-coded signature of fishes

Ward et al (2005) described in a paper the potential of cox1 sequencing, or ‘barcoding’, in to identification of fish species. In this study, two hundred and seven species of fish, mostly Australian marine fish, were sequenced (bar coded) for a 655 bp region of the mitochondrial cytochrome oxidase subunit I gene (cox1). Most species were represented by multiple specimens, and 754 sequences were generated. The GC content of the 143 species of teleosts was higher than the 61 species of sharks and rays (47.1% versus 42.2%), largely due to a higher GC content of codon position 3 in the former (41.1% versus 29.9%). Rays had higher GC than sharks (44.7% versus 41.0%), again largely due to higher GC in the 3rd codon position in the former (36.3% versus 26.8%). Average within-species, genus, family, order and class Kimura two parameter (K2P) distances were 0.39%, 9.93%, 15.46%, 22.18% and 23.27%, respectively. All species could be differentiated by their cox1 sequence, although single individuals of each of two species had haplotypes characteristic of a congener. Although DNA barcoding aims to develop species identification systems, some phylogenetic signal was apparent in the data. In the neighbour-joining tree for all 754 sequences, four major clusters were apparent: chimaerids, rays, sharks and teleosts. Species within genera invariably clustered, and generally so did genera within families. Three taxonomic groups—dogfishes of the genus Squalus, flatheads of the family Platycephalidae, and tunas of the genus Thunnus—were examined more closely. The clades revealed after bootstrapping generally corresponded well with expectations. Individuals from operational taxonomic units designated as Squalus species B through F formed individual clades, supporting morphological evidence for each of these being separate species. This paper is still widely cited for DNA based fish signature.

## 6. Different bioinformatics tool for classification and prediction of molecular data

Advances in genome analysis technology are providing an unprecedented amount of information about animals, bacterial and viral organisms, and hold great potential for pathogen detection and identification. In this section, a rational approach to the development and application of nucleic acid signatures is described based on SNP and STR nucleotides.

Regardless of the origin of the SNPs (e.g., sequencing and public databases), once SNPs from a target organism and its nearest neighbours have been collected, it is necessary to identify those SNPs that will be useful for species and strain identification. The approach that has been taken is to use a database of SNP markers to enable phylogenetic analysis to identify evolutionary clades and the SNPs that define them. The need for large data storage capability, which facilitates data accessibility, automated SNP prediction (with reduction in manual intervention), signature delineation and facilitated complex query capability, has been recognized. Many databases exist as local resources, although some universal databases housing eukaryotic SNP data have been established (e.g., dbSNP). Such global databases have not been developed for microbial SNP data. Each database created for SNP discovery and phylogenetic analysis will have different content and different structure that are determined by the uses of the data. There is no single correct way to design a database but essential content is necessary not only to allow different polymorphism databases to communicate but to provide essential information for analysis of the data. Four essential core elements have been defined and include:

- ✓ A unique SNP identifier (allele)
- ✓ The data source (e.g., experimental or computational)
- ✓ The sequence flanking the allele and the allele(s)

Many databases have been created for the storage and analysis of eukaryotic SNP data, some are comprehensive or genomewide, and others are specialized or locus-specific. Both types of databases are essential. The comprehensive database will provide a genome-wide view of polymorphism, ideal for strain typing and identification. The locus-specific database will provide a more in-depth view of polymorphisms at a particular locus. A database should incorporate accurate information that can be used for downstream analyses and have the ability to integrate with other databases. Some additional information associated with SNPs should be implemented in the databases. A database and its associated pipeline should be able to process and store data from a variety of sources, not only from a sequencing machine but external sequence databases (e.g., GenBank, dbEST). The database should track the organism and project to which a SNP belongs along with genome-, gene- and exon-specific information related to a SNP. A downstream analysis requires not just flanking sequences but also a reference sequence. Other information useful for quality assurance purposes and general data analysis include the algorithm by which a SNP was discovered and whether it was validated experimentally or not validated but computationally predicted and the method by which it was validated (e.g., genotyping assay or sequencing). The type of SNP should also be included (e.g., homozygous or heterozygous) along with the average allele frequency. Useful information, such as the position of the SNP relative to its reference sequence, contig or amplicon and whether the SNP is silent or pathogenic should be incorporated. To meet the needs of signature development, a relational database has been created to store information related to SNP discovery and downstream assay development. The information specific to SNP discovery and assay design is stored logically in database tables or entities enabling complex queries on SNPs and related data. Specifically, the SNP table includes, in addition to the SNP site alleles, the 5´ and 3´ flanking sequences for assay design. Information related to the gene, exon and project are stored to facilitate downstream analysis, such as population studies. Algorithm-specific rank values and method are included, which enable the investigator to assess the actual quality of each SNP. The SNP table is the central entity in the database. Associated with each SNP is a name where each SNP can have more than one name. Each SNP can also be associated with one or more reference sequences. Reference sequences have multiple purposes including:

- ✓ Serving as a template for PCR primer design
- ✓ Providing flanking sequence around a SNP
- ✓ Being included in a Phrap assembly to ensure an accurate assembly

Reference sequences also provide a starting point for functional annotation. The reference sequence has associated with it a name, GenBank accession or GI number, description and sequence. Amplicons are sequences used for SNP prediction. Associated with an amplicon is information, such as the name and description of each amplicon, primers used for its amplification and its expected size. Even though this database was designed for higher eukaryotes and their viruses, the data relationships will remain the same for prokaryotic SNP data. The SNP marker database serves as the repository of information required for downstream signature development and assay design activities.

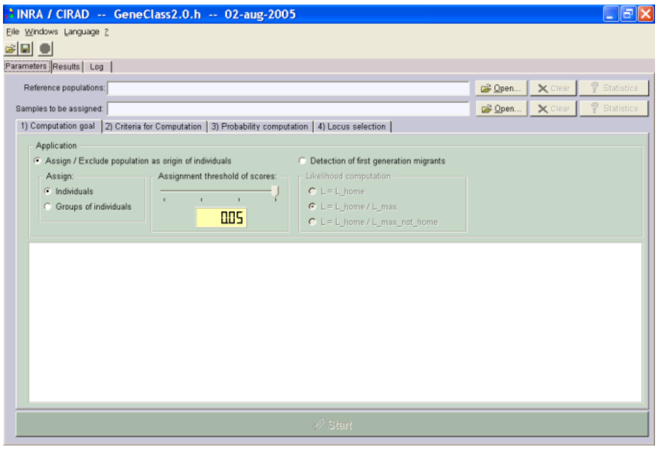
Protocols and basic information of Bioinformatics tools which are important to search SNP, Sequence data analysis, STR data Analysis, and to develop SNP/STR based DNA signatures are shown below:

### *6.1GeneClass 2.0*

The effectiveness of Single Nucleotide Polymorphisms (SNPs) for the assignment of various breeds of cattle and buffalo has already been investigated by analysing numerous SNPs. Breed assignment has been performed by comparing the Bayesian and frequency methods implemented in the STRUCTURE 2.2 and GENECLASS 2 software programs. The use of SNPs for the reallocation of known individuals to their breeds of origin and the assignment of unknown individuals has already been tested. Exampleisgiven with GeneClass2 in Buffalo having reference and unknown data of buffalo breeds (Figure 1 and Figure 2). The steps are as follows

Step 1: Download the GeneClass2 Software(Freely available at
http://www.montpellier.inra.fr/URLB/geneclass/geneclass.html).
Step 2. Preparation of data files for reference and unknown samples.
Step 3. Open the main window of the software (Figure 1) and import both files.

Step 4.Choice of the parameters like Computational goal, Criteria for computation, Probability computation and Selection Criteria.
Step 5. By clicking on the start button we can see the result (Figure 2) and finally interpretation of the result can be drawn.



Reference populations:
Samples to be assigned:

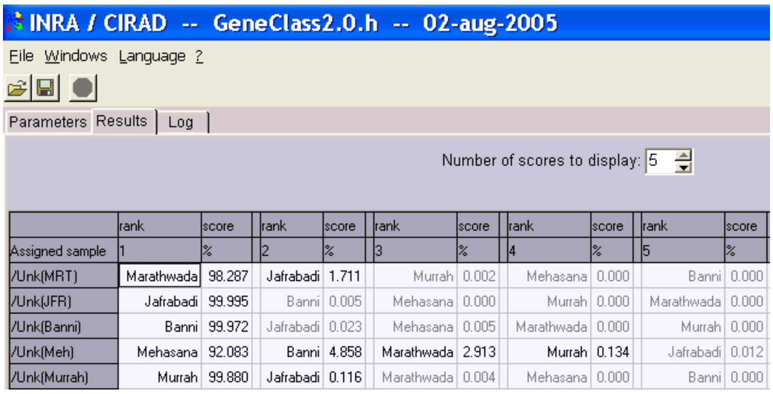
Application
● Assign / Exclude population as origin of individuals
○ Detection of first generation migrants

Assign:
● Individuals
○ Groups of individuals

Assignment threshold of scores: 0.05

Likelihood computation
○ L = L_home
● L = L_home / L_max
○ L = L_home / L_max_not_home

Figure 1. Main window of GeneClass2.0 Software



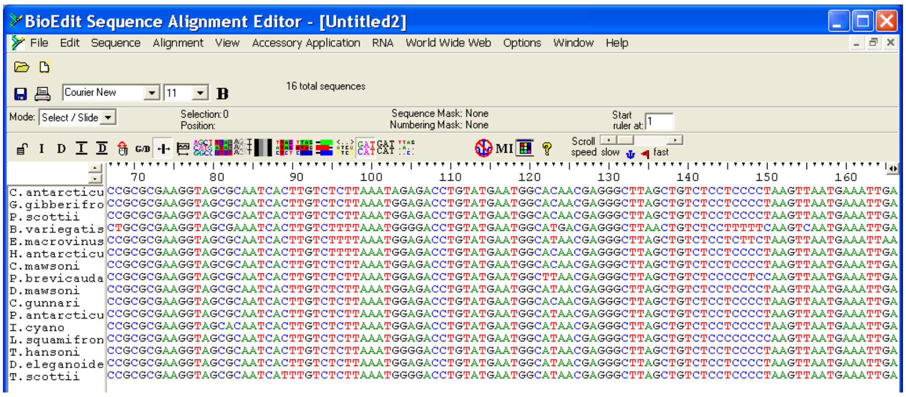
Number of scores to display: 5

| | rank | score | rank | score | rank | score | rank | score | rank | score |
|---|---|---|---|---|---|---|---|---|---|---|
| Assigned sample | 1 | % | 2 | % | 3 | % | 4 | % | 5 | % |
| /Unk(MRT) | Marathwada | 98.287 | Jafrabadi | 1.711 | Murrah | 0.002 | Mehasana | 0.000 | Banni | 0.000 |
| /Unk(JFR) | Jafrabadi | 99.995 | Banni | 0.005 | Mehasana | 0.000 | Murrah | 0.000 | Marathwada | 0.000 |
| /Unk(Banni) | Banni | 99.972 | Jafrabadi | 0.023 | Mehasana | 0.005 | Marathwada | 0.000 | Murrah | 0.000 |
| /Unk(Meh) | Mehasana | 92.083 | Banni | 4.858 | Marathwada | 2.913 | Murrah | 0.134 | Jafrabadi | 0.012 |
| /Unk(Murrah) | Murrah | 99.880 | Jafrabadi | 0.116 | Marathwada | 0.004 | Mehasana | 0.000 | Banni | 0.000 |

Figure 2. Identification of 5 unknown breeds of Buffalo with reference data.

*6.2 BioEdit*

BioEdit is a mouse-driven, easy-to-use sequence alignment editor and sequence analysis tool. This tool can handle most simple sequence and alignment editing and manipulation functions that researchers are likely to do on a daily basis, as well as a few basic sequences analyses. For example alignment of different nucleotide sequence of various bacterial strains in Figure 1 and Figure 2. The steps are as follows:
File→Newalignment→Import→AccessaryApplications→ClustalWAlignment→Multiple Alignment (Figure 3) and to see the Alignment result View→ViewMode→Identity/similarity (Figure 4).



```
C.antarcticu CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATAGAGACCTGTATGAATGGCACAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
G.gibberifro CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCACAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
P.scottii    CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCACAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
B.variegatis CTGCGCGAAGGTAGCGAAATCACTTGTCTTTTAAATGGGGACCTGTATGAATGGCATGACGAGGGCTTAACTGTCTCCTTTTTCAAGTCAATGAAATTGA
E.macrovinus CCGCGCGAAGGTAGCGCAATCACTTGTCTTTTAAATGGAGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCTCTTCTAAGTTAATGAAATTAA
H.antarcticu CCGCGCGAAGGTAGCGCAATCACTTGTCTTTTAAATGGAGACCTGTATGAATGGCACAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
C.mawsoni    CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCACAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
P.brevicauda CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCTTAACGAGGGCTTAGCTGTCTCCCCCTCCAAGTTAATGAAATTGA
D.mawsoni    CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
C.gunnari    CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCACAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
P.antarcticu CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
I.cyano      CCGCGCGAAGGTAGCACAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
L.squamifron CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCCCCCCCAAGTTAATGAAATTGA
T.hansoni    CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGGGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
D.eleganoide CCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
T.scottii    CCGCGCGAAGGTAGCGCAATCATTTGTCTCTTAAATGGGGACCTGTATGAATGGCATAACGAGGGCTTAGCTGTCTCCTCCCCTAAGTTAATGAAATTGA
```

Figure 3. Nucleotide Sequence Data (16 Different Microbial strains) import in the main window

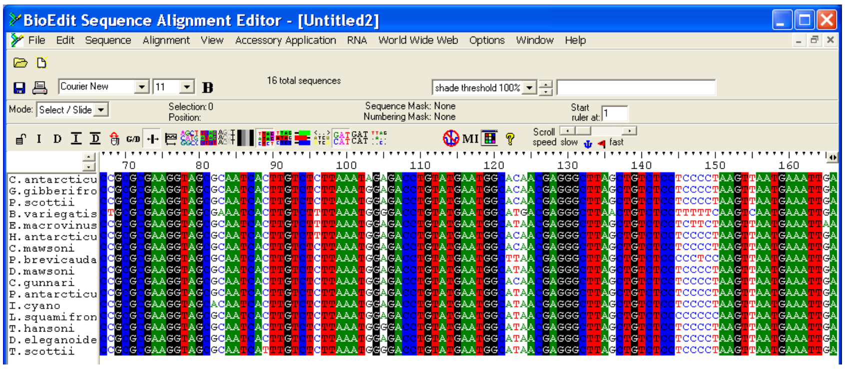
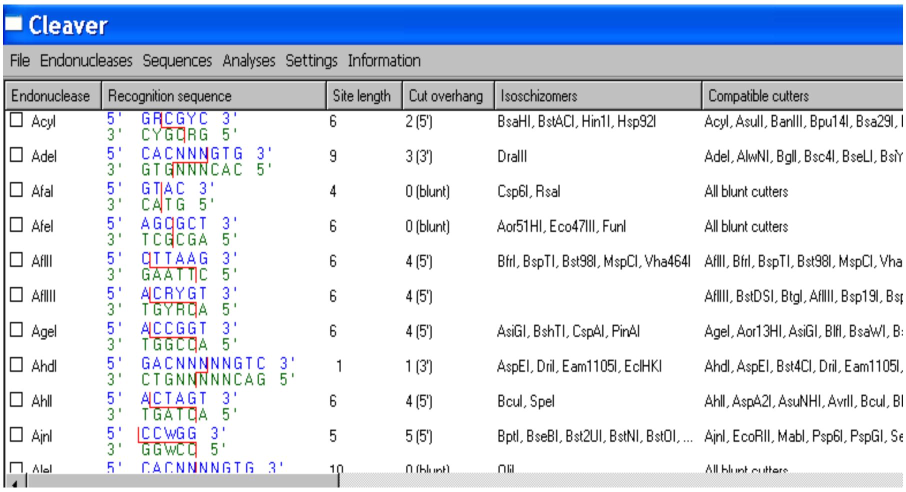


Figure 4. Alignment of all sequences showing nucleotide differences

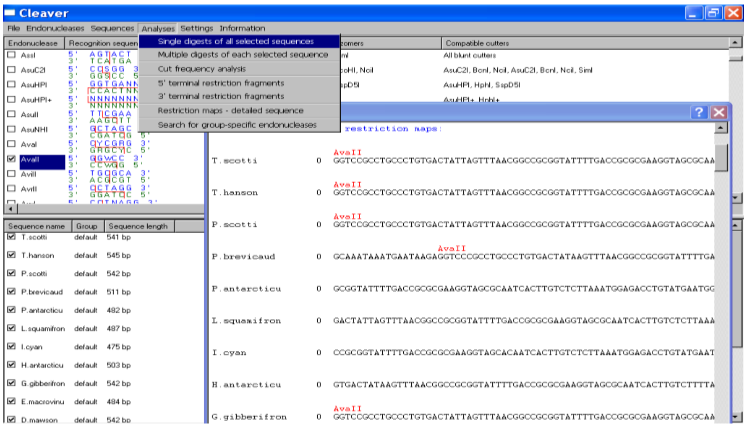
### *6.3 Cleaver*

Cleaver is an application for identifying restriction endonuclease recognition sites that occur in some taxa (Jarman, 2006). Differences in DNA fragment restriction patterns among taxa are the basis for many diagnostic assays for taxonomic identification; and are used in some procedures for removing the DNA of some taxa from pools of DNA from mixed sources. Cleaver analyses restriction digestion of groups of orthologous DNA sequences simultaneously to allow identification of differences in restriction pattern among the fragments derived from different taxa. Cleaver is freely available without registration from its website (http://cleaver.sourceforge.net/). The program can be run as a script for computers that have Python 2.3 and necessary extra modules installed. This allows it to run on Gnu/Linux, Unix, MacOSX and Windows platforms. Standalone executable versions for Windows and MacOSX operating systems are also available. The protocol for using the software is shown in Figure 5 and Figure 6.



| Endonuclease | Recognition sequence | Site length | Cut overhang | Isoschizomers | Compatible cutters |
|---|---|---|---|---|---|
| □ AcyI | 5' GRCGYC 3'<br>3' CYGCRG 5' | 6 | 2 (5') | BsaHI, BstACI, Hin1I, Hsp92I | AcyI, AsuII, BanIII, Bpu14I, Bsa29I, I |
| □ AdeI | 5' CACNNNGTG 3'<br>3' GTGNNNCAC 5' | 9 | 3 (3') | DraIII | AdeI, AlwNI, BglI, Bsc4I, BseLI, BsiY |
| □ AfaI | 5' GTAC 3'<br>3' CATG 5' | 4 | 0 (blunt) | Csp6I, RsaI | All blunt cutters |
| □ AfeI | 5' AGCGCT 3'<br>3' TCGCGA 5' | 6 | 0 (blunt) | Aor51HI, Eco47III, FunI | All blunt cutters |
| □ AflII | 5' CTTAAG 3'<br>3' GAATTC 5' | 6 | 4 (5') | BfrI, BspTI, Bst98I, MspCI, Vha464I | AflII, BfrI, BspTI, Bst98I, MspCI, Vha |
| □ AflIII | 5' ACRYGT 3'<br>3' TGYRCA 5' | 6 | 4 (5') | | AflIII, BstDSI, BtgI, AflIII, Bsp19I, Bsp |
| □ AgeI | 5' ACCGGT 3'<br>3' TGGCCA 5' | 6 | 4 (5') | AsiGI, BshTI, CspAI, PinAI | AgeI, Aor13HI, AsiGI, BlfI, BsaWI, B: |
| □ AhdI | 5' GACNNNNNGTC 3'<br>3' CTGNNNNNCAG 5' | 1 | 1 (3') | AspEI, DriI, Eam1105I, EclHKI | AhdI, AspEI, Bst4CI, DriI, Eam1105I, |
| □ AhlI | 5' ACTAGT 3'<br>3' TGATCA 5' | 6 | 4 (5') | BcuI, SpeI | AhlI, AspA2I, AsuNHI, AvrII, BcuI, BI |
| □ AjnI | 5' CCWGG 3'<br>3' GGWCC 5' | 5 | 5 (5') | BptI, BseBI, Bst2UI, BstNI, BstOI, ... | AjnI, EcoRII, MabI, Psp6I, PspGI, Se |
| □ AleI | 5' CACNNNNGTG 3' | 10 | 0 (blunt) | OliI | All blunt cutters |

Figure 5. Main Window of Cleaver Software



Cleaver
File Endonucleases Sequences Analyses Settings Information
Single digests of all selected sequences
Multiple digests of each selected sequence
Cut frequency analysis
5' terminal restriction fragments
3' terminal restriction fragments
Restriction maps - detailed sequence
Search for group-specific endonucleases
Endonuclease
Recognition sequen
Compatible cutters
AssI
AsuC2I
AsuHPI
AsuHPI+
AsuII
AsuNHI
AvaI
AvaII
AviI
AvrII
All blunt cutters
AsuC2I, BcnI, NciI, AsuC2I, BcnI, NciI, SimI
AsuHPI, HphI, SspD5I
restriction maps:

| Sequence name | Group | Sequence length |
|---|---|---|
| ☑ T.scotti | default | 541 bp |
| ☑ T.hanson | default | 545 bp |
| ☑ P.scotti | default | 542 bp |
| ☑ P.brevicaud | default | 511 bp |
| ☑ P.antarcticu | default | 482 bp |
| ☑ L.squamifron | default | 487 bp |
| ☑ I.cyan | default | 475 bp |
| ☑ H.antarcticu | default | 503 bp |
| ☑ G.gibberifron | default | 542 bp |
| ☑ E.macrovinu | default | 484 bp |
| ☑ D.mawson | default | 542 bp |

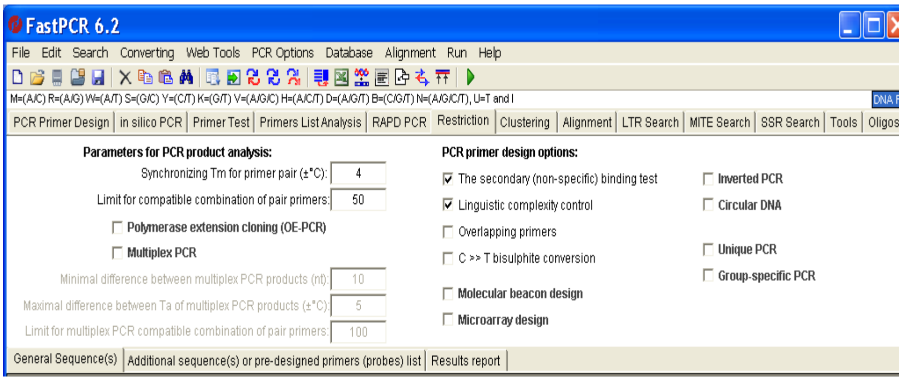
```
                   AvaII
T.scotti       0   GGTCCGCCTGCCCTGTGACTATTAGTTTAACGGCCGCGGTATTTTGACCGCGCGAAGGTAGCGCAA
                   AvaII
T.hanson       0   GGTCCGCCTGCCCTGTGACTATTAGTTTAACGGCCGCGGTATTTTGACCGCGCGAAGGTAGCGCAA
                   AvaII
P.scotti       0   GGTCCGCCTGCCCTGTGACTATTAGTTTAACGGCCGCGGTATTTTGACCGCGCGAAGGTAGCGCAA
                                     AvaII
P.brevicaud    0   GCAAATAAATGAATAAGAGGTCCCGCCTGCCCTGTGACTATAAGTTTAACGGCCGCGGTATTTTGA
P.antarcticu   0   GCGGTATTTTGACCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAATGGAGACCTGTATGAATGG
L.squamifron   0   GACTATTAGTTTAACGGCCGCGGTATTTTGACCGCGCGAAGGTAGCGCAATCACTTGTCTCTTAAA
I.cyan         0   CCGCGGTATTTTGACCGCGCGAAGGTAGCACAATCACTTGTCTCTTAAATGGAGACCTGTATGAAT
H.antarcticu   0   GTGACTATAAGTTTAACGGCCGCGGTATTTTGACCGCGCGAAGGTAGCGCAATCACTTGTCTTTTA
                   AvaII
G.gibberifron  0   GGTCCGCCTGCCCTGTGACTATTAGTTTAACGGCCGCGGTATTTTGACCGCGCGAAGGTAGCGCAA
```

Figure 6.Restriction Map analysis of variable sequences of different Bacterial genomes using Cleaver software.

### 6.4 FastPCR

The FastPCRis an integrated tool for PCR primers or probe design, *in silico*PCR, oligonucleotide assembly and analyses, alignment and repeat searching (Figure 7)**.** The software utilizes combinations of normal and degenerated primers for all tools and for the melting temperature calculation are based on the nearest neighbour thermodynamic parameters.The "*in silico*" (virtual) PCR primers or probe searching or *in silico* PCR against whole genome(s) or a list of chromosome - prediction of probable PCR products and search of potential mismatching location of the specified primers or probes. Comprehensive primer test, the melting temperature calculation for standard and degenerate oligonucleotides, primer PCR efficiency, primer's linguistic complexity, and dilution and resuspension calculator. Primers (probes) are analyzed for all primer secondary structures including G-quadruplexes detection, hairpins, self-dimers and cross-dimers in primer pairs. FastPCR has the capacity to handle long sequences and sets of nucleic acid or protein sequences and it allowed the individual task and parameters for each given sequences and joining several different tasks for single run. It also allows sequence editing and databases analysis. Efficient and complete detection of various types of repeats developed (for DNA based signature) and applied to the program with a visualisation.
The program includes various bioinformatics tools for analysis of sequences with GC or AT

skew, CG content and purine-pyrimidine skew, the linguistic sequence complexity; generation random DNA sequence, restriction analysis and supports the clustering of sequences and consensus sequence generation and sequences similarity and conservancy analysis.


FastPCR 6.2
File Edit Search Converting Web Tools PCR Options Database Alignment Run Help
DNA F
PCR Primer Design | in silico PCR | Primer Test | Primers List Analysis | RAPD PCR | Restriction | Clustering | Alignment | LTR Search | MITE Search | SSR Search | Tools | Oligos
General Sequence(s) | Additional sequence(s) or pre-designed primers (probes) list | Results report

M=(A/C) R=(A/G) W=(A/T) S=(G/C) Y=(C/T) K=(G/T) V=(A/G/C) H=(A/C/T) D=(A/G/T) B=(C/G/T) N=(A/G/C/T), U=T and I

Parameters for PCR product analysis:

Synchronizing Tm for primer pair (±°C): 4
Limit for compatible combination of pair primers: 50
□ Polymerase extension cloning (OE-PCR)
□ Multiplex PCR
Minimal difference between multiplex PCR products (nt): 10
Maximal difference between Ta of multiplex PCR products (±°C): 5
Limit for multiplex PCR compatible combination of pair primers: 100

PCR primer design options:

☑ The secondary (non-specific) binding test
☑ Linguistic complexity control
□ Overlapping primers
□ C >> T bisulphite conversion
□ Molecular beacon design
□ Microarray design
□ Inverted PCR
□ Circular DNA
□ Unique PCR
□ Group-specific PCR

Figure 7.Main Window of FastPCR software.

For SSR search or any other analysis just we need to prepare data file in notepad file and import in the main window. As per our need we can import the data and analyse by clicking on Run/SSR search/Primer list analysis etc. option looking in main window.

## References

Bustamante, CD., Nielsen, R. and Hartl, DL.(2003). Maximum likelihood and Bayesian methods for estimating the distribution of selective effects among classes of mutations using DNA polymorphism data.*Theoretical Population Biology*.**63**: 91-103.

Corander, J.,Waldmann, P. and Sillanpaa, MJ.(2003). Bayesian analysis of genetic differentiation between populations.*Genetics*.**163**: 367-374.

Goldstein, DB., Linares, AR.,Cavalli-Sforza, LL. and Feldman, MW. (1995). Genetic absolute dating based on microsatellites an origin of modern humans. *PNAS USA*.**92**: 6723-6727.

Hebert, PDN., Penton, EH., Burns, JM., Janzen, DH. AndHallwachs, W. (2004a). Ten Species in One: DNA Barcoding Reveals Cryptic Species in the Neotropical Skipper Butterfly Astraptesfulgerator. *Proc. Natl. Acad. Sci. USA***101(41)**: 14812-14817.

Hebert, PDN.,Stoeckle, MY.,Zemlak, TS. and Francis, CM. (2004b). Identification of Birds Through DNA Barcodes. *PLoS Biol*. **2(10)**: 1657-1663.

Jarman.(2006). Cleaver: software for identifying taxon specific restrictionendonuclease recognition sites. Bioinformatics Advance Access (http://bioinformatics.oxfordjournals.org/content/early/2006/06/20/bioinformatics.btl330.full.pdf.)

Kress, WJ.,Wurdack, KJ., Zimmer, EA.,Weigt, LA. and Janzen, DH. (2005). Use of DNA Barcodes to Identify Flowering Plants.*Proc. Nat. Acad. Sci. USA*,**102(23):** 8369-8374.

Paetkau, D., Calvert, W., Stirling, I. and Strobeck, C. (1995). Microsatellite analysis of population structure in Canadian polar bears.*Molecular Ecology*.**4**: 347-354.

Rannala, B. and Mountain, JL. (1997). Detecting immigration by using multi locus genotypes *PNAS, USA*. **94**: 9197-9221.

Sasazaki S., Hosokawa D., Ishihara R., Aihara H., Oyama K., Mannen, H. (2011).Development of discrimination markers between Japanese domestic and imported beef.*Animal Science Journal*,**82(1):**67-72.

Suekawa, Y., Aihara, H., Araki, M., Hosokawa, D., Mannen, H., Sasazaki, S. (2010). Development of breed identification markers based on a bovine 50K SNP array .*Meat Science*,**85(2),** 285–288.

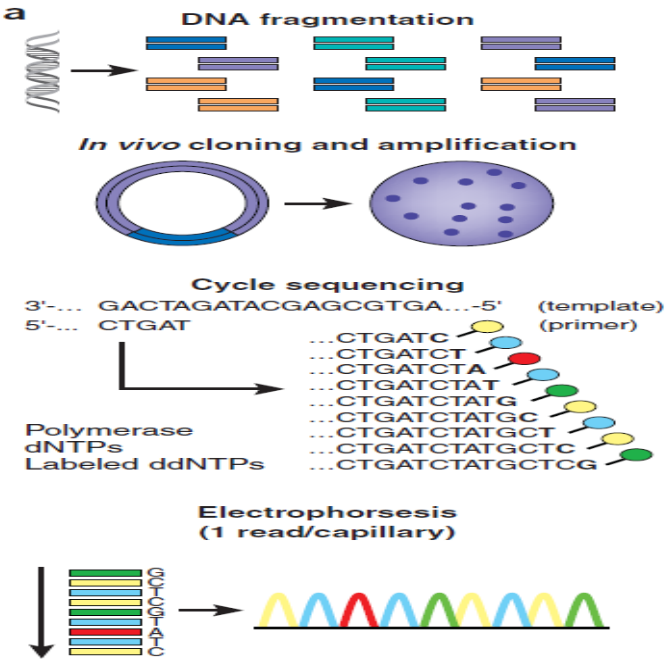
# Genome Assembly

**Dwijesh Chandra Mishra, Sanjeev Kumar, Sudhir Srivastava and Neeraj Budhlakoti**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Sanger Sequencing

- DNA is fragmented
- Cloned to a plasmid vector
- Cyclic sequencing reaction
- Separation by electrophoresis
- Readout with fluorescent tags



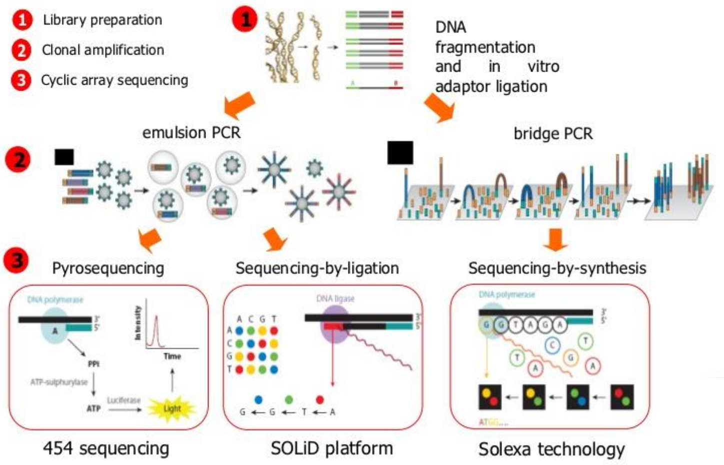
## Sanger Vs NGS

- ‘Sanger sequencing’ has been the only DNA sequencing method for 30 years but…
- …hunger for even greater sequencing throughput and more economical sequencing technology…
- NGS has the ability to process millions of sequence reads in parallel rather than 96 at a time (1/6 of the cost)

**NGS Platforms:** Different sequencing techniques used for next generation sequencing are:

- Roche/454 FLX: 2004
- Illumina Solexa Genome Analyzer: 2006
- Applied Biosystems SOLiD$^{TM}$ System: 2007
- Helicos Heliscope$^{TM}$ : 2009
- Pacific Biosciencies SMRT: 2010

## General Experimental Procedure

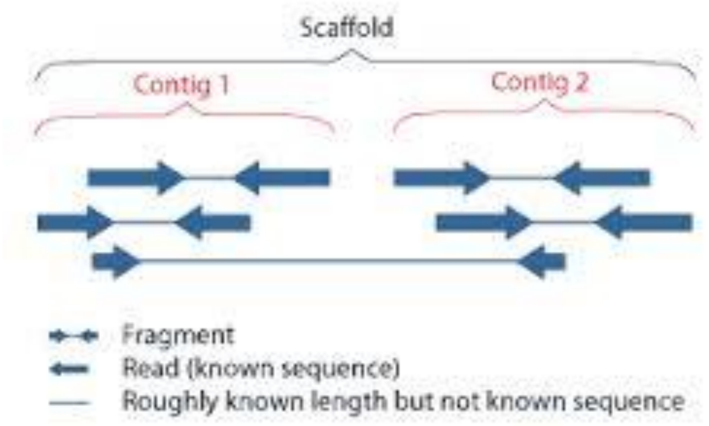


## Sequencing Technology at a Glance

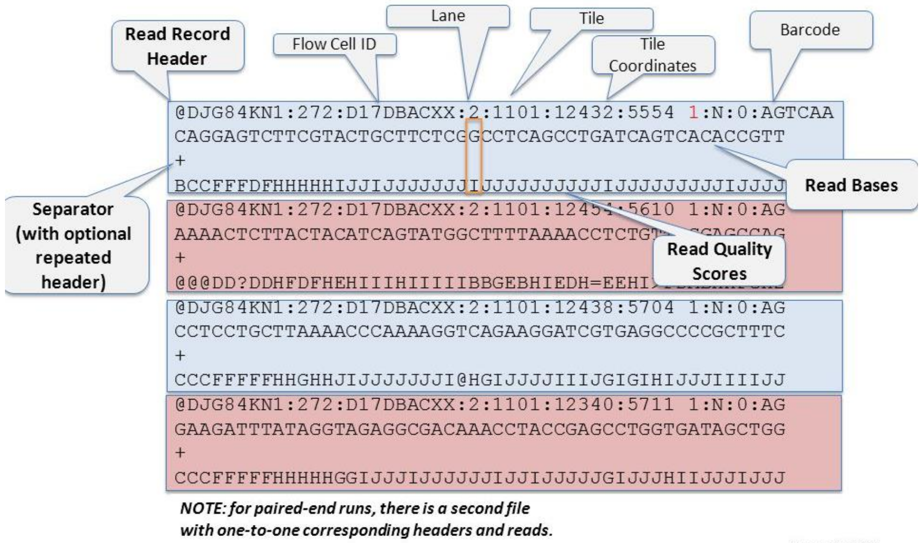
| Method | Read length | Accuracy | Time per run | Cost per 1 million bases | Advantages | Disadvantages |
|---|---|---|---|---|---|---|
| Chain termination (Sanger sequencing) | 400 to 900 bp | 99.9% | 20 minutes to 3 hours | Rs 144000 | Long individual reads. Useful for many applications. | More expensive and impractical for larger sequencing projects. |
| Pyrosequencing (454) | 700 bp | 99.9% | 24 hours | Rs 600 | Long read size. Fast | Runs are expensive. Homopolymer errors. |
| Sequencing by synthesis (Illumina) | 50 to 300 bp | 98% | 1 to 10 days, depending upon sequencer and specified read length | Rs 3 to 9 | Potential for high sequence yield, depending upon sequencer model and desired application. | Equipment can be very expensive. Requires high concentrations of DNA. |
| Sequencing by ligation (SOLiD sequencing) | 50+35 or 50+50 bp | 99.9% | 1 to 2 weeks | Rs 78 | Low cost per base. | Slower than other methods. Have issue sequencing palindromic sequence. |
| Single-molecule real-time sequencing (Pacific Bio) | 10,000 bp to 15,000 bp avg. (14,000 bp); | 87% | 30 minutes to 4 hours | Rs 7.8–36 | Longest read length. Fast. | Moderate throughput. Equipment can be very expensive. |

## Reads, Contigs and Scaffolds

- Reads are what you start with (35bp-800bp)
- Fragmented assemblies produce contigs that can be kilobases in length
- Putting contigs together into scaffolds is the next step



## FASTQ Format



```
@DJG84KN1:272:D17DBACXX:2:1101:12432:5554 1:N:0:AGTCAA
CAGGAGTCTTCGTACTGCTTCTCGGCCTCAGCCTGATCAGTCACACCGTT
+
BCCFFFDFHHHHHIJJIJJJJJJJIJJJJJJJJJIJJJJJJJJJIJJJJ
@DJG84KN1:272:D17DBACXX:2:1101:1245[illegible]:5610 1:N:0:AG
AAAACTCTTACTACATCAGTATGGCTTTTAAAACCTCTGT[illegible]CGAGCCAG
+
@@@DD?DDHFDFHEHIIIHIIIIIBBGEBHIEDH=EEHI>[illegible]
@DJG84KN1:272:D17DBACXX:2:1101:12438:5704 1:N:0:AG
CCTCCTGCTTAAAACCCAAAAGGTCAGAAGGATCGTGAGGCCCCGCTTTC
+
CCCFFFFFHHGHHJIJJJJJJJI@HGIJJJJIIIJGIGIHIJJJIIIIJJ
@DJG84KN1:272:D17DBACXX:2:1101:12340:5711 1:N:0:AG
GAAGATTTATAGGTAGAGGCGACAAACCTACCGAGCCTGGTGATAGCTGG
+
CCCFFFFFHHHHHGGIJJJIJJJJJJIJJIJJJJJGIJJJHIIJJJIJJJ
```

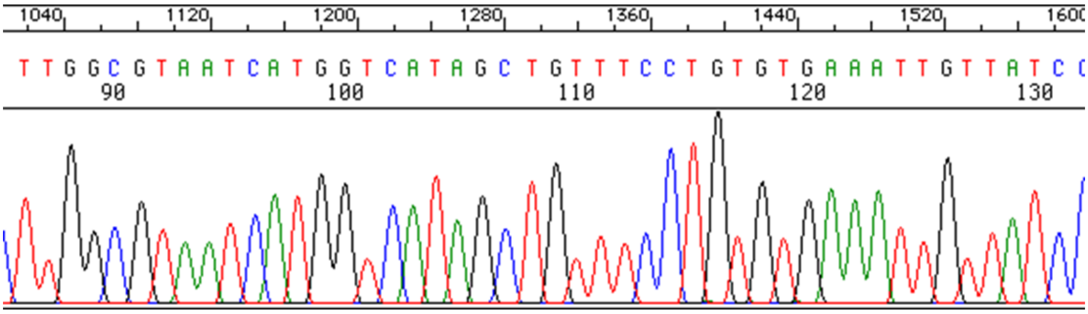
*NOTE: for paired-end runs, there is a second file with one-to-one corresponding headers and reads.*

(Passarelli, 2012)

## Before Assembly

### <u>Fragment readout</u>

- DNA characters in a fragment are determined from chromatogram
- Base call is a DNA character that is determined from chromatogram



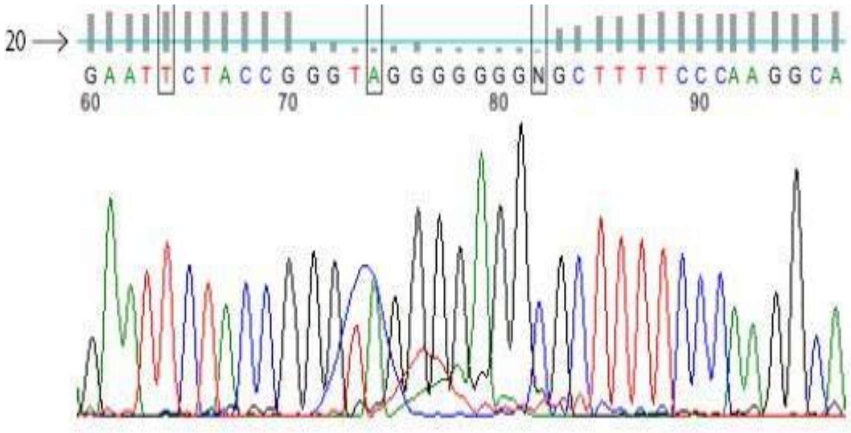
## Fragment readout

- Phred Score- determine the quality value of a base

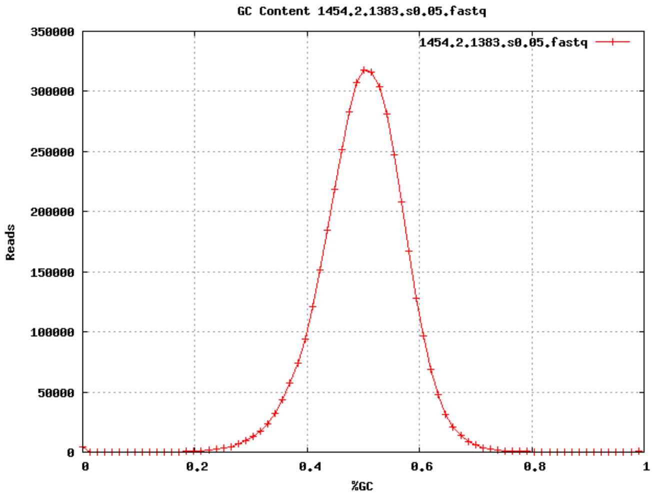
$$q = -10 \times log_{10}(\text{p})$$

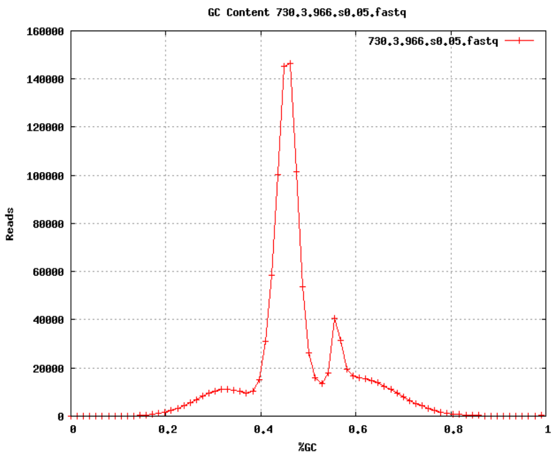
  where p is the estimated error probability for the base

- if Phred assigns a quality score of 30 to a base, the chances that this base is called incorrectly are 1 in 1000
- The most commonly used method is to count the bases with a quality score of 20 and above
- Phred Score

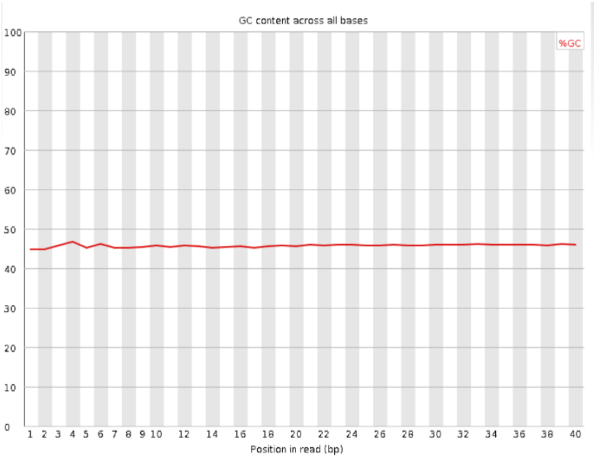


## Genome Properties

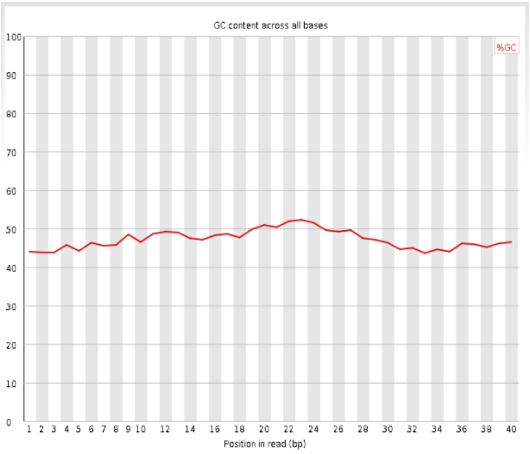




**PASS** **FAIL**

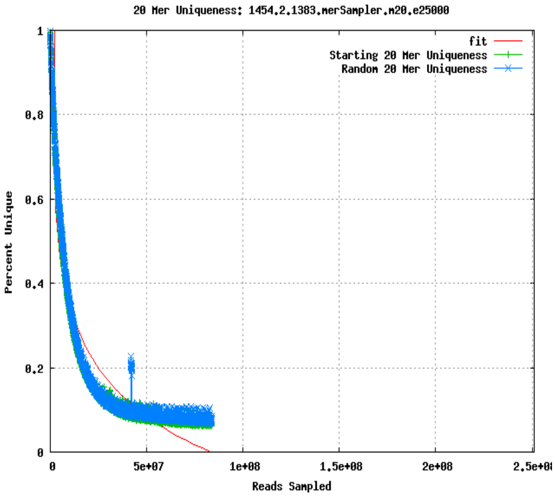


**PASS**

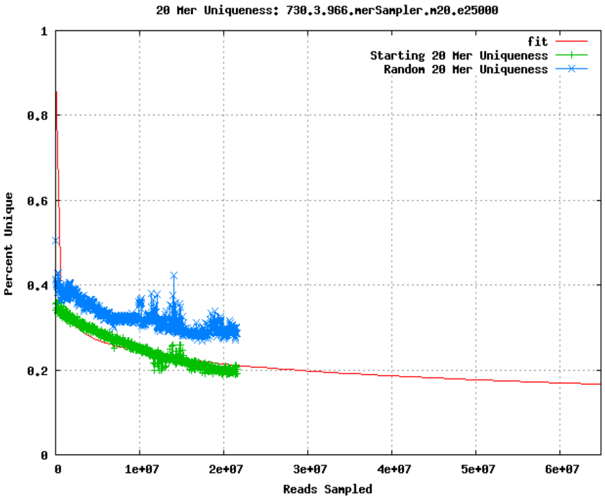


**FAIL**

## Library Quality

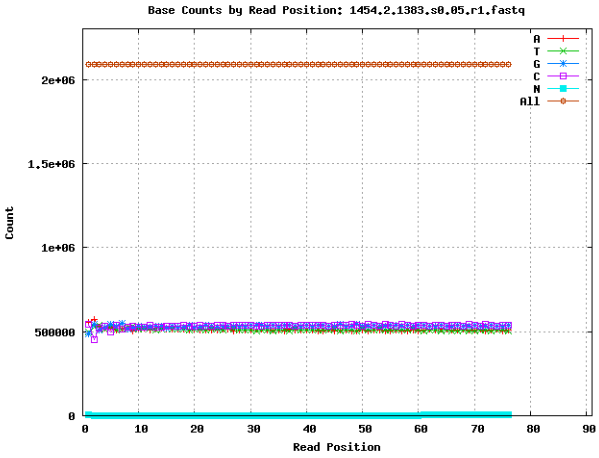


**PASS**

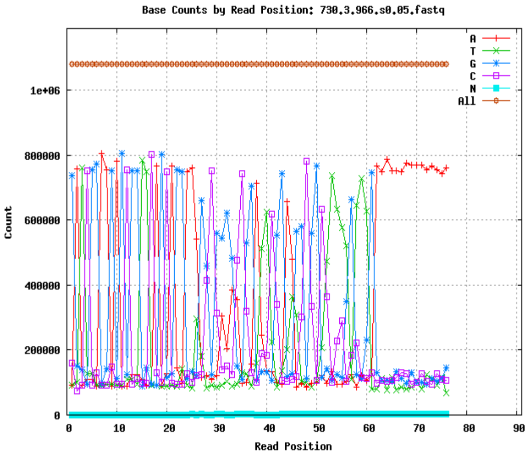


**FAIL**

## Run Quality

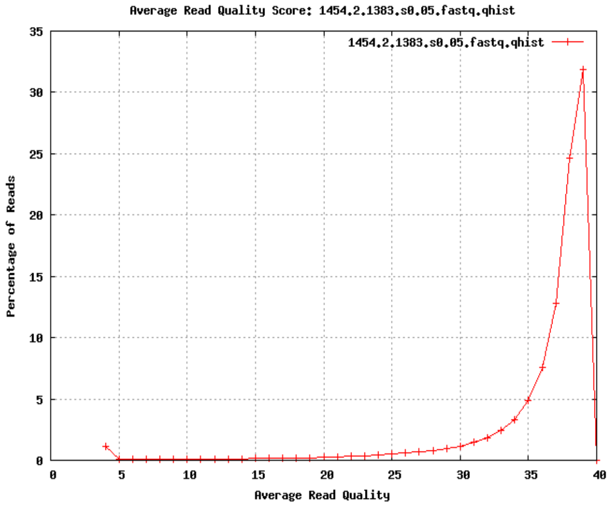


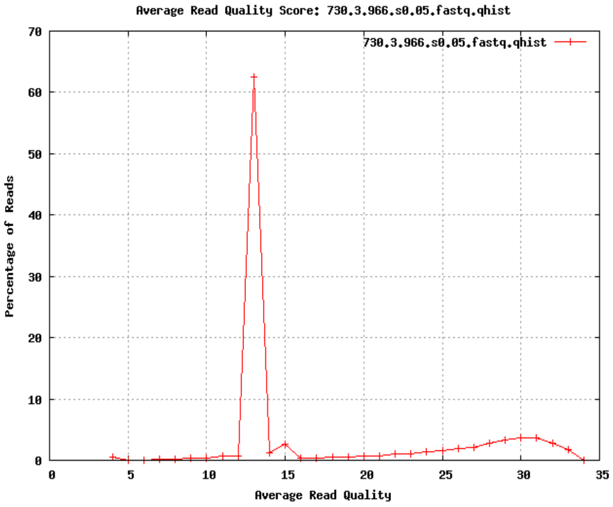
**PASS**



**FAIL**

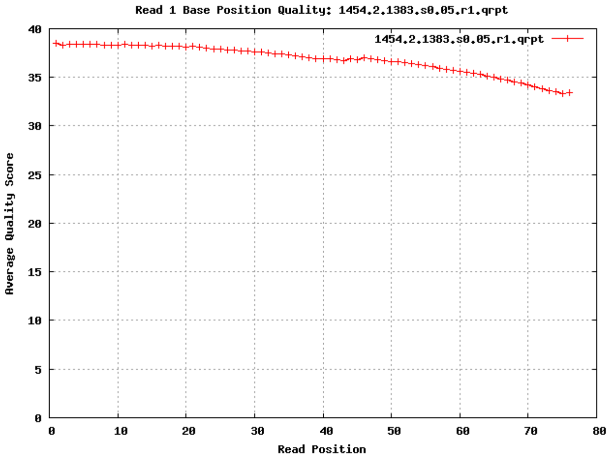
## Read Quality

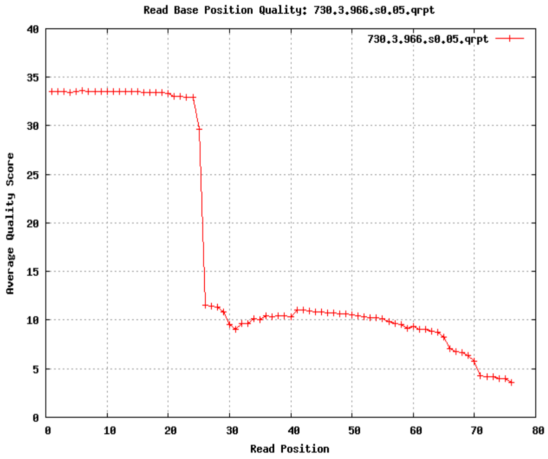




**PASS**

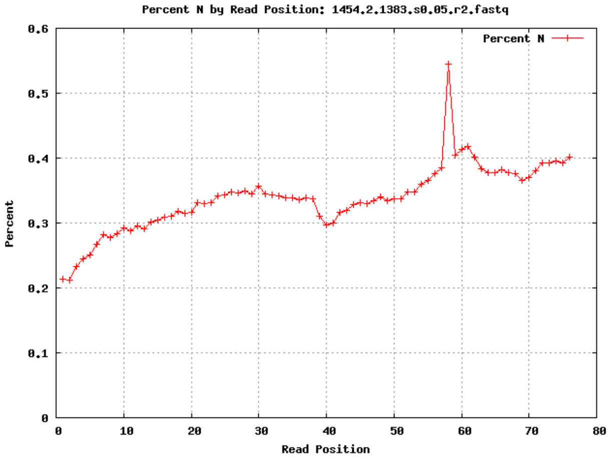
**FAIL**

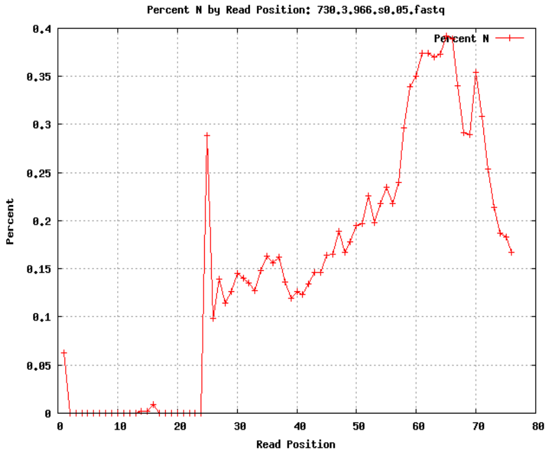




**PASS**

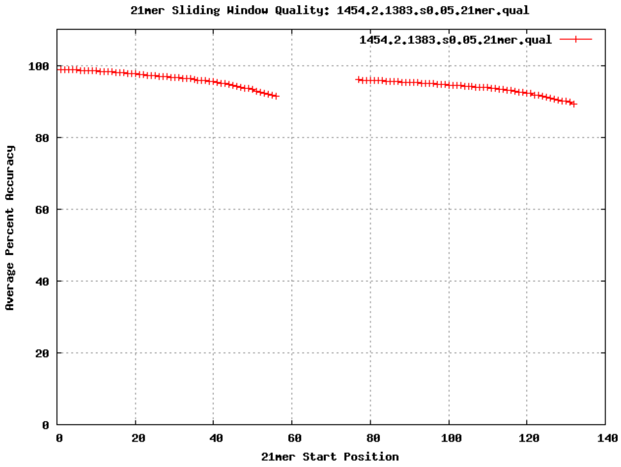
**FAIL**



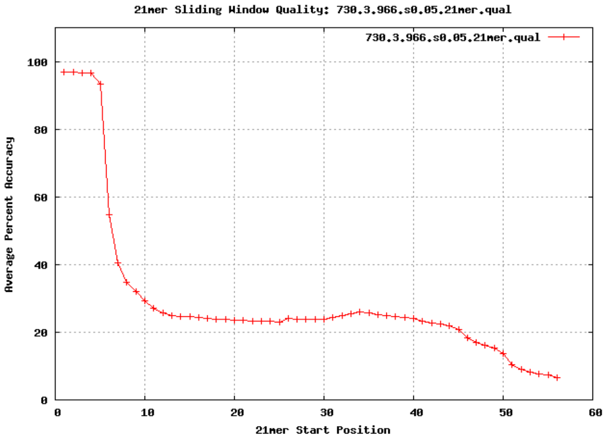


**PASS**

**FAIL**



PASS



FAIL

## Trimming

- Trimming low-quality sequences

  -removal of reads containing poor quality base calls

- Trimming vector sequences

  -removal of reads containing vector sequences

# Genome Annotation

**Sanjeev Kumar, D. C. Mishra, Sneha Murmu and Jyotika Bhati**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

Until the genome revolution, genes were identified by researchers with specific interests in a particular protein or cellular process. Once identified, these genes were isolated, typically by cloning and sequencing cDNAs, usually followed by targeted sequencing of the longer genomics segments that code for the cDNAs. Once an organism's entire genome sequence becomes available, there is strong motivation for finding all the genes encoded by a genome at once rather than in a piecemeal approach. Such catalogue is immensely valuable to researchers, as they can learn much more from the whole picture than from a much more limited set of genes. For example, genes of similar sequence can be identified, evolutionary and functional relationships can be elucidated, and a global picture of how many and what types of genes are present in a genome can be seen. A significant portion of the effort in genome sequencing is devoted to the process of *annotation*, in which genes, regulatory elements, and other features of the sequence are identifies as thoroughly as possible and catalogued in a standard format in public databases so that researchers can easily use the information. Functional genomics research has expanded enormously in the last decade and particularly the plant biology research community. Functional annotation of novel DNA sequences is probably one of the top requirements in functional genomics as this holds, to a great extent, the key to the biological interpretation of experimental results.

## Computational Gene Prediction

Computational gene prediction is becoming more and more essential for the automatic analysis and annotation of large uncharacterized genomic sequences. In the past two decades, many algorithms have been evolved to predict protein coding regions of the DNA sequences. They all have in common, to varying degree, the ability to differentiate between gene features like Exons, Introns, Splicing sites, Regulatory sites etc. Gene prediction methods predicts coding region in the query sequences and then annotates the sequences databases.

## Gene Structure and Expression

The gene structure and the gene expression mechanism in eukaryotes are far more complicated than in prokaryotes. In typical eukaryotes, the region of the DNA coding for a protein is usually not continuous. This region is composed of alternating stretches of *exons* and *introns*. During transcription, both exons and introns are transcribed onto the RNA, in their linear order. Thereafter, a process called *splicing* takes place, in which, the intron

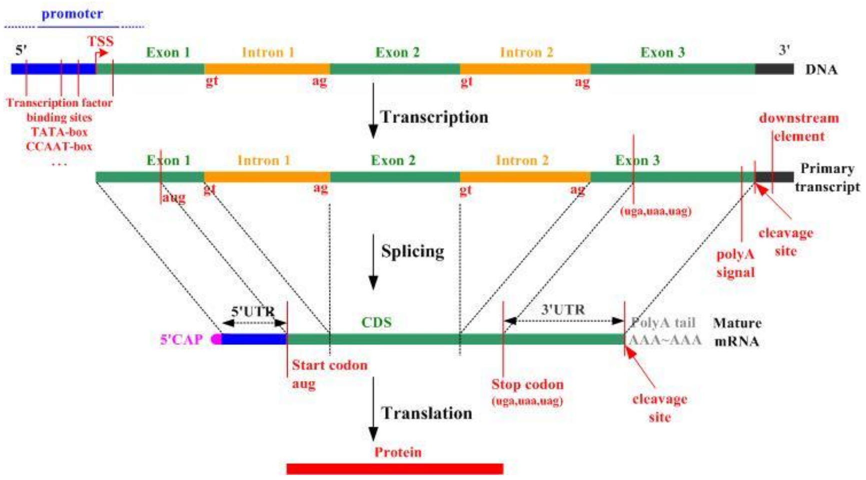


**Fig. 1: Representative Diagram of Protein Coding Eukaryotic Gene**

sequences are excised and discarded from the RNA sequence. The remaining RNA segments, the ones corresponding to the exons are ligated to form the mature RNA strand. A typical multi-exon gene has the following structure (as illustrated in Fig. 1). It starts with the promoter region, which is followed by a transcribed but non-coding region called *5' untranslated region (5' UTR)*. Then follows the initial exon which contains the start codon. Following the initial exon, there is an alternating series of introns and internal exons, followed by the terminating exon, which contains the stop codon. It is followed by another non-coding region called the *3' UTR*. Ending the eukaryotic gene, there is a polyadenylation (polyA) signal: the nucleotide Adenine repeating several times. The exon-intron boundaries (i.e., the splice sites) are signalled by specific short (2bp long) sequences. The 5'(3') end of an intron (exon) is called the *donor* site, and the 3'(5') end of an intron (exon) is called the *acceptor* site. The problem of gene identification is complicated in the case of eukaryotes by the vast variation that is found in gene structure.

## Gene Prediction Methods

There are mainly two classes of methods for computational gene prediction (Fig. 2). One is based on sequence similarity searches, while the other is gene structure and signal-based searches, which is also referred to as Ab initio gene finding.

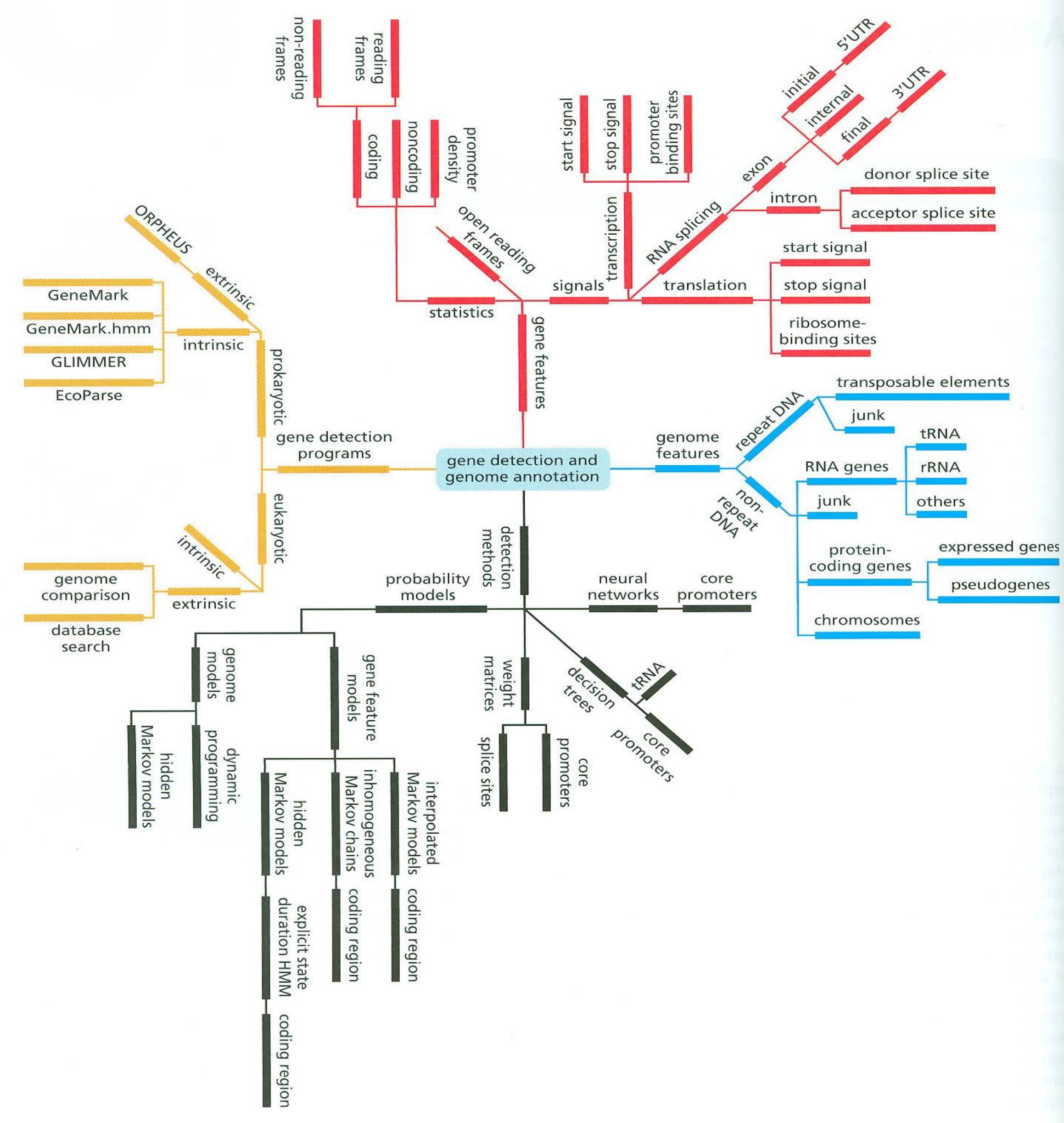
### Sequence Similarity Searches

Sequence similarity search is a conceptually simple approach that is based on finding similarity in gene sequences between ESTs (expressed sequence tags), proteins, or other genomes to the input genome. This approach is based on the assumption that functional regions (exons) are more conserved evolutionarily than non-functional regions (intergenic or intronic regions). Once there is similarity between a certain genomic region and an EST, DNA, or protein, the similarity information can be used to infer gene structure or function of that region. EST-based sequence similarity usually has drawbacks in that ESTs only correspond to small portions of the gene sequence, which means that it is often difficult to predict the complete gene structure of a given region. Local alignment and global alignment are two methods based on similarity searches. The most common local alignment tool is the BLAST family of programs, which detects sequence similarity to known genes, proteins, or ESTs. The biggest limitation to this type of approaches is that only about half of the genes being discovered have significant homology to genes in the databases.

**Ab initio Gene Prediction Methods**

The second class of methods for the computational identification of genes is to use gene structure as a template to detect genes, which is also called *ab initio* prediction. *Ab initio* gene predictions rely on two types of sequence information: signal sensors and content sensors. Signal sensors refer to short sequence motifs, such as splice sites, branch points, poly pyrimidine tracts, start codons and stop codons. Exon detection must rely on the content sensors, which refer to the patterns of codon usage that are unique to a species, and allow coding sequences to be distinguished from the surrounding non-coding sequences by statistical detection algorithms.

Many algorithms are applied for modeling gene structure, such as Dynamic Programming, linear discriminant analysis, Linguist methods, Hidden Markov Model and Neural Network.



Based on these models, a great number of *ab initio* gene prediction programs have been developed.

**Fig. 2: Diagrammatic Representation of Gene Prediction and Annotation**

## Gene Discovery in Prokaryotic Genomes

Discovery of genes in Prokaryote is relatively easy, due to the higher gene density typical of prokaryotes and the absence of introns in their protein coding regions. DNA sequences that encode proteins are transcribed into mRNA, and the mRNA is usually translated into proteins without significant modification. The longest ORFs (open reading frames) running from the first available start codon on the mRNA to the next stop codon in the same reading frame generally provide a good, but not assured prediction of the protein coding regions. Several methods have been devised that use different types of Markov models in order to capture the compositional differences among coding regions, “shadow" coding regions (coding on the opposite DNA strand), and noncoding DNA. Such methods, including ECOPARSE, the widely used GENMARK, and Glimmer program, appear to be able to identify most protein coding genes with good performance (Fig. 3).

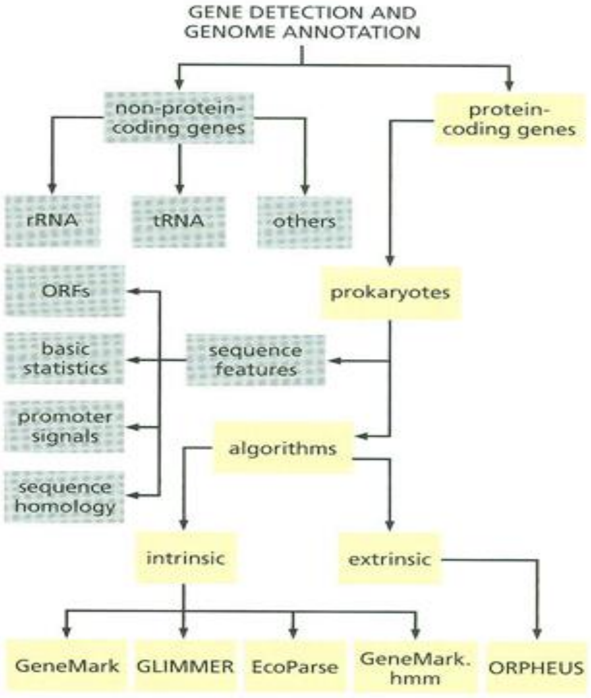


**Fig. 3: Flow Diagram of Prokaryotic Gene Discovery**

## Gene Discovery in Eukaryotic Genome

It is a quite different problem from that encountered in prokaryotes. Transcription of protein coding regions initiated at specific promoter sequences is followed by removal of noncoding sequences (introns) from pre-mRNA by a splicing mechanism, leaving the protein encoding exons. Once the introns have been removed and certain other modifications to the mature RNA have been made, the resulting mature mRNA can be translated in the 5` to 3` direction, usually from the first start codon to the first stop codon. As a result of the presence of intron sequences in the genomic DNA sequences of eukaryotes, the ORF corresponding to an encoded gene will be interrupted by the presence of introns that usually generate stop codons (Fig.4).

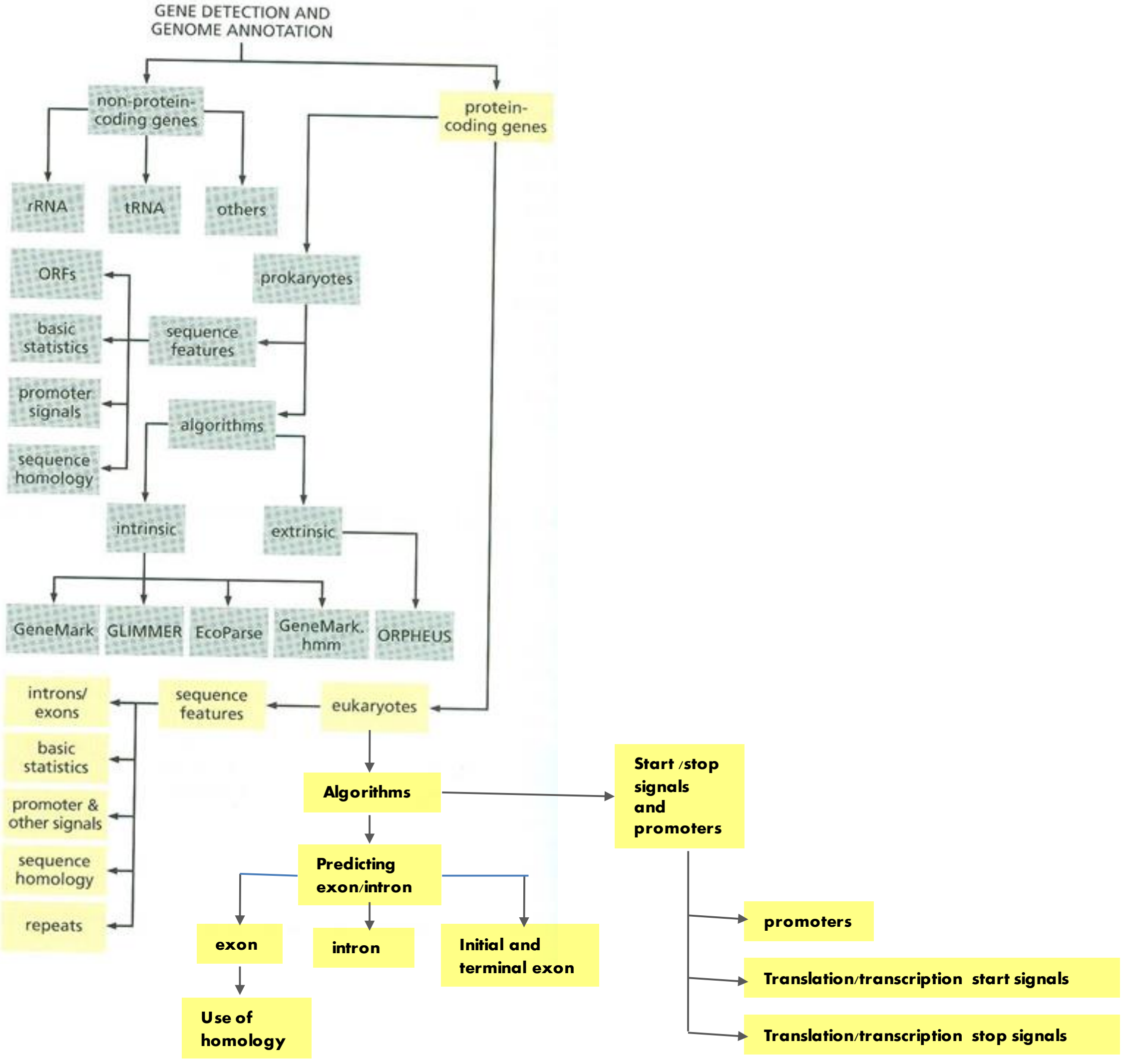


**Fig. 4: Flow Diagram of Eukaryotic Gene Discovery**

## Gene Prediction Program

There are two basic problems in gene prediction: prediction of protein coding regions and prediction of the functional sites of genes. Gene prediction program can be classified into four generations. The first generation of programs was designed to identify approximate locations of coding regions in genomic DNA. The most widely known programs were probably TestCode and GRAIL. But they could not accurately predict precise exon locations. The second generation, such as SORFIND and Xpound, combined splice signal and coding region identification to predict potential exons, but did not attempt to assemble predicted exons into complete genes. The next generation of programs attempted the more difficult task of predicting complete gene structures. A variety of programs have been developed, including GeneID, GeneParser, GenLang, and FGENEH. However, the performance of those programs remained rather poor. Moreover, those programs were all based on the assumption that the input sequence contains exactly one complete gene, which is not often the case. To solve this problem and improve accuracy and applicability further, GENSCAN and AUGUSTUS were developed, which could be classified into the fourth generation.

### GeneMark

GeneMark uses a Markov Chain model to represent the statistics of the coding and noncoding frames. The method uses the dicodon statistics to identify coding regions. Consider the analysis of a sequence x whose base at the ith position is called $x_i$. The Markov chains used are fifth order, and consist of a terms such as P(a/$x_1x_2x_3x_4x_5$), which represent the probability of the sixth base of the sequence x being given a given that the previous five bases in the sequence x where $x_1x_2x_3x_4x_{5,}$ resulting in the first dicodon of the sequence being $x_1x_2x_3x_4x_5$a. These terms must be defined for all possible pentamers with the general sequence $b_1b_2b_3b_4b_5$. The values of these terms can be obtained of analysis of data, consisting of nucleotide sequence in which the coding regions have been actually identified. When there are sufficient data, they are given by

$$P(\frac{a}{b_1b_2b_3b_4b_5}) = \frac{n_{b_{1b_2b_3}b_4b_5a}}{\sum_{a=A,C,G,T} n_{b_{1b_2b_3}b_4b_5a}}$$

where, $n_{b_{1b_2b_3}b_4b_5a}$ is the number of times the sequence $b_1b_2b_3b_4b_5$a occurs in the training data. This is the maximum likelihood estimators of the probability from the training data.

### Glimmer

The core of Glimmer is Interpolated Markov Model (IMM), which can be described as a generalized Markov chain with variable order. After GeneMark introduces the fixed-order Markov chains, Glimmer attempts to find a better approach for modeling the genome content. The motivational fact is that the bigger the order of the Markov chain, the more non-randomness can be described. However, as we move to higher order models, the number of probabilities that we must estimate from the data increases exponentially. The major limitation of the fixed-order Markov chain is that models from higher order require exponentially more training data, which are limited and usually not available for new sequences. However, there are some oligomers from higher order that occur often enough to be extremely useful predictors. For the purpose of using these higher-order statistics, whenever sufficient data is available, Glimmer IMMs.

Glimmer calculates the probabilities for all Markov chains from $0^{th}$ order to $8^{th}$. If there are longer sequences (e.g. 8-mers) occurring frequently, IMM makes use of them even when there is insufficient data to train an 8-th order model. Similarly, when the statistics from the 8-th order model do not provide significant information, Glimmer refers to the lower-order models to predict genes.

Opposed to the supervised GeneMark, Glimmer uses the input sequence for training. The ORFs longer than a certain threshold are detected and used for training, because there is high probability that they are genes in prokaryotes. Another training option is to use the sequences with homology to known genes from other organisms, available in public databases. Moreover, the user can decide whether to use long ORFs for training purposes or choose any set of genes to train and build the IMM.

### GeneMark.hmm

GeneMark.hmm is designed to improve GeneMark in finding exact gene starts. Therefore, the properties of GeneMark.hmm are complementary to GeneMark. GeneMark.hmm uses GeneMark models of coding and non-coding regions and incorporates them into hidden Markov model framework. In short terms, Hidden Markov Models (HMM) are used to describe the transitions from non-coding to coding regions and vice versa. GeneMark.hmm predicts the most likely structure of the genome using the Viterbi algorithm, a dynamic programming algorithm for finding the most likely sequence of hidden states. To further improve the prediction of translation start position, GeneMark.hmm derives a model of the ribosome binding site (6-7 nucleotides preceding the start codon, which are bound by the ribosome when initiating protein translation). This model is used for refinement of the results.

Both GeneMark and GeneMark.hmm detect prokaryotic genes in terms of identifying open reading frames that contain real genes. Moreover, they both use pre-computed species-specific gene models as training data, in order to determine the parameters of the protein-coding and non-coding regions.

### Orpheus

The ORPHEUS program uses homology, codon statistics and ribosome binding sites to improve the methods presented so far by using information that those programs ignored. One of the key differences is that it uses database searches to help determine putative genes, and is thus an extrinsic method. This initial set of genes is used to define the coding statistics for the organism, in this case working at the level of codon, not dicodons. These statistics are then used to define a larger set of candidate ORFs. From this set, those ORFs with an unambiguous start codon end are used to define a scoring matrix for the ribosome-binding site, which is then used to determine the 5` end of those ORFs where alternative start are present.

### EcoParse

EcoParse is one of the first HMM model based gene finder, was developed for gene finding in *E.coli*. It focuses on the uses the codon structure of genes. With EcoParse a flora of HMM based gene finder, usuing dynamic programming and the viterbi algorithm to parse a sequence, emerged.

### Evaluation of Gene Prediction Programs

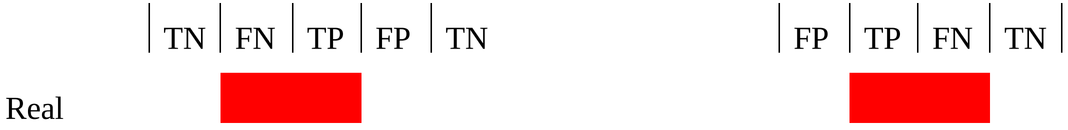
In the field of gene prediction accuracy can be measured at three levels

a. Coding nucleotides (base level)

b. Exon structure (exon level)

c. Protein product (protein level)

At base level gene predictions can be evaluated in terms of *true positives (TP)* (predicted features that are real), *true negatives* (TN) (non-predicted features that are not real), *false positives (FP)* (predicted features that are not real), and *false negatives (FN)* (real features that were not predicted) Fig. 5. Usually the base assignment is to be in a coding or non coding segment, but this analysis can be extended to include non coding parts of genes, or any functional parts of the sequences.



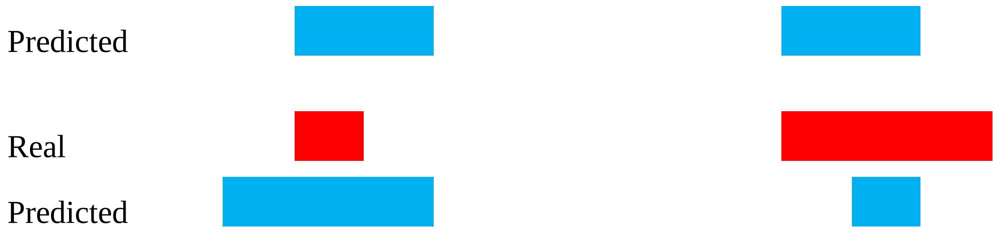


**Fig. 5: Four Possible Comparisons of Real and Predicted Genes**

Sensitivity (Sn): The fraction of bases in real genes that are correctly predicted to be in genes is the sensitivity and interpreted as the probability of correctly predicting a nucleotide to be in a given gene that it actually is.

$$Sn = \frac{TP}{TP + FN}$$

Specificity (Sp): The fraction of those bases which are predicted to be in genes that actually are is called the specificity and interpreted as the probability of a nucleotide actually being in a gene given that it has been predicted to be.

$$Sp = \frac{TP}{TP + FP}$$

Care has to be taken in using these two values to assess a gene prediction program because, as with the normal definition of specificity, extreme results can make them misleading.

Approximate correlation coefficient (AC) has been proposed as a single measure to circumvent these difficulties. This defined as AC=2(ACP-0.5), where

$$ACP = \frac{1}{n}\left(\frac{TP}{TP+FN} + \frac{TP}{TP+FP} + \frac{TN}{TN+FP} + \frac{TN}{TN+FN}\right),$$

At the exon level, determination of prediction accuracy depends on the exact prediction of exon start and end points. There are two measures of sensitivity and specificity used in the field, each of which measures a different but useful property.

The sensitivity measures used are

$S_{n1}$ = CE/AE and Sn2 = ME/AE

The specificity measures used are

$S_{p1}$=CE/PE and $S_{p2}$=WE/PE

Where,

AE = No of actual exons in the data

PE = No of predicted exons in the data

CE = No of correct predicted exons

ME = No of missing exons (rarely occurs)

WE = No of wrongly predicted exons (Figure-5)

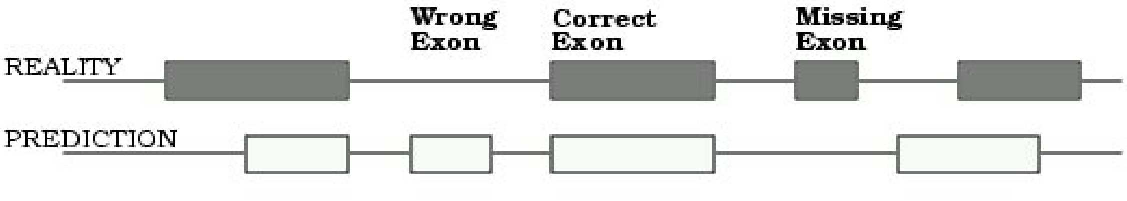


**Fig. 6: Real and Predicted Exons**

## Gene Ontology

The gene ontology (GO, http:www.geneontology.org) is probably the most extensive scheme today for the description of gene product functions but other systems such as enzyme codes, KEGG pathways, FunCat, or COG are also widely used. Here, we describe the Blast2GO (B2G, www.blast2go.org) application for the functional annotation, management, and data mining of novel sequence data through the use of common controlled vocabulary schemas. The main application domain of the tool is the functional genomics of nonmodel organisms and it is primarily intended to support research in experimental labs. Blast2GO strives to be the application of choice for the annotation of novel sequences in functional genomics projects where thousands of fragments need to be characterized. Functional annotation in Blast2GO is based on homology transfer. Within this framework, the actual annotation procedure is configurable and permits the design of different annotation strategies. Blast2GO annotation parameters include the choice of search database, the strength and number of blast results, the extension of the query-hit match, the quality of the transferred annotations, and the inclusion of motif annotation. Vocabularies supported by B2G are gene ontology terms, enzyme codes (EC), InterPro IDs, and KEGG pathways.

Fig.7 shows the basic components of the Blast2GO suite. Functional assignments proceed through an elaborate annotation procedure that comprises a central strategy plus refinement functions. Next, visualization and data mining engines permit exploiting the annotation results to gain functional knowledge. GO annotations are generated through a 3-step process: blast, mapping, annotation. InterPro terms are obtained from InterProScan at EBI, converted and merged to GOs. GO annotation can be modulated from Annex, GOSlim web services and manual editing. EC and KEGG annotations are generated from GO. Visual tools include sequence color code, KEGG pathways, and GO graphs with node highlighting and filtering options. Additional annotation data-mining tools include statistical charts and gene set enrichment analysis functions.

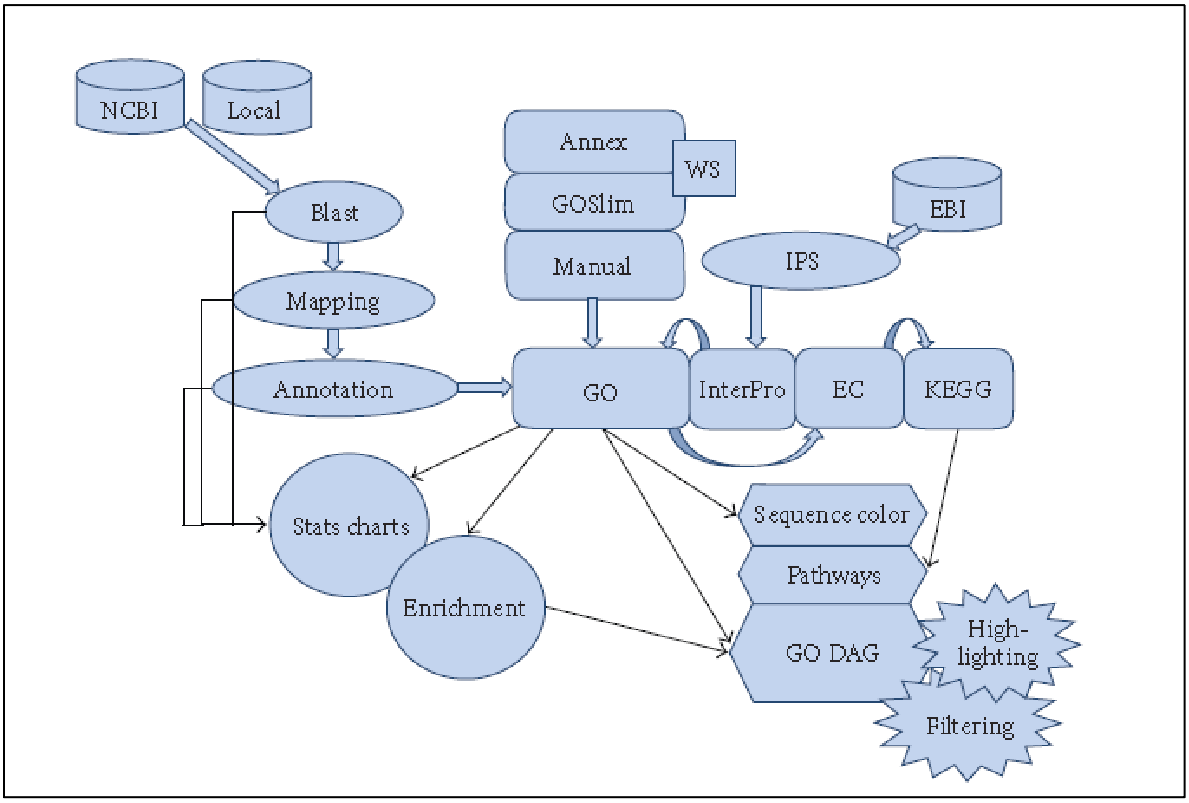


**Fig. 7: Schematic Representation of Blast2GO Application.**

The Blast2GO annotation procedure consists of three main steps: blast to find homologous sequences, mapping to collect GO terms associated to blast hits, and annotation to assign trustworthy information to query sequences.

## Blast Step

The first step in B2G is to find sequences similar to a query set by blast. B2G accepts nucleotide and protein sequences in FASTA format and supports the four basic blast programs (blastx, blastp, blastn, and tblastx). Homology searches can be launched against public databases such as (the) NCBI nr using a query-friendly version of blast (QBlast). This is the default option and in this case, no additional installations are needed. Alternatively, blast can be run locally against a proprietary FASTA-formatted database, which requires a working www-blast installation. The Make Filtered Blast-GO-BD function in the Tools menu allows the creation of customized databases containing only GO annotated entries, which can be used in combination with the local blast option. Other configurable parameters at the blast step are the expectation value (e-value) threshold, the number of retrieved hits, and the minimal alignment length (hsp length) which permits the exclusion of hits with short, low e-value matches from the sources of functional terms. Annotation, however, will ultimately be based on sequence similarity levels as similarity percentages are independent of database size and more intuitive than e-values. Blast2GO parses blast results and presents the information for each sequence in table format. Query sequence descriptions are obtained by applying a language processing algorithm to hit descriptions, which extracts informative names and avoids low content terms such as “hypothetical protein” or “expressed protein”.

## Mapping Step

Mapping is the process of retrieving GO terms associated to the hits obtained after a blast search. B2G performs three different mappings as follows.

a. Blast result accessions are used to retrieve gene names (symbols) making use of two mapping files provided by NCBI (geneinfo, gene2accession). Identified gene names are searched in the species-specific entries of the gene product table of the GO database.

b. Blast result GI identifiers are used to retrieve UniProt IDs making use of a mapping file from PIR (Non-redundant Reference Protein database) including PSD, UniProt, Swiss-Prot, TrEMBL, RefSeq, GenPept, and PDB.

c. Blast result accessions are searched directly in the DBXRef Table of the GO database.

### Annotation Step

This is the process of assigning functional terms to query sequences from the pool of GO terms gathered in the mapping step. Function assignment is based on the gene ontology vocabulary. Mapping from GO terms to enzyme codes permits the subsequent recovery of enzyme codes and KEGG pathway annotations. The B2G annotation algorithm takes into consideration the similarity between query and hit sequences, the quality of the source of GO assignments, and the structure of the GO DAG. For each query sequence and each candidate GO term, an annotation score (AS) is computed (see Figure 8). The AS is composed of two terms. The first, direct term (DT), represents the highest similarity value among the hit sequences bearing this GO term, weighted by a factor corresponding to its evidence code (EC). A GO term EC is present for every annotation in the GO database to indicate the procedure of functional assignment.

$$
\begin{aligned}
&DT = \max\,(\text{similarity} \times EC_{\text{weight}}) \\
&AT = (\#GO - 1) \times GO_{\text{weight}} \\
&AR\ :\ \text{lowest.node}(AS(DT + AT) \geq \text{threshold})
\end{aligned}
$$

**Fig. 8: Blast2GO Annotation Rule**

ECs vary from experimental evidence, such as inferred by direct assay (IDA) to unsupervised assignments such as inferred by electronic annotation (IEA). The second term (AT) of the annotation rule introduces the possibility of abstraction into the annotation algorithm. Abstraction is defined as the annotation to a parent node when several child nodes are present in the GO candidate pool. This term multiplies the number of total GOs unified at the node by a user defined factor or GO weight (GOw) that controls the possibility and strength of abstraction. When all ECw’s are set to 1 (no EC control) and the GOw is set to 0 (no abstraction is possible), the annotation score of a given GO term equals the highest similarity value among the blast hits annotated with that term. If the ECw is smaller than one, the DT decreases and higher query-hit similarities are required to surpass the annotation threshold. If the GOw is not equal to zero, the AT becomes contributing and the annotation of a parent node is possible if multiple child nodes coexist that do not reach the annotation cutoff. Default values of B2G annotation parameters were chosen to optimize the ratio between annotation coverage and annotation accuracy. Finally, the AR selects the lowest terms per branch that exceed a user-defined threshold.

Blast2GO includes different functionalities to complete and modify the annotations obtained through the above-defined procedure. Enzyme codes and KEGG pathway annotations are generated from the direct mapping of GO terms to their enzyme code equivalents. Additionally, Blast2GO offers InterPro searches directly from the B2G interface. B2G launches sequence queries in batch, and recovers, parses, and uploads InterPro results. Furthermore, InterPro IDs can be mapped to GO terms and merged with blast-derived GO annotations to provide one integrated annotation result. In this process, B2G ensures that only the lowest term per branch remains in the final annotation set, removing possible parent-child relationships originating from the merging action.

### References

1. Conesa, S. Gotz, J. M. Garcia-Gomez, J. Terol, M. Talon, and M. Robles, “Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research,” Bioinformatics, vol. 21, no. 18, pp. 3674–3676, 2005.
2. Conesa and S. Gotz, "Blast2GO: A Comprehensive Suite for Functional Analysis in Plant Genomics," International Journal of Plant Genomics, vol. 2008, 2008.
3. H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa, “KEGG: Kyoto Encyclopedia of Genes and Genomes,” Nucleic Acids Research, vol. 27, no. 1, pp. 29–34, 1999.
4. J.D. Watson, R.M. Myers, A.A. Caudy and J.A. Witkowski, "Recombinant DNA: Genes and Genomes - A Short Course," 3rd Ed., 2007.
5. M. Ashburner, C. A. Ball, J. A. Blake, et al., “Gene Ontology: tool for the unification of biology. The Gene Ontology Consortium,” Nature Genetics, vol. 25, no. 1, pp. 25–29, 2000.
6. Ruepp, A. Zollner, D. Maier, et al., “The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes,” Nucleic Acids Research, vol. 32, no. 18, pp. 5539–5545, 2004.
7. R. L. Tatusov, N. D. Fedorova, J. D. Jackson, et al., “The COG database: an updated version includes eukaryotes,” BMC Bioinformatics, vol. 4, p. 41, 2003.
8. Schomburg, A. Chang, C. Ebeling, et al., “BRENDA, the enzyme database: updates and major new developments,” Nucleic Acids Research, vol. 32, Database issue, pp. D431–D433, 2004.
9. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, “Basic local alignment search tool,” Journal of Molecular Biology, vol. 215, no. 3, pp. 403–410, 1990.
10. S. Myhre, H. Tveit, T. Mollestad, and A. Lægreid, “Additional Gene Ontology structure for improved biological reasoning,” Bioinformatics, vol. 22, no. 16, pp. 2020–2027, 2006.

# Hands-on Session for Genome Annotation

**Sneha Murmu**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

Genome annotation is the process of identifying functional elements within a genome, such as genes, regulatory regions, and repeat elements. The goal of genome annotation is to create an accurate and comprehensive description of the genome's structure and function. This can be a time-consuming process, but it is essential for understanding how genes and other functional elements work together to control an organism's biology.

One powerful tool for genome annotation is Blast2GO (Conesa et al., 2005). Blast2GO is a commercial bioinformatics software suite that provides comprehensive functional annotation of nucleotide and protein sequences. It combines powerful sequence similarity search algorithms, such as BLAST (Altschul et al., 1997) and HMMER (Finn et al., 2011), with functional annotation tools, such as InterProScan (Zdobnov et al., 2001) and Gene Ontology (GO) mapping, to provide a detailed functional analysis of genomic and transcriptomic data.

Blast2GO works by first performing a sequence similarity search, typically using BLAST, to identify sequences with homology to known sequences in public databases. The resulting hits are then annotated using a variety of functional annotation tools, including InterProScan, which identifies conserved protein domains and functional motifs, and GO mapping, which assigns GO terms based on the functional categories of annotated genes.

Blast2GO also includes tools for statistical analysis and data visualization, allowing users to explore functional trends and patterns in their data. It can be used to analyze a wide range of genomic and transcriptomic data sets. One of the strengths of Blast2GO is its user-friendly interface, which allows even non-experts to perform complex functional annotation analyses. Blast2GO is also highly customizable, allowing users to tailor the annotation process to their specific needs and research questions.

Here are the four broad steps involved in genome annotation using Blast2GO:

- Sequence quality control and assembly: Before annotating a genome, it is important to ensure that the quality of the sequencing data is high and that the genome has been properly assembled. This may involve trimming low-quality sequences, filtering out contaminants, and performing de novo assembly or mapping to a reference genome.
- Sequence similarity search: The first step in genome annotation is to identify sequences with homology to known sequences in public databases. This is typically done using BLAST or a similar tool. The resulting hits can provide clues about the function and evolutionary relationships of the sequences in question.
- Functional annotation: Once sequences have been identified using a sequence similarity search, functional annotation tools can be used to identify functional domains and motifs, assign Gene Ontology terms, and perform other types of functional analysis. Blast2GO includes a number of annotation tools, including InterProScan, which searches for conserved domains and motifs in protein sequences, and GO mapping, which assigns Gene Ontology terms based on the functional categories of annotated genes.
- Data analysis and visualization: Once the sequences have been annotated with functional information, the data can be analyzed and visualized in a variety of ways. Blast2GO includes tools for statistical analysis and data visualization. The results of the analysis can be exported in a variety of formats for further analysis.

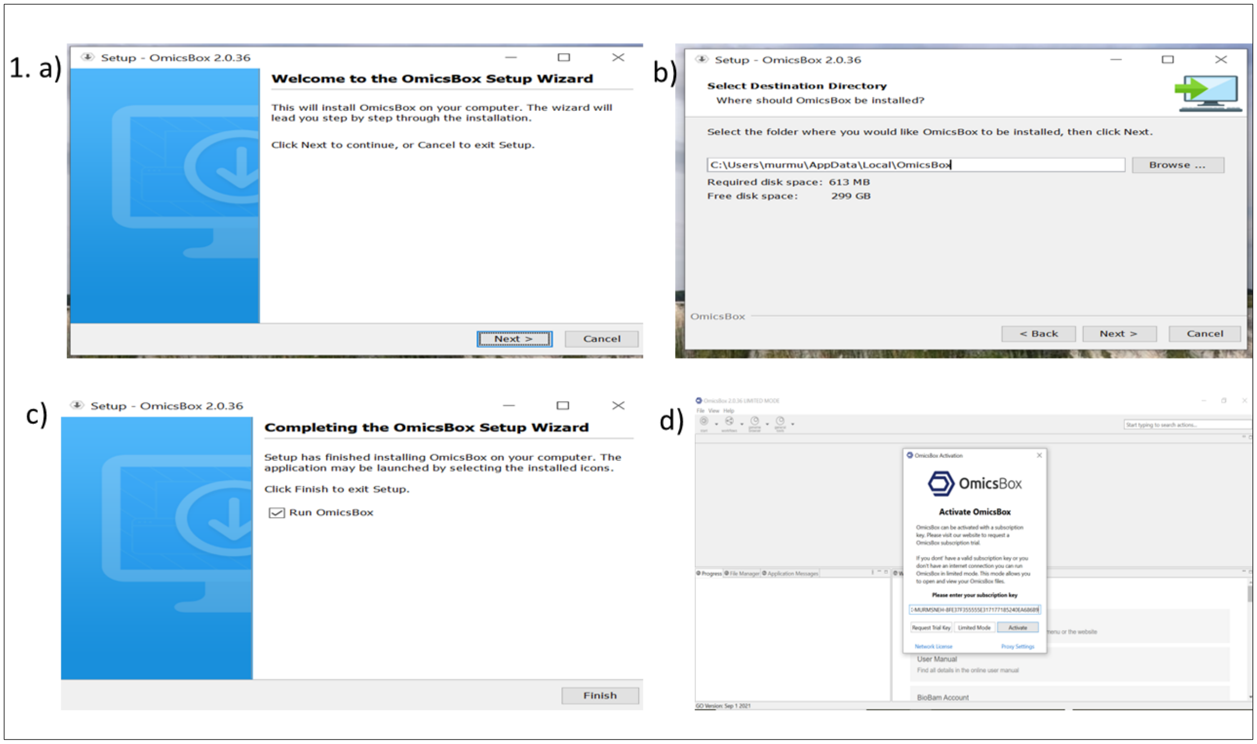
**Installation of Blast2GO:**

Following are the general steps to install Blast2GO:

1. System requirements: Check that your computer meets the system requirements for Blast2GO. Blast2GO is compatible with Windows, macOS, and Linux operating systems, and requires at least 8 GB of RAM.
2. Download Blast2GO: Visit the Blast2GO website (https://www.blast2go.com/) and download the appropriate installation file for your operating system. You may need to create an account and purchase a license, depending on your intended use of the software.
3. Install Blast2GO: Double-click the downloaded installation file and follow the on-screen instructions to install Blast2GO (as depicted in Figure 1). You may need to provide administrator permissions, depending on your operating system and security settings.

4. Configure Blast2GO: Once Blast2GO is installed, you will need to configure it to work with your specific computing environment. This may include setting preferences for sequence databases, annotation tools, and other settings.
5. Activate license: If you have purchased a license for Blast2GO, you will need to activate it before you can use the software. This typically involves entering a license key or activating the license through an online portal.

Once Blast2GO is installed and configured, you can begin using it to analyze and annotate your genomic or transcriptomic data.



Setup - OmicsBox 2.0.36

Welcome to the OmicsBox Setup Wizard

This will install OmicsBox on your computer. The wizard will lead you step by step through the installation.

Click Next to continue, or Cancel to exit Setup.

Next > Cancel

Setup - OmicsBox 2.0.36

Select Destination Directory
Where should OmicsBox be installed?

Select the folder where you would like OmicsBox to be installed, then click Next.

C:\Users\murmu\AppData\Local\OmicsBox Browse ...

Required disk space: 613 MB
Free disk space: 299 GB

OmicsBox

< Back Next > Cancel

Setup - OmicsBox 2.0.36

Completing the OmicsBox Setup Wizard

Setup has finished installing OmicsBox on your computer. The application may be launched by selecting the installed icons.

Click Finish to exit Setup.

☑ Run OmicsBox

Finish

OmicsBox 2.0.36 LIMITED MODE

File View Help

Start typing to search actions...

OmicsBox Activation

OmicsBox

Activate OmicsBox

OmicsBox can be activated with a subscription key. Please visit our website to request a OmicsBox subscription trial.

If you dont' have a valid subscription key or you don't have an internet connection you can run OmicsBox in limited mode. This mode allows you to open and view your OmicsBox files.

Please enter your subscription key

I-MURMSNEH-8FE37F355555E3171771852406A68689

Request Trial Key Limited Mode Activate

Network License Proxy Settings

Progress File Manager Application Messages

User Manual
Find all details in the online user manual

BioBam Account

GO Version: Sep 1 2021

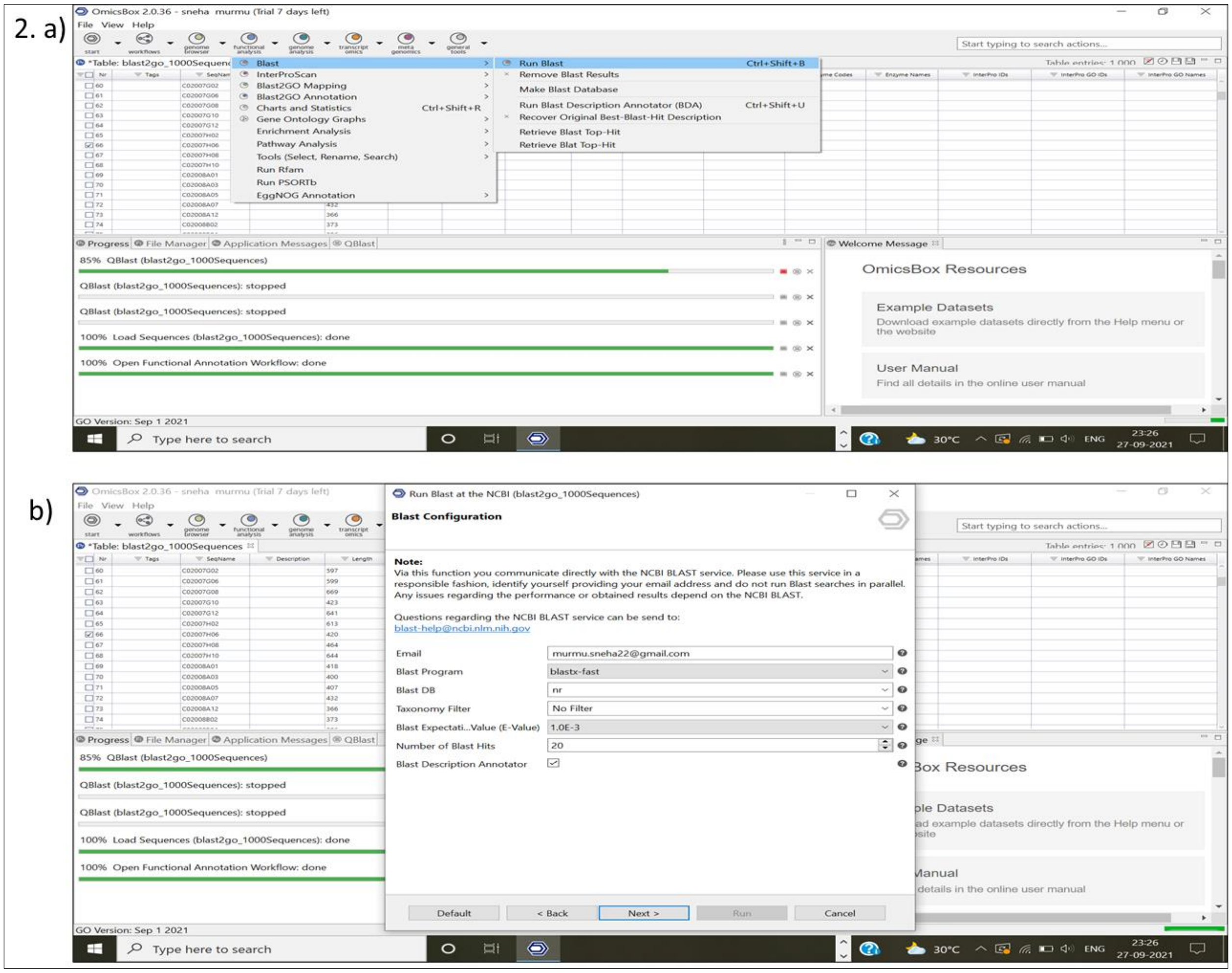
Figure 1: Installation steps of Blast2GO in Windows system.

**Stepwise guide to perform annotation using Blast2GO**

1. Open Blast2GO: Launch Blast2GO on your computer.
2. Load sequences: Load your sequence file(s) into Blast2GO. This can be done by clicking on "Load data" in the main menu and selecting the appropriate file type (e.g., FASTA).
3. Run **BLAST** search: In the main menu, click on "Run BLAST" and select the appropriate database for your search (e.g., NCBI non-redundant protein database) as shown in Figure 2. You can choose to run a BLASTP (protein query against protein database) or a BLASTX

(nucleotide query against protein database) search. You can also set various search parameters, such as the e-value threshold and the maximum number of hits to return.

4. View BLAST results: Once the BLAST search is complete, you can view the results in the BLAST results table (as shown in Figure 3). The table will show the sequence ID, the best hit, the e-value, the bit score, and other relevant information. You can sort the table by various columns to help you identify the best hits.
5. Import BLAST results: To import the BLAST results into the Blast2GO annotation pipeline, select the sequences you want to annotate and click on "Import selected hits". This will import the BLAST results and link them to the appropriate sequences in the annotation pipeline.



85% QBlast (blast2go_1000Sequences)

QBlast (blast2go_1000Sequences): stopped

QBlast (blast2go_1000Sequences): stopped

100% Load Sequences (blast2go_1000Sequences): done

100% Open Functional Annotation Workflow: done

Run Blast at the NCBI (blast2go_1000Sequences)

Blast Configuration

**Note:**
Via this function you communicate directly with the NCBI BLAST service. Please use this service in a responsible fashion, identify yourself providing your email address and do not run Blast searches in parallel. Any issues regarding the performance or obtained results depend on the NCBI BLAST.

Questions regarding the NCBI BLAST service can be send to:
blast-help@ncbi.nlm.nih.gov

| | |
|---|---|
| Email | murmu.sneha22@gmail.com |
| Blast Program | blastx-fast |
| Blast DB | nr |
| Taxonomy Filter | No Filter |
| Blast Expectati...Value (E-Value) | 1.0E-3 |
| Number of Blast Hits | 20 |
| Blast Description Annotator | ☑ |

Figure 2: BLAST search.

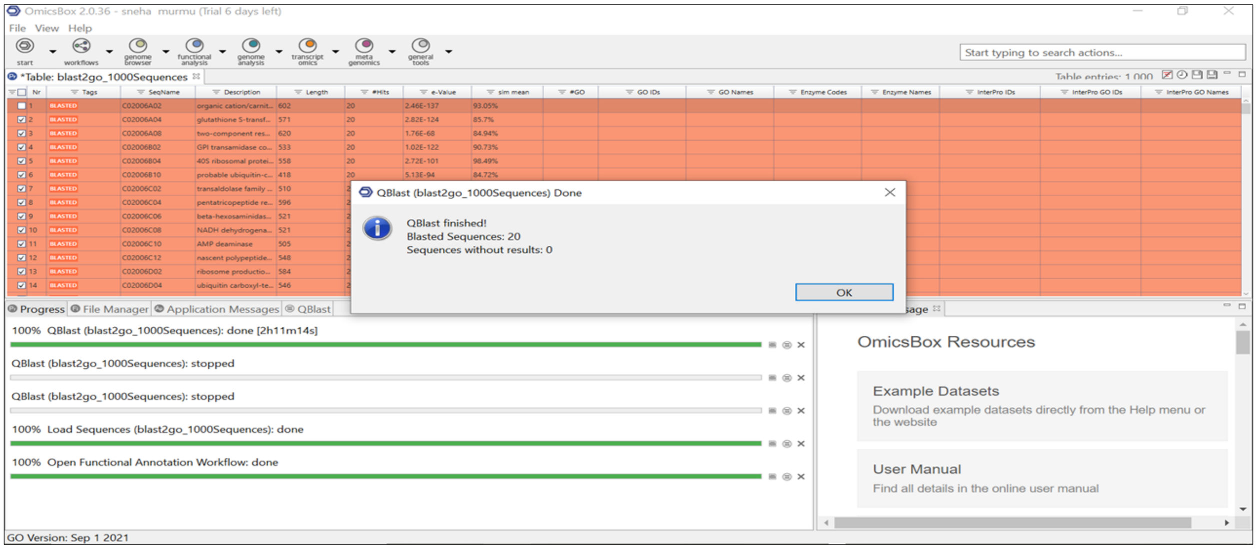

OmicsBox 2.0.36 - sneha  murmu (Trial 6 days left)

*Table: blast2go_1000Sequences

| Nr | Tags | SeqName | Description | Length | #Hits | e-Value | sim mean |
|---|---|---|---|---|---|---|---|
| □ 1 | BLASTED | C02006A02 | organic cation/carnit... | 602 | 20 | 2.46E-137 | 93.05% |
| ☑ 2 | BLASTED | C02006A04 | glutathione S-transf... | 571 | 20 | 2.82E-124 | 85.7% |
| ☑ 3 | BLASTED | C02006A08 | two-component res... | 620 | 20 | 1.76E-68 | 84.94% |
| ☑ 4 | BLASTED | C02006B02 | GPI transamidase co... | 533 | 20 | 1.02E-122 | 90.73% |
| ☑ 5 | BLASTED | C02006B04 | 40S ribosomal protei... | 558 | 20 | 2.72E-101 | 98.49% |
| ☑ 6 | BLASTED | C02006B10 | probable ubiquitin-c... | 418 | 20 | 5.13E-94 | 84.72% |
| ☑ 7 | BLASTED | C02006C02 | transaldolase family ... | 510 | | | |
| ☑ 8 | BLASTED | C02006C04 | pentatricopeptide re... | 596 | | | |
| ☑ 9 | BLASTED | C02006C06 | beta-hexosaminidas... | 521 | | | |
| ☑ 10 | BLASTED | C02006C08 | NADH dehydrogena... | 521 | | | |
| ☑ 11 | BLASTED | C02006C10 | AMP deaminase | 505 | | | |
| ☑ 12 | BLASTED | C02006C12 | nascent polypeptide... | 548 | | | |
| ☑ 13 | BLASTED | C02006D02 | ribosome productio... | 584 | | | |
| ☑ 14 | BLASTED | C02006D04 | ubiquitin carboxyl-te... | 546 | | | |

QBlast (blast2go_1000Sequences) Done

QBlast finished!
Blasted Sequences: 20
Sequences without results: 0

OK

100% QBlast (blast2go_1000Sequences): done [2h11m14s]

QBlast (blast2go_1000Sequences): stopped

QBlast (blast2go_1000Sequences): stopped

100% Load Sequences (blast2go_1000Sequences): done

100% Open Functional Annotation Workflow: done

OmicsBox Resources

Example Datasets
Download example datasets directly from the Help menu or the website

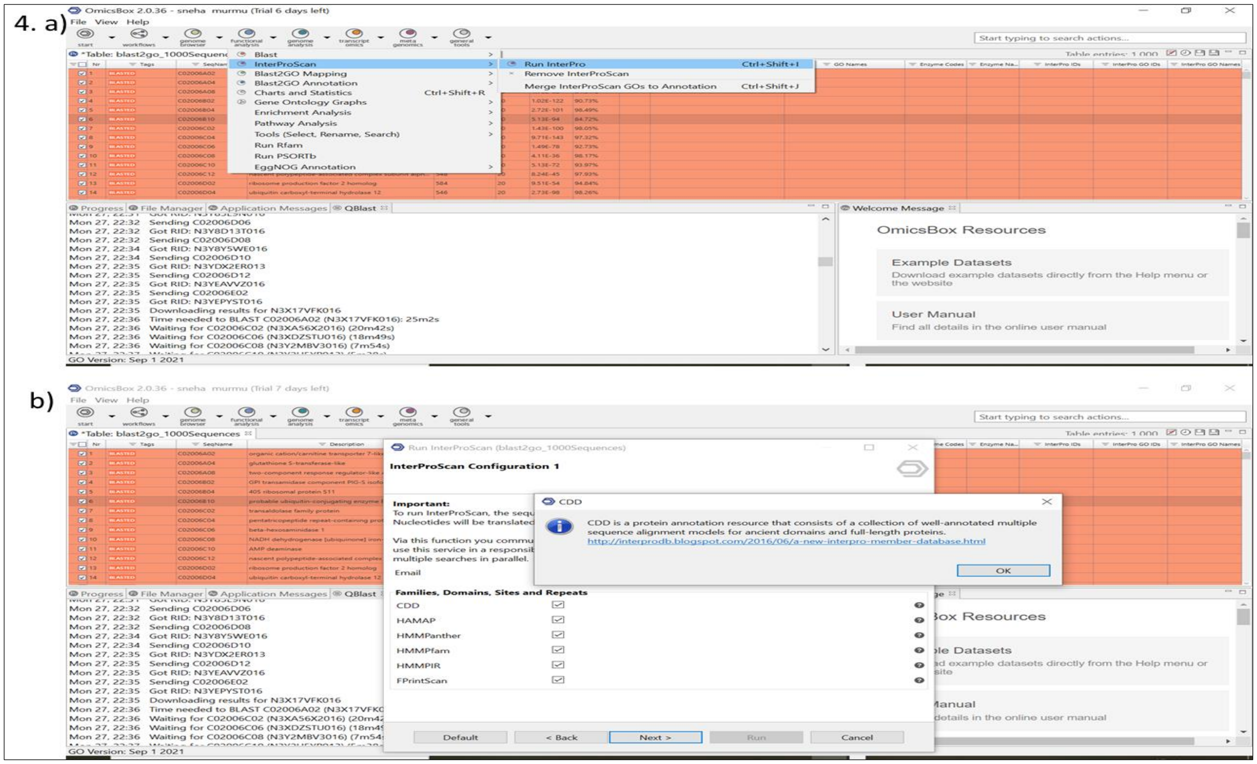
User Manual
Find all details in the online user manual

GO Version: Sep 1 2021

Figure 3: BLAST result.

6. Run **InterProScan**: In the main menu, click on "Run InterProScan" and select the appropriate database for your search (e.g., InterPro database). You can choose to run the search on protein or nucleotide sequences (Figure 4a).

7. Set search parameters: You can set various search parameters, such as the e-value threshold, the maximum number of sequences to align, and the type of analysis to perform (e.g., Pfam, Prosite, SMART, etc.) (Figure 4b).


4. a)
OmicsBox 2.0.36 - sneha murmu (Trial 6 days left)
Blast
InterProScan
Blast2GO Mapping
Blast2GO Annotation
Charts and Statistics Ctrl+Shift+R
Gene Ontology Graphs
Enrichment Analysis
Pathway Analysis
Tools (Select, Rename, Search)
Run Rfam
Run PSORTb
EggNOG Annotation
Run InterPro Ctrl+Shift+I
Remove InterProScan
Merge InterProScan GOs to Annotation Ctrl+Shift+J
b)
OmicsBox 2.0.36 - sneha murmu (Trial 7 days left)

```
Mon 27, 22:32   Sending C02006D06
Mon 27, 22:32   Got RID: N3Y8D13T016
Mon 27, 22:32   Sending C02006D08
Mon 27, 22:34   Got RID: N3Y8Y5WE016
Mon 27, 22:34   Sending C02006D10
Mon 27, 22:35   Got RID: N3YDX2ER013
Mon 27, 22:35   Sending C02006D12
Mon 27, 22:35   Got RID: N3YEAVVZ016
Mon 27, 22:35   Sending C02006E02
Mon 27, 22:35   Got RID: N3YEPYST016
Mon 27, 22:35   Downloading results for N3X17VFK016
Mon 27, 22:36   Time needed to BLAST C02006A02 (N3X17VFK016): 25m2s
Mon 27, 22:36   Waiting for C02006C02 (N3XA56X2016) (20m42s)
Mon 27, 22:36   Waiting for C02006C06 (N3XDZSTU016) (18m49s)
Mon 27, 22:36   Waiting for C02006C08 (N3Y2MBV3016) (7m54s)
```

GO Version: Sep 1 2021

Run InterProScan (blast2go_1000Sequences)

InterProScan Configuration 1

Important:
To run InterProScan, the sequ
Nucleotides will be translated

Via this function you commu
use this service in a responsib
multiple searches in parallel.

Email

CDD

CDD is a protein annotation resource that consists of a collection of well-annotated multiple sequence alignment models for ancient domains and full-length proteins.
http://interprodb.blogspot.com/2016/06/a-new-interpro-member-database.html

OK

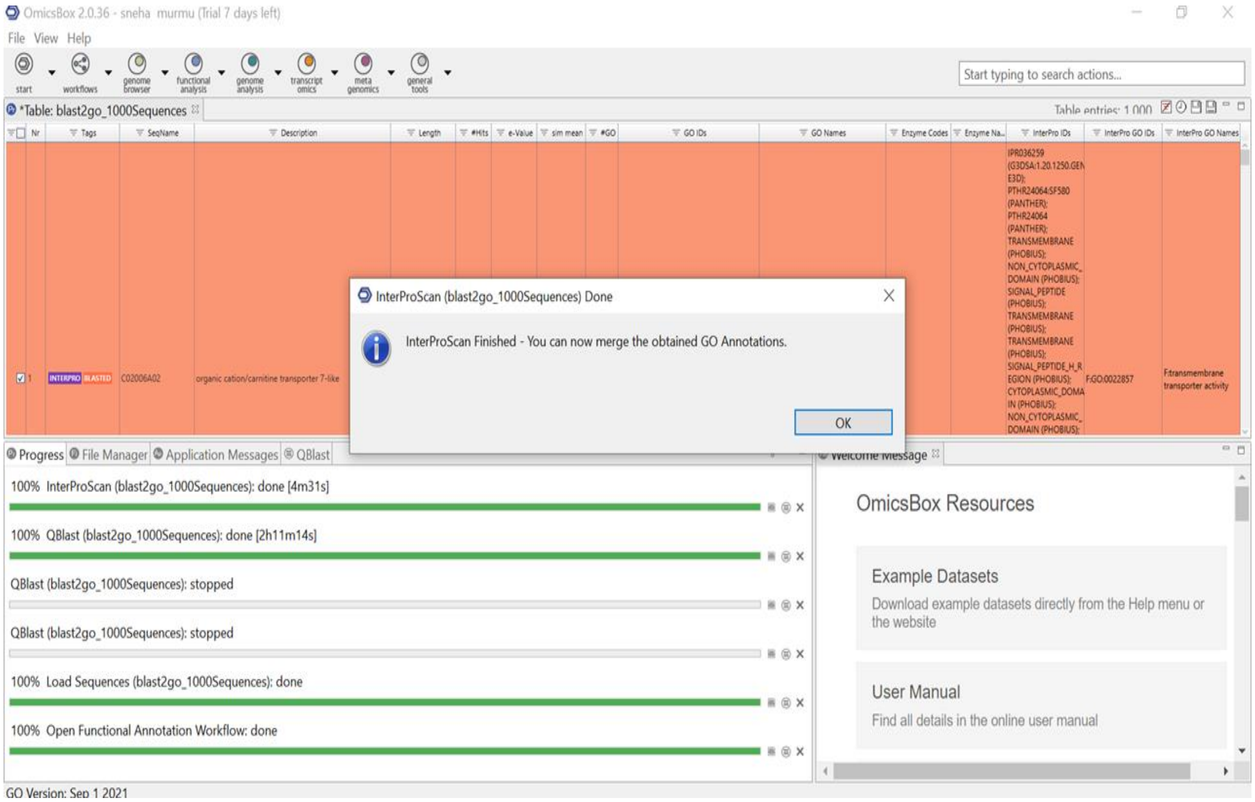
Families, Domains, Sites and Repeats

| | |
|---|---|
| CDD | ☑ |
| HAMAP | ☑ |
| HMMPanther | ☑ |
| HMMPfam | ☑ |
| HMMPIR | ☑ |
| FPrintScan | ☑ |

Default  < Back  Next >  Run  Cancel

Figure 4: InterProScan search.

8. View InterProScan results: Once the InterProScan search is complete, you can view the results in the InterProScan results table. The table will show the sequence ID, the best match, the e-value, the score, and other relevant information (Figure 5). You can sort the table by various columns to help you identify the best matches.



| ☐ | Nr | Tags | SeqName | Description | Length | #Hits | e-Value | sim mean | #GO | GO IDs | GO Names | Enzyme Codes | Enzyme Na... | InterPro IDs | InterPro GO IDs | InterPro GO Names |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ☑ | 1 | INTERPRO BLASTED | C02006A02 | organic cation/carnitine transporter 7-like | | | | | | | | | | IPR036259 (G3DSA:1.20.1250.GENE3D); PTHR24064:SF580 (PANTHER); PTHR24064 (PANTHER); TRANSMEMBRANE (PHOBIUS); NON_CYTOPLASMIC_DOMAIN (PHOBIUS); SIGNAL_PEPTIDE (PHOBIUS); TRANSMEMBRANE (PHOBIUS); TRANSMEMBRANE (PHOBIUS); SIGNAL_PEPTIDE_H_REGION (PHOBIUS); CYTOPLASMIC_DOMAIN (PHOBIUS); NON_CYTOPLASMIC_DOMAIN (PHOBIUS); | F:GO:0022857 | F:transmembrane transporter activity |

InterProScan (blast2go_1000Sequences) Done

InterProScan Finished - You can now merge the obtained GO Annotations.

100% InterProScan (blast2go_1000Sequences): done [4m31s]

100% QBlast (blast2go_1000Sequences): done [2h11m14s]

QBlast (blast2go_1000Sequences): stopped

QBlast (blast2go_1000Sequences): stopped

100% Load Sequences (blast2go_1000Sequences): done

100% Open Functional Annotation Workflow: done

OmicsBox Resources

Example Datasets

Download example datasets directly from the Help menu or the website

User Manual

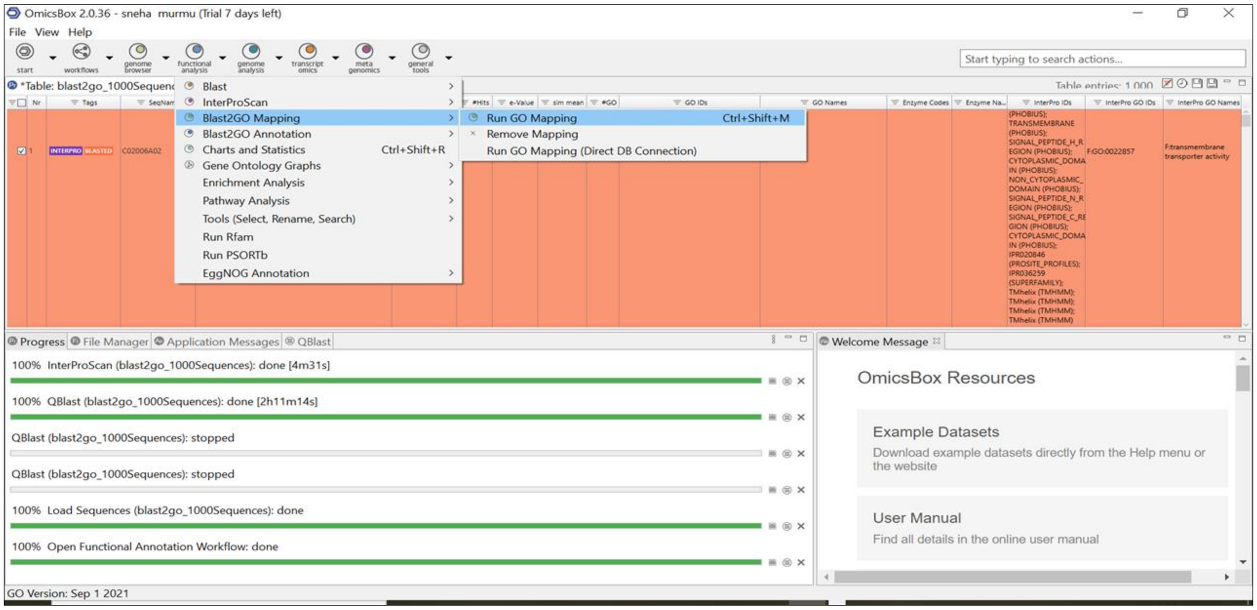
Find all details in the online user manual

GO Version: Sep 1 2021

Figure 5: InterProScan result.

9. Import InterProScan results: To import the InterProScan results into the Blast2GO annotation pipeline, select the sequences you want to annotate and click on "Import selected hits". This will import the InterProScan results and link them to the appropriate sequences in the annotation pipeline.

10. Perform **mapping**: Once the BLAST results have been imported, you can use the Blast2GO mapping tools to map your sequences to Gene Ontology (GO) terms (Figure 6). This involves using the BLAST results to transfer functional annotations from similar sequences to your own sequences.



100% InterProScan (blast2go_1000Sequences): done [4m31s]

100% QBlast (blast2go_1000Sequences): done [2h11m14s]

QBlast (blast2go_1000Sequences): stopped

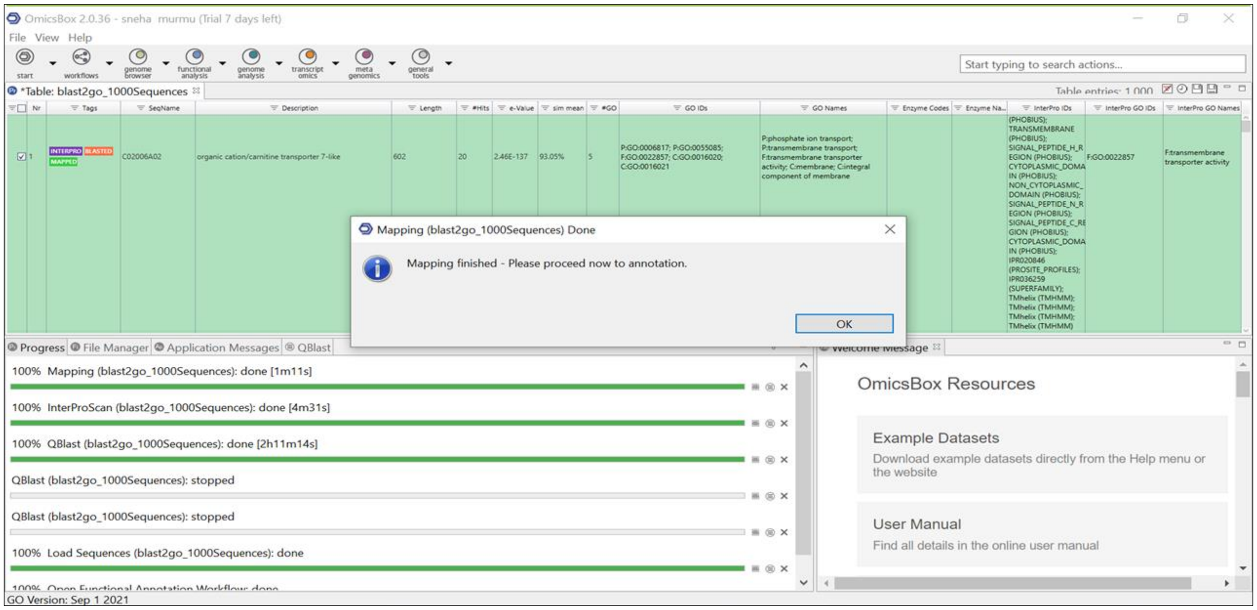
QBlast (blast2go_1000Sequences): stopped

100% Load Sequences (blast2go_1000Sequences): done

100% Open Functional Annotation Workflow: done

Figure 6: Mapping.

11. Edit mappings: You can edit the mappings manually, by adding or removing GO terms, or by changing the evidence codes. You can also remove or filter out low-confidence mappings, based on various criteria such as the e-value, the similarity score, or the GO term specificity.
12. Export mapping results: Once your sequences have been mapped, you can export the results in a variety of formats, such as tab-delimited text files or FASTA files (Figure 7). These results can be used for further analysis.



| Nr | Tags | SeqName | Description | Length | #Hits | e-Value | sim mean | #GO | GO IDs | GO Names | Enzyme Codes | Enzyme Na... | InterPro IDs | InterPro GO IDs | InterPro GO Names |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ☑ 1 | INTERPRO BLASTED MAPPED | C02006A02 | organic cation/carnitine transporter 7-like | 602 | 20 | 2.46E-137 | 93.05% | 5 | P:GO:0006817; P:GO:0055085; F:GO:0022857; C:GO:0016020; C:GO:0016021 | P:phosphate ion transport; P:transmembrane transport; F:transmembrane transporter activity; C:membrane; C:integral component of membrane | | | (PHOBIUS); TRANSMEMBRANE (PHOBIUS); SIGNAL_PEPTIDE_H_REGION (PHOBIUS); CYTOPLASMIC_DOMAIN (PHOBIUS); NON_CYTOPLASMIC_DOMAIN (PHOBIUS); SIGNAL_PEPTIDE_N_REGION (PHOBIUS); SIGNAL_PEPTIDE_C_REGION (PHOBIUS); CYTOPLASMIC_DOMAIN (PHOBIUS); IPR020846 (PROSITE_PROFILES); IPR036259 (SUPERFAMILY); TMhelix (TMHMM); TMhelix (TMHMM); TMhelix (TMHMM); TMhelix (TMHMM) | F:GO:0022857 | F:transmembrane transporter activity |

Mapping finished - Please proceed now to annotation.

100% Mapping (blast2go_1000Sequences): done [1m11s]

100% InterProScan (blast2go_1000Sequences): done [4m31s]

100% QBlast (blast2go_1000Sequences): done [2h11m14s]

QBlast (blast2go_1000Sequences): stopped

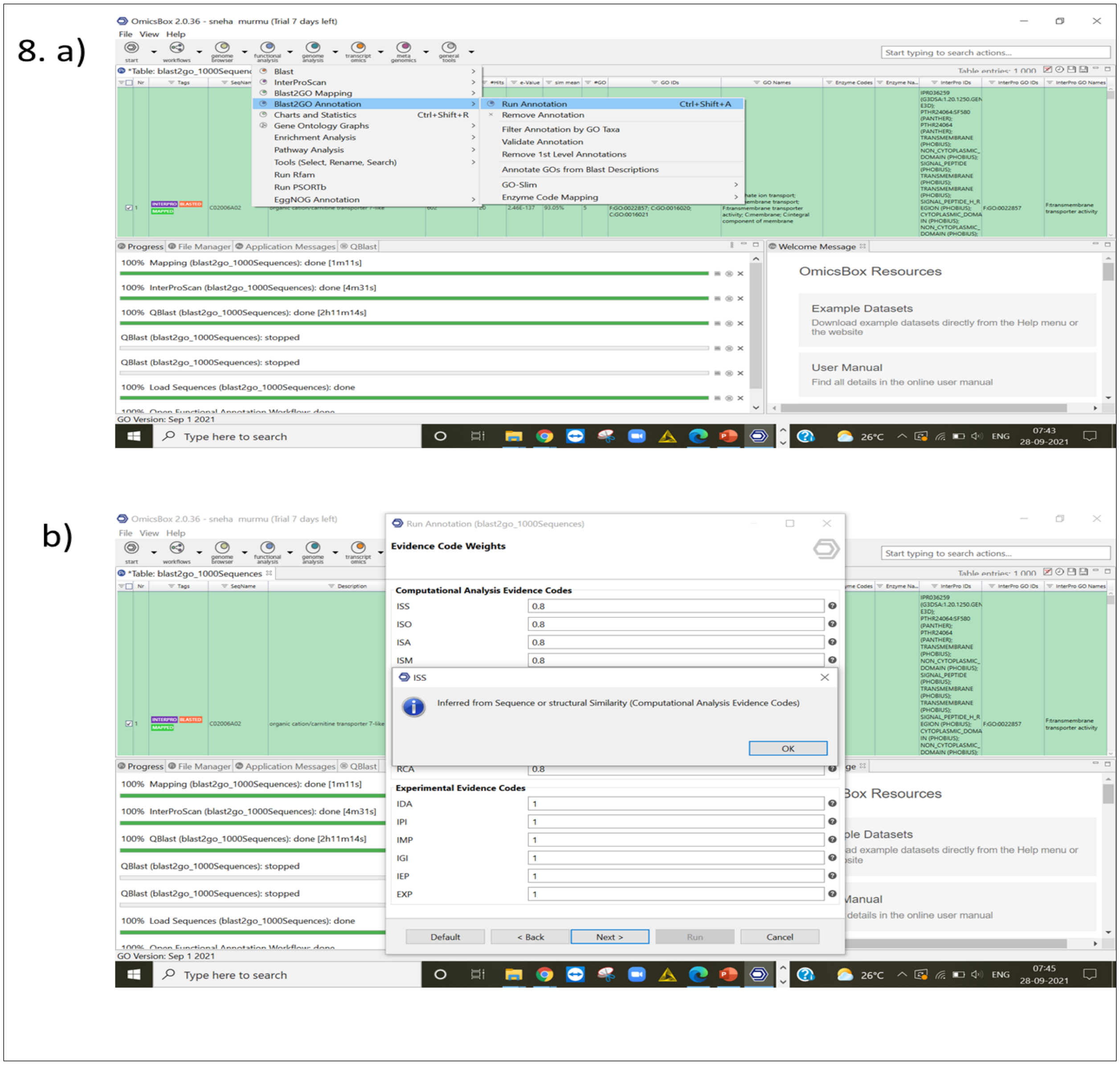
QBlast (blast2go_1000Sequences): stopped

100% Load Sequences (blast2go_1000Sequences): done

100% Open Functional Annotation Workflow: done

Figure 7: Mapping result.

13. **Annotate** sequences: Once the InterProScan results have been imported, you can use the Blast2GO annotation tools to assign functional information to your sequences (Figure 8). This may include mapping Gene Ontology (GO) terms, performing enrichment analysis, and performing other types of functional analysis.



100% Mapping (blast2go_1000Sequences): done [1m11s]
100% InterProScan (blast2go_1000Sequences): done [4m31s]
100% QBlast (blast2go_1000Sequences): done [2h11m14s]
QBlast (blast2go_1000Sequences): stopped
QBlast (blast2go_1000Sequences): stopped
100% Load Sequences (blast2go_1000Sequences): done
100% Open Functional Annotation Workflow: done
GO Version: Sep 1 2021

Computational Analysis Evidence Codes

| | |
|---|---|
| ISS | 0.8 |
| ISO | 0.8 |
| ISA | 0.8 |
| ISM | 0.8 |
| RCA | 0.8 |

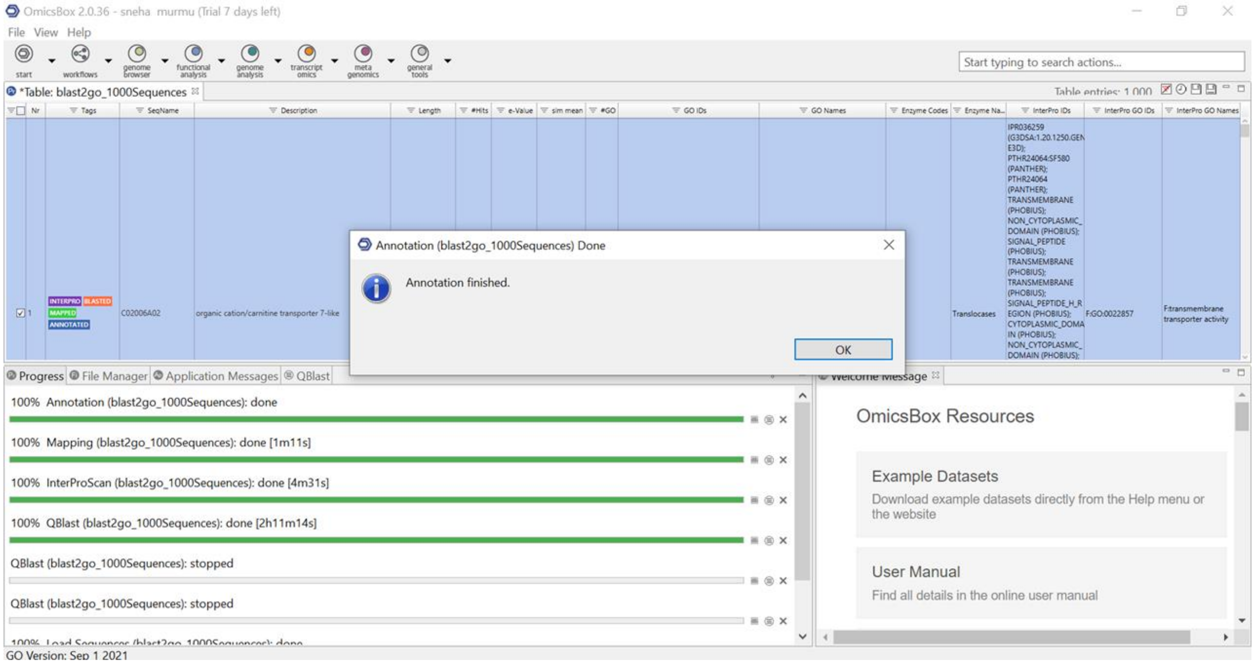
Inferred from Sequence or structural Similarity (Computational Analysis Evidence Codes)

Experimental Evidence Codes

| | |
|---|---|
| IDA | 1 |
| IPI | 1 |
| IMP | 1 |
| IGI | 1 |
| IEP | 1 |
| EXP | 1 |

Figure 8: Annotate.

14. Export annotation results: Once your sequences have been annotated, you can export the results in a variety of formats, such as tab-delimited text files or FASTA files. These results can be used for further analysis, visualization, or sharing with collaborators.



Annotation (blast2go_1000Sequences) Done

Annotation finished.

OK

100% Annotation (blast2go_1000Sequences): done

100% Mapping (blast2go_1000Sequences): done [1m11s]

100% InterProScan (blast2go_1000Sequences): done [4m31s]

100% QBlast (blast2go_1000Sequences): done [2h11m14s]

QBlast (blast2go_1000Sequences): stopped

QBlast (blast2go_1000Sequences): stopped

OmicsBox Resources

Example Datasets

Download example datasets directly from the Help menu or the website

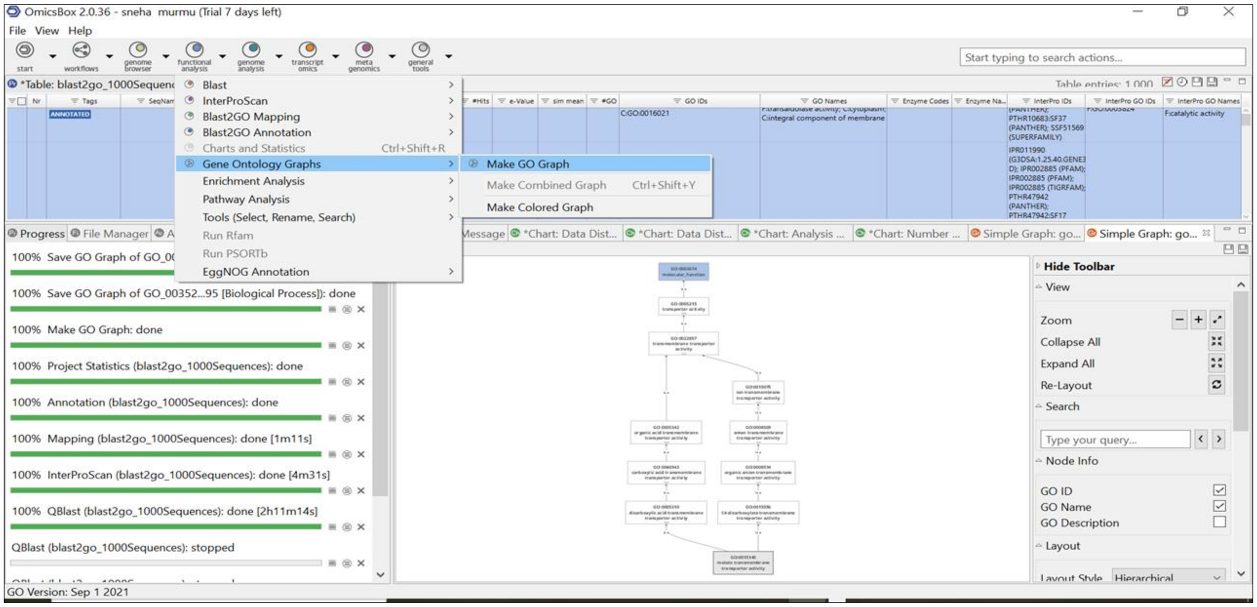
User Manual

Find all details in the online user manual

GO Version: Sep 1 2021

Figure 9: Annotate result.

15. Generate Gene Ontology (GO) graph: To create a GO graph in Blast2GO, click on "Graphs" in the main menu and select "GO Graph" (Figure 10). This will generate a graphical representation of the GO terms assigned to your sequences, based on the hierarchical structure of the Gene Ontology.



100% Save GO Graph of GO_00352...95 [Biological Process]): done

100% Make GO Graph: done

100% Project Statistics (blast2go_1000Sequences): done

100% Annotation (blast2go_1000Sequences): done

100% Mapping (blast2go_1000Sequences): done [1m11s]

100% InterProScan (blast2go_1000Sequences): done [4m31s]

100% QBlast (blast2go_1000Sequences): done [2h11m14s]

QBlast (blast2go_1000Sequences): stopped

GO Version: Sep 1 2021

Figure 10. Generate GO graph.

16. Customize GO graph: You can customize the appearance of the GO graph by changing the colors, font sizes, or layout. You can also filter the GO terms based on various criteria such as

the level in the hierarchy, the number of sequences assigned to the term, or the statistical significance of the enrichment.

17. Analyze GO graph: Once you have generated a GO graph, you can use it to analyze the functional annotations of your sequences. This can include identifying overrepresented or underrepresented GO terms, comparing the GO profiles of different datasets or treatments, or visualizing the relationships between different biological processes, molecular functions, or cellular components (Figure 11).

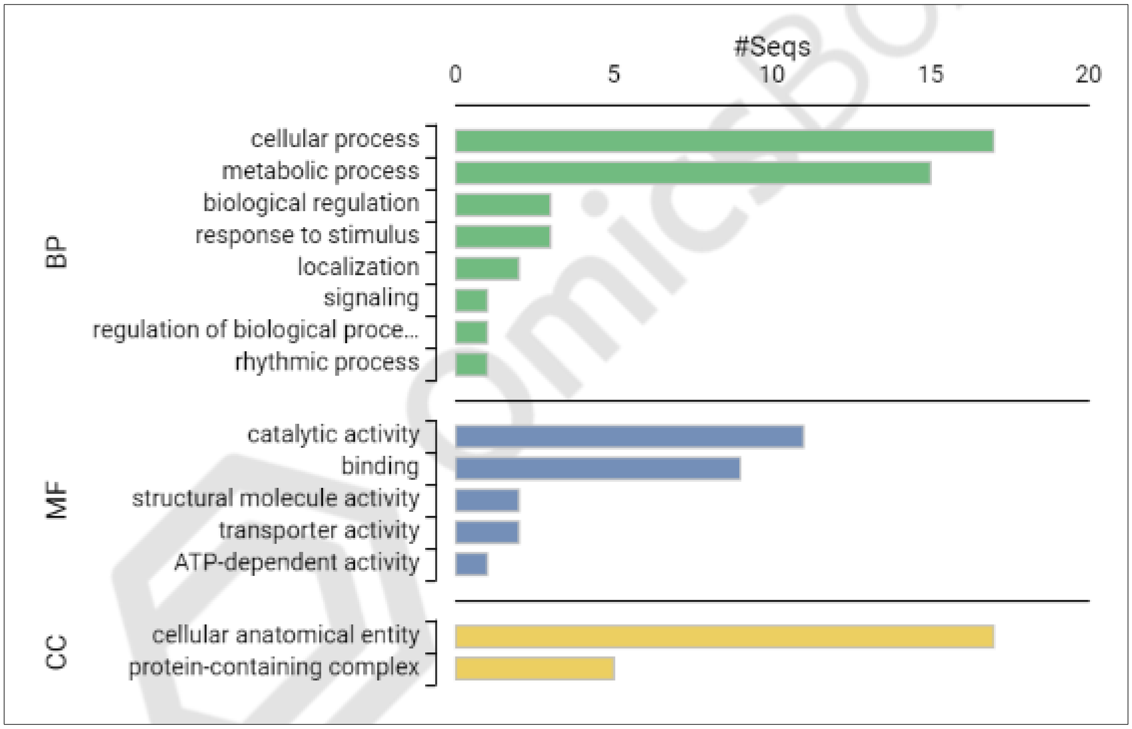
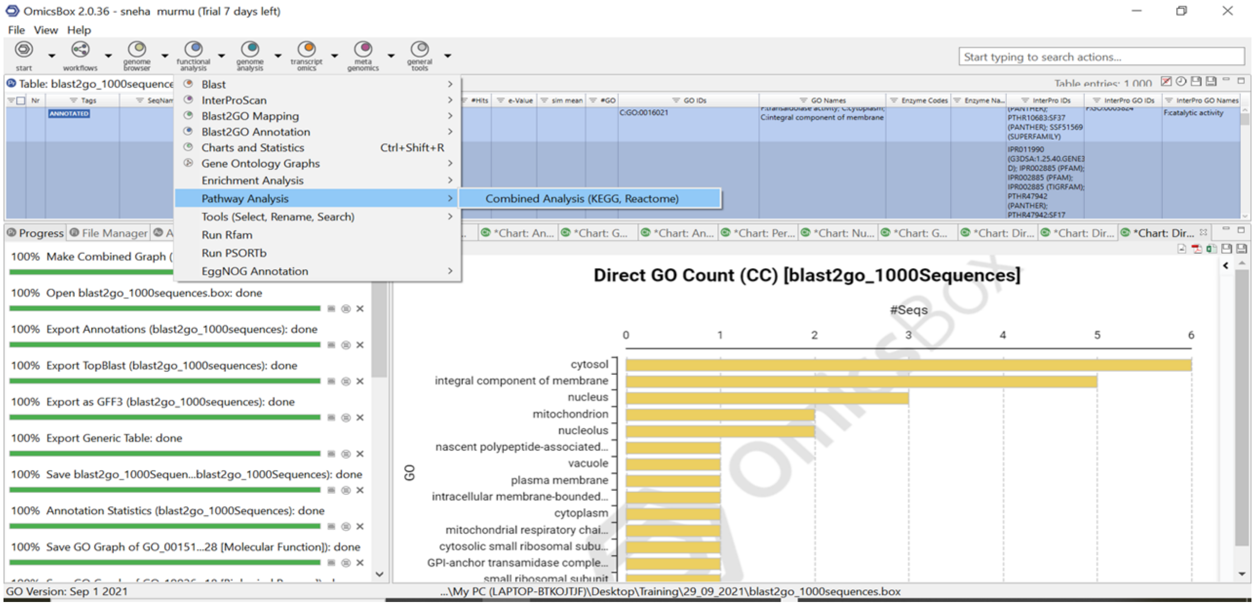


Figure 11: GO graph.

18. Export GO graph: Once you have customized and analyzed your GO graph, you can export it in a variety of formats, such as PNG, PDF, or SVG. These graphs can be used for presentations, publications, or further analysis with other tools or software.

19. Perform pathway analysis: To perform pathway analysis in Blast2GO, you need to use the KEGG (Kyoto Encyclopedia of Genes and Genomes) pathway database. In the main menu, click on "Annotation" and select "Pathway annotation". This will open the pathway annotation dialog box (Figure 12).


OmicsBox 2.0.36 - sneha murmu (Trial 7 days left)
File View Help
start
workflows
genome browser
functional analysis
genome analysis
transcript omics
meta genomics
general tools
Start typing to search actions...
Table: blast2go_1000sequence
Table entries: 1.000
Blast
InterProScan
Blast2GO Mapping
Blast2GO Annotation
Charts and Statistics Ctrl+Shift+R
Gene Ontology Graphs
Enrichment Analysis
Pathway Analysis
Tools (Select, Rename, Search)
Run Rfam
Run PSORTb
EggNOG Annotation
Combined Analysis (KEGG, Reactome)
Progress
File Manager
Direct GO Count (CC) [blast2go_1000Sequences]
#Seqs
0
1
2
3
4
5
6
GO
cytosol
integral component of membrane
nucleus
mitochondrion
nucleolus
nascent polypeptide-associated...
vacuole
plasma membrane
intracellular membrane-bounded...
cytoplasm
mitochondrial respiratory chai...
cytosolic small ribosomal subu...
GPI-anchor transamidase comple...
small ribosomal subunit

100% Make Combined Graph (

100% Open blast2go_1000sequences.box: done

100% Export Annotations (blast2go_1000sequences): done

100% Export TopBlast (blast2go_1000sequences): done

100% Export as GFF3 (blast2go_1000sequences): done

100% Export Generic Table: done

100% Save blast2go_1000Sequen...blast2go_1000Sequences): done

100% Annotation Statistics (blast2go_1000Sequences): done

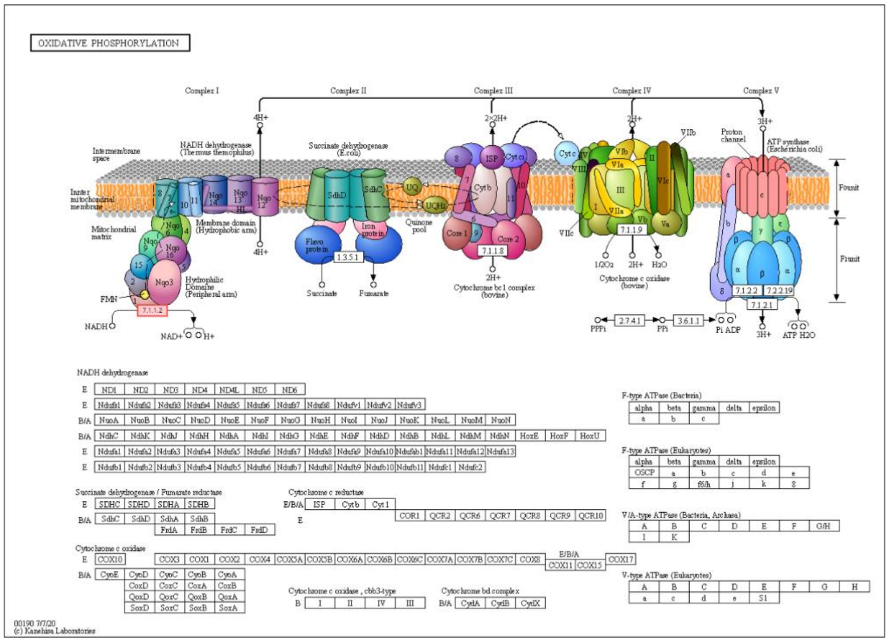
100% Save GO Graph of GO_00151...28 [Molecular Function]): done

GO Version: Sep 1 2021

...\My PC (LAPTOP-BTKOJTJF)\Desktop\Training\29_09_2021\blast2go_1000sequences.box

Figure 12. Run Pathway Analysis.

20. Select pathway database: In the pathway annotation dialog box, select the "KEGG" database and click on "Start". Blast2GO will download and install the latest version of the KEGG database on your computer.
21. Run pathway analysis: Once the KEGG database is installed, you can use the Blast2GO pathway analysis tools to identify the KEGG pathways that are enriched in your sequences. This involves comparing the frequency of KEGG pathway terms in your sequences to the frequency of these terms in a reference dataset, such as the entire KEGG database.
22. Filter and visualize pathways: Once the pathway analysis is complete, you can use the Blast2GO pathway analysis tools to filter and visualize the enriched pathways. This can involve setting statistical thresholds, such as the false discovery rate (FDR) or the p-value, or selecting specific pathways based on their relevance to your research question.
23. Analyze pathways: Once you have identified the enriched pathways, you can use the Blast2GO pathway analysis tools to analyze the functional annotations and gene products associated with these pathways. This can include identifying the key enzymes or regulators, comparing the pathway profiles of different datasets or treatments, or visualizing the relationships between different metabolic or signaling pathways (Figure 13).



NADH dehydrogenase

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E | ND1 | ND2 | ND3 | ND4 | ND4L | ND5 | ND6 | | | | | | | | | | |
| E | Ndufs1 | Ndufs2 | Ndufs3 | Ndufs4 | Ndufs5 | Ndufs6 | Ndufs7 | Ndufs8 | Ndufv1 | Ndufv2 | Ndufv3 | | | | | | |
| B/A | NuoA | NuoB | NuoC | NuoD | NuoE | NuoF | NuoG | NuoH | NuoI | NuoJ | NuoK | NuoL | NuoM | NuoN | | | |
| B/A | NdhC | NdhK | NdhJ | NdhH | NdhA | NdhI | NdhG | NdhE | NdhF | NdhD | NdhB | NdhL | NdhM | NdhN | HoxE | HoxF | HoxU |
| E | Ndufa1 | Ndufa2 | Ndufa3 | Ndufa4 | Ndufa5 | Ndufa6 | Ndufa7 | Ndufa8 | Ndufa9 | Ndufa10 | Ndufab1 | Ndufa11 | Ndufa12 | Ndufa13 | | | |
| E | Ndufb1 | Ndufb2 | Ndufb3 | Ndufb4 | Ndufb5 | Ndufb6 | Ndufb7 | Ndufb8 | Ndufb9 | Ndufb10 | Ndufb11 | Ndufc1 | Ndufc2 | | | | |

Succinate dehydrogenase / Fumarate reductase

| | | | | | |
|---|---|---|---|---|---|
| E | SDHC | SDHD | SDHA | SDHB | | |
| B/A | SdhC | SdhD | SdhA | SdhB | | |
| | | | FrdA | FrdB | FrdC | FrdD |

Cytochrome c reductase

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| E/B/A | ISP | Cytb | Cyt1 | | | | | | |
| E | | | | COR1 | QCR2 | QCR6 | QCR7 | QCR8 | QCR9 | QCR10 |

Cytochrome c oxidase

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E | COX10 | | COX3 | COX1 | COX2 | COX4 | COX5A | COX5B | COX6A | COX6B | COX6C | COX7A | COX7B | COX7C | COX8 | COX17 |
| E/B/A | | | | | | | | | | | | | | | COX11<br>COX15 | |
| B/A | CyoE | CyoD | CyoC | CyoB | CyoA | | | | | | | | | | | |
| | | CoxD | CoxC | CoxA | CoxB | | | | | | | | | | | |
| | | QoxD | QoxC | QoxB | QoxA | | | | | | | | | | | |
| | | SoxD | SoxC | SoxB | SoxA | | | | | | | | | | | |

Cytochrome c oxidase , cbb3-type

| | | | | |
|---|---|---|---|---|
| B | I | II | IV | III |

Cytochrome bd complex

| | | | |
|---|---|---|---|
| B/A | CydA | CydB | CydX |

F-type ATPase (Bacteria)

| | | | | |
|---|---|---|---|---|
| alpha | beta | gamma | delta | epsilon |
| a | b | c | | |

F-type ATPase (Eukaryotes)

| | | | | | |
|---|---|---|---|---|---|
| alpha | beta | gamma | delta | epsilon | |
| OSCP | a | b | c | d | e |
| f | g | f6/h | j | k | 8 |

V/A-type ATPase (Bacteria, Archaea)

| | | | | | | |
|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G/H |
| I | K | | | | | |

V-type ATPase (Eukaryotes)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | H |
| a | c | d | e | S1 | | | |

Figure 13. Pathway graph.

24. Export pathway data: Once you have customized and analyzed your pathway data, you can export it in a variety of formats, such as Excel, CSV, or XML. These data can be used for further analysis with other tools or software, or for visualizing and communicating the results of your pathway analysis.

## References

Conesa, A., Götz, S., García-Gómez, J. M., Terol, J., Talón, M., & Robles, M. (2005). Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research. Bioinformatics, 21(18), 3674-3676.

Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic acids research, 25(17), 3389-3402.

Finn, R. D., Clements, J., & Eddy, S. R. (2011). HMMER web server: interactive sequence similarity searching. Nucleic acids research, 39(suppl_2), W29-W37.

Zdobnov, E. M., & Apweiler, R. (2001). InterProScan–an integration platform for the signature-recognition methods in InterPro. Bioinformatics, 17(9), 847-848.

# Introduction to R for Bioinformatics

**Sudhir Srivastava, D. C. Mishra and Deepa Bhatt**

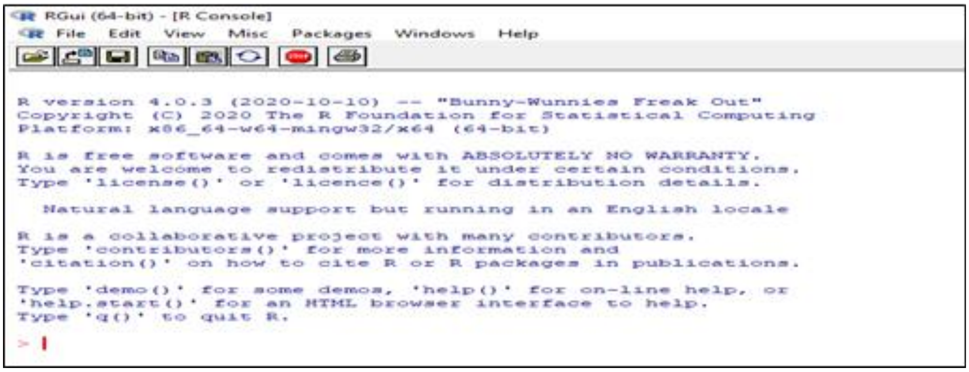
**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

R is a programming language that allows for advanced statistical computing and graphics. It was created by the statisticians Ross Ihaka and Robert Gentleman. It is supported by the R Core Team and the R Foundation for Statistical Computing. The language is very powerful for writing programs. Output may be limited based on the function, but even small code can generate wonderful graphics. It is very sensitive to syntax, case, punctuation used, even spacing. R is open source and free on the Internet. R is used among statisticians, computer scientists and bioinformaticians for data analysis and developing statistical software. The official R software environment is an open-source free software environment within the GNU package, available under the GNU General Public License. It is written primarily in C, Fortran, and R itself (partially self-hosting). Precompiled executables are provided for various operating systems. R has a command line interface as well as multiple third-party graphical user interfaces such as RStudio (an integrated development environment) and Jupyter (a notebook interface).

## Working in R and RStudio

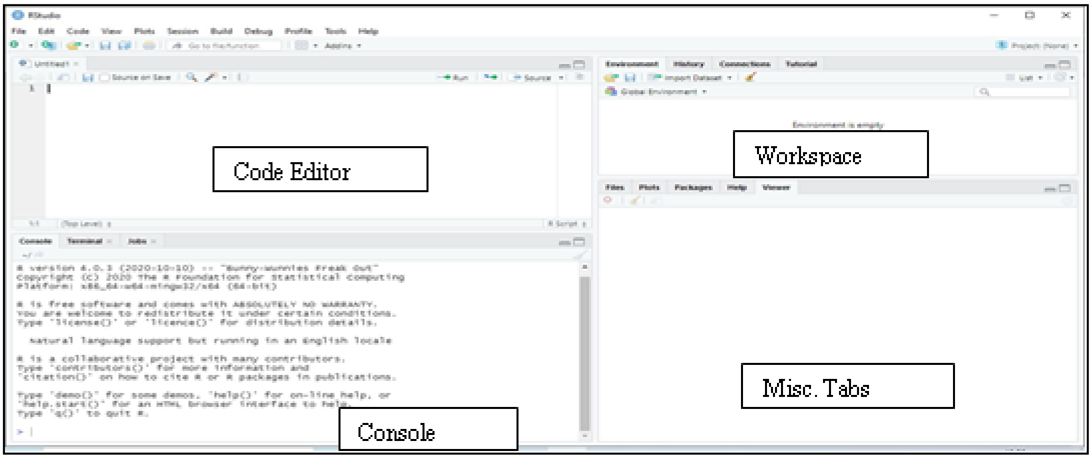
R can be installed in Linux, Unix, Windows and Mac platforms from www.r-project.org. For downloading R, please visit https://cloud.r-project.org/.



```
R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

The R GUI

RStudio is a free, open-source IDE (integrated development environment) for R. It can be downloaded from https://www.rstudio.com/products/rstudio/download/. One must install R before installing RStudio. The interface is organized so that the user can clearly view graphs, data tables, R code, and output all at the same time.

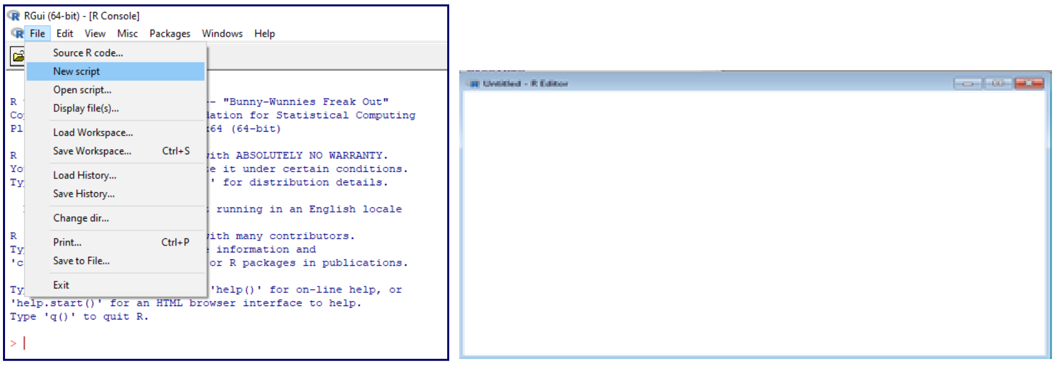


```
R version 4.0.3 (2020-10-10) -- "Bunny-Wunnies Freak Out"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

R Studio Interface

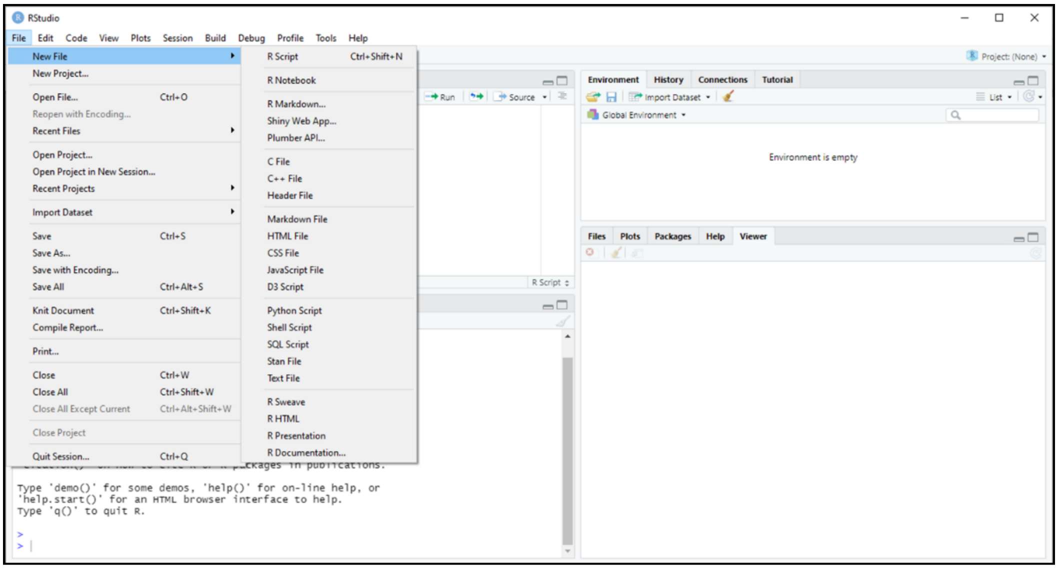
There are various ways for working in R:

- Work directly from the R editor to type in your script and execute the script completely (batch) or line-by-line (highlight and execute)
- Write script in an external editor (Notepad or software that interfaces with R) and execute in R by copy/paste or highlighting
- Beyond the native R GUI, external GUI can work with R to help in writing scripts, selecting functions, procedures, statistical tests, or graphics



```
R                            -- "Bunny-Wunnies Freak Out"
Co                           ation for Statistical Computing
Pl                           64 (64-bit)

R                            ith ABSOLUTELY NO WARRANTY.
Yo                           e it under certain conditions.
Ty                           ' for distribution details.

                             running in an English locale

R                            ith many contributors.
Ty                           information and
'c                           or R packages in publications.

Ty                           'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

Getting started: R



```
                            ckages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
> |
```

Getting started: RStudio

R is an expression language with a very simple syntax. It is case sensitive as are most UNIX based packages. For example, A and a are different symbols and refer to different variables. The set of symbols which can be used in R names depends on the operating system and country within which R is being run (technically on the locale in use). Normally all alphanumeric symbols are allowed (and in some countries this includes accented letters) plus '.' and '_', with the restriction that a name must start with '.' or a letter, and if it starts with '.' the second character must not be a digit. Elementary commands consist of either expressions or assignments. If an expression is given as a command, it is evaluated, printed (unless specifically made invisible), and the value is lost. An assignment evaluates an expression and passes the value to a variable but the result is not automatically printed. Commands are separated either by a semi-colon (';'), or by a newline. Elementary commands can be grouped together into one compound expression by braces ('{' and '}'). Comments can be put almost anywhere, starting with a hashmark ('#'), everything to the end of the line is a comment. If a command is not complete at the end of a line, R will give a different prompt, by default + on second and subsequent lines and continue to read input until the command is syntactically complete.

**R Workspace**

R workspace is temporary space on your CPU's RAM that "disappears" at the end of R session. It includes any user-defined objects (vectors, matrices, data frames, lists, functions). All data, analyses, output are stored as objects in the R workspace. This workspace is not saved on disk unless you tell R to do so. This means that your objects are lost when you close R and not save the objects, or worse when R or your system crashes on you during a session. When you close the RGui or the R console window, the system will ask if you want to save the workspace image. If you select to save the workspace image then all the objects in your current R session are saved in a file ".RData". ".RData" is a binary file located in the working directory of R, which is by default the installation directory of R. During your R session, you can also explicitly save the workspace image.

Go to the 'Session' menu and then select 'Save Workspace as'

```
> save.image("example1.Rdata")
```

If you have saved a workspace image and you start R the next time, it will restore the workspace. So all your previously saved objects are available again.

Go to the 'Session' menu and then select 'Load Workspace'.

```
> load.image("example1.Rdata")
```

- Windows uses a \ (left slash) to delineate locations in CPU:

  C:\Users\hp\Documents

- R uses / (right slash) to delineate locations in CPU:

  C:/Users/hp/Documents

- An alternative to R's / (single right) is \\ (two left) slashes:

  C:\\Users\\hp\\Documents

- There is no issue in the MAC OS/Linux as they have retained the / (right slash) as the basis for directory delineation

- Print the current working directory

  ```
  > getwd()
  ```

- List the objects in the current workspace

  ```
  > ls()
  ```

- Change to my directory

  ```
  > setwd(mydirectory)
  ```

- Display last 25 commands

  ```
  > history()
  ```

- Display all previous commands

  ```
  > history(max.show=Inf)
  ```

- Saving R workspace

  ```
  > x <- 5 # object x; x is assigned value 5
  > y <- 10 # object y; y is assigned value 10
  > z <- x+y # object z (addition of numbers x and y); z is assigned the value x+y
  > save(x, y, file = "example1_xy.RData") # save two specified objects x and y
  > save.image(file = "example1.RData") # save entire workspace
  ```

- Removing objects R workspace: Use rm()

  ```
    > ls()
     [1]    "x" "y" "z"
    > rm(x, y) # removes objects x and y
    > ls()
    [1] "z"
  ```

- Use load() to add previously saved objects or workspaces to your current R session.

  ```
    > load(file = "example1.RData")
    > ls()
    [2]  "x" "y" "z"
  ```

## Getting help with functions and features

To get more information on any specific named function, use help() function or ? help operator.

```
        > help(lm) or > help(“lm”)
        > ?lm
```

For a feature specified by special characters, the argument must be enclosed in double or single quotes, making it a “character string”. This is also necessary for a few words with syntactic meaning including if, for and function.

```
> help("[[")
```

The convention is to use double quote marks for preference.

On most R installations help is available in HTML format by running help.start() which will launch a Web browser that allows the help pages to be browsed with hyperlinks. The help.search command (alternatively ??) allows searching for help in various ways.

```
          > help.search("lm")
          > ??lm
```

The examples on a help topic can normally be run by

```
          > example(lm)
```

Windows versions of R have other optional help systems: Use ?help for further details.

## R Datasets

R comes with a number of sample datasets that you can experiment with. One has to type data( ) to see the available datasets. The results will depend on which packages you have loaded. For getting details on a sample dataset, type help(datasetname). Example: > help("AirPassengers")

## R Packages

One of the strengths of R is that the system can easily be extended. The system allows you to write new functions and package those functions in a so called `R package' (or `R library’). The R package may also contain other R objects, for example data sets or documentation. There is a lively R user community and many R packages have been written and made available on CRAN for other users. For example, there are packages for statistics, bioinformatics and many more. To attach package to the system you can use the menu or the library function.

- Via the menu in RGui: Select the ‘Packages’ menu and select ‘load package...’, a list of available packages on your system will be displayed. Select one and click ‘OK’.
- Via the library function: > library( )

## Data Management

Everything in R is an object. An object is simply a data structure that has some methods and attributes. The data elements in any R object has attributes. These attributes describe the nature of the elements. Object attributes are modes, class and types.

- **Modes:** logical (TRUE, FALSE), numeric, character (string), complex (complex number)
- **Type** (e.g. vectors can be character, numeric, logical or complex)
- **Class:** Describes object type and mode of object or element that is specified.

**Objects in R:**

- **Scalar**: a single number (1x1 vector)
- **Vector**: all elements of the same type (Type: logical, character, numeric or complex)
- **List**: can contain objects of different types
- **Matrix**: table of vectors, where all elements are numeric (or complex)
- **Data frame**: table of number and/or character vectors. Can contain lists, too.

Data objects in R can exist in many different modes, classes, and types. mode( ) function returns the mode of an object. Some object classes like arrays and matrices require all elements to be of the same mode. A vector can have only mode type of elements. It can have only numeric, character, logical or complex elements. Other objects (data frames, lists) allow for different modes to exist, i.e. objects within data frames and lists can be of different modes. Class describes object type and mode of object or element that is specified. class( ) function returns class of an object.
Examples: “vector”, “data.frame”, “numeric”, “factor”

```
> z <- 0:9
> z
 [1] 0 1 2 3 4 5 6 7 8 9
```

```
> digits <- as.character (z)
> digits
[1] "0" "1" "2" "3" "4" "5" "6" "7" "8" "9"
> d <- as.integer (digits)
> d
[1] 0 1 2 3 4 5 6 7 8 9
> class (z)
[1] "integer"
> class (digits)
[1] "character"
> class (d)
[1] "integer"
```

## **Vector Arithmetic**

<- the arrow is the assignment symbol, used to assign a value or function to a symbol or object. The ‘=’ operator can be used as an alternative.

```
> 5+10
[1] 15
> x <- 5 # object x; x is assigned value 5
> y <- 10 # object y; y is assigned value 10
> z <- x+y # object z; z is assigned the value x+y
> z # Display z
[1] 15
> sqrt(z)
[1] 3.872983
 > ls() # List objects
 [1] "x" "y" "z"
```

Here, x, y and z are scalar objects, each having a single value.

### ***Assignment statement using* c() *function***

```
> x <- c(9.5, 10.8, 2.5, 3.9, 19.6)
> x
[1]  9.5 10.8  2.5  3.9 19.6
> assign("x", c(9.5, 10.8, 2.5, 3.9, 19.6))
> x
```

```
[1]  9.5 10.8  2.5  3.9 19.6
> c(9.5, 10.8, 2.5, 3.9, 19.6) -> x
> x
[1]  9.5 10.8  2.5  3.9 19.6
> 1/x
[1] 0.10526316 0.09259259 0.40000000 0.25641026 0.05102041
> y <- c(x, 1, 0, 1, x)
> y
 [1]  9.5 10.8  2.5  3.9 19.6  1.0  0.0  1.0  9.5 10.8  2.5  3.9 19.6
```

***The elementary arithmetic operators:***

- +, -, *, / and ^
- log, exp, sin, cos, tan, sqrt
- max and min select the largest and smallest elements of a vector respectively.
- range is a function whose value is a vector of length two, namely c(min(x), max(x)).
- length(x) is the number of elements in x.
- sum(x) gives the total of the elements in x.
- prod(x) gives the product.

```
> x <- c(1:10)
> x
[1]  1  2  3  4  5  6  7  8  9 10
> x [x>6]
[1]  7  8  9 10
> x [(x>6) | (x<4)]
[1]  1  2  3  7  8  9 10
> x <- seq (1,10)
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> rev (x) # reverse order
 [1] 10  9  8  7  6  5  4  3  2  1
> x <- (1:4)^2
> x
[1]  1  4  9 16
```

### *Missing values*

Arithmetic functions on missing values yield missing values.

```
> x <- c(1, 5, 4, NA, 6)
> x
[1]  1  5  4 NA  6
> mean(x)
[1] NA
> mean(x, na.rm = TRUE)
[1] 4
```

The function is.na(x) gives a logical vector of the same size as x with value TRUE if and only if the corresponding element in x is NA.

```
> is.na(x)
[1] FALSE FALSE FALSE  TRUE FALSE
```

Impossible values (e.g., dividing by zero) are represented by the symbol NaN (Not a Number).

```
> 5/0
[1] Inf
> 0/0
[1] NaN
> Inf - Inf
[1] NaN
```

is.na(xx) is TRUE both for NA and NaN values.

is.nan(xx) is only TRUE for NaNs.

```
> color <-c("red", "green", "blue")
> color # the values of character variable color are red, green and blue
[1] "red"   "green" "blue"
> cat(color) # remove quotation marks
red green blue
> cat(color[1])
red
```

### *Assign names to the Elements*

```
> x <- c(Delhi="red", Mumbai="green", Kolkata="blue")
> x
  Delhi  Mumbai Kolkata
  "red" "green"  "blue"
```

```
> names(x)
[1] "Delhi"   "Mumbai"  "Kolkata"
> fruit <- c(2, 3, 6)
> names(fruit) <- c("orange", "apple", "banana")
> fruit
orange  apple banana
     2      3      6
> fruit[c("apple","orange")]
 apple orange
     3      2
> Fruit <- c(orange=2, apple=3, banana=6)
> Fruit
orange  apple banana
           2      3      6
```

All elements of a vector must have the same type. If you concatenate vectors of different types, they will be converted to the least "restrictive" type.

```
> c(2, "car")
[1] "2"   "car”
```

Logical values are converted to 0 / 1 OR "TRUE"/ "FALSE".

```
> c(FALSE, 5)
[1] 0 5
> c(FALSE, "red")
[1] "FALSE" "red"
```

***Background in Vector Arithmetic*:** Vector addition required the vectors to be the same length (dimension).

```
x <- c(9, 2)
> x
[1] 9 2
> y <- c(5, 1)
> y
[1] 5 1
> x + 5
[1] 14  7
> x + y
```

```
[1] 14  3
> x - y
[1] 4 1
> x*y
[1] 45  2
> 2*x+y+5
[1] 28 10
> x/y
[1] 1.8 2.0
```

***Concatenate* – c()**

**c(x, y)**

```
> z <- c(6, 4, 1, 0)
> z
[1] 6 4 1 0
> x <- c(6, 4)
> x
[1] 6 4
> y <- c(1, 0)
> y
[1] 1 0
> z <- c(x, y)
> z
[1] 6 4 1 0
```

***Generating regular sequences – seq()***

```
> x1 <- 1:10
> x1
 [1]  1  2  3  4  5  6  7  8  9 10
> x2 <- seq(1, 10)
> x2
 [1]  1  2  3  4  5  6  7  8  9 10
> x3 <- seq(1, 10, by = 2)
> x3
[1] 1 3 5 7 9
> x4 <- seq(10, 22, length = 5)
> x4
[1] 10 13 16 19 22
> x5 <- seq(length = 31, from = -5, by = 3)
> x5
 [1] -5 -2  1  4  7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70
73
[28] 76 79 82 85
```

### *Generating regular sequences* – rep()

Replicate or repeat

```
    > x6 <- rep(3, 5)
    > x6
    [1] 3 3 3 3 3
    > x7 <- 1:3
    > x7
    [1] 1 2 3
    > x8 <- rep(x7, times = 5) # put five copies of x7 end-to-end in x8
    > x8
     [1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
    > x9 <- rep(x7, each = 5) # repeats each element of x7 five times before moving
    on to the next
    > x9
     [1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
```

### *Summaries and Subscripting*

```
> x <- c(1, 3, 4, 7, 11, 32)
> x[1:3]
[1] 1 3 4
> x[c(1:3, 6)]
[1]  1  3  4 32
> x[-(1:4)]
[1] 11 32
> mean(x) # Mean
[1] 9.666667
> m1 <- sum(x)/length(x)
> m1
```

```
[1] 9.666667
> var(x) # Variance
[1] 131.8667
> sum((x-m1)^2)/(length(x)-1)
[1] 131.8667
> sd(x) # Standard deviation
[1] 11.48332
> sqrt(sum((x-m1)^2)/(length(x)-1))
[1] 11.48332
> summary(x) # Summary
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   3.250   5.500   9.667  10.000  32.000
> summary(x[1:4]) # Summary
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    2.50    3.50    3.75    4.75    7.00
```

## Matrices

Matrices or more generally arrays are multi-dimensional generalizations of vectors. In fact, they are vectors that can be indexed by two or more indices.

```
> X <- matrix(1:12, nrow = 3, ncol = 4)
> X
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> dim(X)
[1] 3 4
> Y <- matrix(1:12, nrow = 3, ncol = 4, byrow = TRUE)
> Y
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8
[3,]    9   10   11   12
```

***Assigning names to rows and columns***

```
> rownames(X) <- c("A", "B", "C")
> X
  [,1] [,2] [,3] [,4]
A    1    4    7   10
B    2    5    8   11
C    3    6    9   12
> colnames(X) <- c("X1", "X2", "X3", "X4")
> X
  X1 X2 X3 X4
A  1  4  7 10
B  2  5  8 11
C  3  6  9 12
```

***Accessing elements of a matrix***

```
> X
  X1 X2 X3 X4
A  1  4  7 10
B  2  5  8 11
C  3  6  9 12
> X[,1]
A B C
1 2 3
> X[1,]
X1 X2 X3 X4
 1  4  7 10
> X[2, 3]
[1] 8
```

***Adding additional rows or binding matrices*** – rbind()

***Adding additional columns or binding matrices*** – cbind()

```
> X <- matrix(1:12, nrow = 3, ncol = 4)
> X
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
> rbind(X, c(5, 1, 2, 6))
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
```

```
[2,]    2    5    8   11
[3,]    3    6    9   12
[4,]    5    1    2    6
> cbind(X, c(3, 4, 9))
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    4    7   10    3
[2,]    2    5    8   11    4
[3,]    3    6    9   12    9
```

Transpose – t(); Determinant – det(); Inverse – solve()

```
> X <- matrix(c(1, 3, 8, 12), nrow = 2, byrow = TRUE)
> X
     [,1] [,2]
[1,]    1    3
[2,]    8   12
> t(X) # Transpose of matrix
     [,1] [,2]
[1,]    1    8
[2,]    3   12
> det(X) # Determinant of matrix
[1] -12
> solve(X) # Inverse of matrix
           [,1]        [,2]
[1,] -1.0000000  0.25000000
[2,]  0.6666667 -0.08333333
```

### List and Data Frame

An R list is an object consisting of an ordered collection of objects known as its components.

```
> Lst <- list(name="Fred", wife="Mary", no.children=3, child.ages=c(4,7,9))
> Lst
$name
[1] "Fred"
$wife
[1] "Mary"
$no.children
```

```
[1] 3
$child.ages
[1] 4 7 9
> length(Lst) # Length
[1] 4
> names(Lst) # Names
[1] "name"        "wife"        "no.children" "child.ages“
 > Lst$no.children
[1] 3
> Lst[[3]]
[1] 3
```

A data frame object in R has similar dimensional properties to a matrix but it may contain categorical data, as well as numeric (mixed modes). The standard layout is to put data for one observation across a row and variables as columns. Columns can be thought of as vectors, being either numeric or character. Columns can have column names, similar to variable names. Column names can be of any length, consisting of letters, numbers and a period (.) if desired. Underscores are not allowed. Column names must start with a letter. Columns (vectors) in a data.frame must be of the same length. On one level, as the notation will reflect, a data frame is a list. Each component corresponds to a variable, i.e., the vector of values of a given variable for each sample. Therefore, a data frame is like a list with components as columns of table. Lists have columns of the same lengths.
A list can be made into a data.frame:

- ✓ Components must be vectors (numeric, character, logical) or factors.
- ✓ All vectors and factors must have the same lengths.

Matrices and even other data frames can be combined with vectors to form a data frame if the dimensions match up.

```
> students <- data.frame(gender = c("F", "M","F"), ht = c(170, 188.5, 168.3), wt =
c(91.8,90, 82.6))
> students
  gender    ht   wt
1      F 170.0 91.8
2      M 188.5 90.0
3      F 168.3 82.6
> students[1, 2]  # Identify the row 1, col 2 element in object Students
```

```
[1] 170
> names(students) # Identify the column names in object Students
[1] "gender" "ht"     "wt"
> rownames(students) <- c("S1", "S2", "S3") # Apply row names to object Students
> students
  gender    ht   wt
S1      F 170.0 91.8
S2      M 188.5 90.0
S3      F 168.3 82.6
```

**Lists**

Lists **c**ombine a collection of objects into a larger composite object.

```
> intake.pre <- c(23,35,34,13,46, 45,34)
> intake.post <- c(56,57,36,58,36,67,32)
> mylist <- list(before=intake.pre, after=intake.post)
> mylist
$before
[1] 23 35 34 13 46 45 34
$after
[1] 56 57 36 58 36 67 32
> mylist[1]
$before
[1] 23 35 34 13 46 45 34
> mylist[[1]]
[1] 23 35 34 13 46 45 34
> dat <- data.frame(intake.pre, intake.post)
> dat
  intake.pre intake.post
1         23          56
2         35          57
3         34          36
4         13          58
5         46          36
6         45          67
7         34          32
> dat$intake.pre
```

```
[1] 23 35 34 13 46 45 34
> dat$intake.pre[3]
[1] 34
> dat$intake.pre[c(1,3)]
[1] 23 34
> dat$intake.pre[-3]
[1] 23 35 13 46 45 34
```

**Factor**

Factors are the data objects which are used to categorize the data and store it as levels. They can store both strings and integers. They are useful in the columns which have a limited number of unique values such as gender (Male, Female), etc.

```
factor(x = character(), levels, labels = levels, ordered = is.ordered(x))
> gender <- c("male","male","female","female","male","female","male")
> gender
[1] "male"   "male"   "female" "female" "male"   "female" "male"
> class(gender)
[1] "character“
> gender <- factor(gender)
> gender
[1] male   male   female female male   female male
Levels: female male
> class(gender)
[1] "factor"
```

**Two-way Layout**

Consider our two-way layout problem, where we produced the indicator variables using rep(). A better way to do this is using the function *gl*, which will generate factors.

```
> clevels <- gl(3,8)
> clevels
 [1] 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3
+ 3
Levels: 1 2 3
> rlevels <- gl(4,2,length=24)
> rlevels
```

```
[1] 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4 4 1 1 2 2 3 3 4
+ 4
Levels: 1 2 3 4
```

Use the function expand.grid to produce a data frame with the desired factors.

```
> reps
[1] 1 2
> colLevels <- 1:3
> colLevels
[1] 1 2 3
> rowLevels <- 1:4
> rowLevels
[1] 1 2 3 4
> height = seq(60, 80, 10)
> height
[1] 60 70 80
> weight = seq(100, 200, 50)
> weight
[1] 100 150 200
> sex = c("Male","Female")
> sex
[1] "Male"   "Female"
```

### Generating Random Numbers

As a language for statistical analysis, R has a comprehensive library of functions for generating random numbers from various statistical distributions.

Example: Generate 5 random integers between 1 and 10

```
> set.seed (100) # function in R used to reproduce results
> sample (1:10, 5) # sampling without
 replacement is the default
[1] 10  7  6  3  1
> sample (1:10, 5, replace = TRUE)
[1] 10  7  6  6  4
> sample (c("H","T"),5, replace = TRUE)
[1] "H" "T" "T" "H" "H"
> runif (5, 0, 1) # generating between 0 and 1, excluding 0 and 1
[1] 0.6902905 0.5358112 0.7108038 0.5383487 0.7489722
```

```
> rnorm (5, 1, 3) # generating random numbers from normal dist with (1,3)
[1]  0.3950981  3.2195215  1.3701385  0.9120499 -0.1665627
```

**Importing Data**

```
> mydata <- read.table ("mydata.txt", header=TRUE) # From Text file
> head(mydata)
  Height        Weight         Sex
1     60        100            Male
2     70        100            Male
3     80        100            Male
4     60        150            Male
5     70        150            Male
6      80       150            Male
> mydata <- read.table ("mydata.csv", header=TRUE) # From CSV file
> mydata <- read.delim ("mydata.csv") # Importing file with a separator character
> mydata <- read.delim2("mydata.csv")
```

***Importing from Excel*****:** Importing from 1st worksheet

We will require a package named ‘xlsx’.

```
> library(xlsx)
Warning message:
package ‘xlsx’ was built under R version 4.0.5
> mydata <- read.xlsx("mydata.xlsx", 1)
```

***Importing SPSS***

```
library(foreign)
mydata <- read.spss(“mydata.sav”, to.data.frame=TRUE,
use.value.labels=FALSE)
```

***Importing SAS files***

```
library(sas7bdat)
mydata <- read.sas7bdat(“mydata.sas7bdat”)
```

***Importing Minitab files***

```
library(foreign)
mydata <- read.mtp(“mydata.mtp”)
```

**Descriptive Statistics**

Descriptive statistics investigates the variables separately. Various descriptive statistics can be computed by using in-built R functions as given below.

| **Name of function** | **Use of function** |
|---|---|
| mean | calculates the mean of an input |
| median | calculates the median of an input |
| var | calculates the variance of an input |
| sd | calculates the standard deviation of an input |
| IQR | calculates the interquartile range of an input |
| min | calculates the minimum value of an input |
| max | calculates the maximum of an input |
| range | returns a vector containing the minimum and maximum of all given arguments |
| summary | returns a vector containing a mixture of the above functions (minimum, first quartile, median, mean, third quartile, maximum) |

```
> data(trees)
> head(trees)
  Girth Height Volume
1   8.3     70   10.3
2   8.6     65   10.3
3   8.8     63   10.2
4  10.5     72   16.4
5  10.7     81   18.8
6  10.8     83   19.7

> summary(trees)
     Girth          Height       Volume
 Min.   : 8.30   Min.   :63   Min.   :10.20
 1st Qu.:11.05   1st Qu.:72   1st Qu.:19.40
 Median :12.90   Median :76   Median :24.20
 Mean   :13.25   Mean   :76   Mean   :30.17
 3rd Qu.:15.25   3rd Qu.:80   3rd Qu.:37.30
 Max.   :20.60   Max.   :87   Max.   :77.00
```

```
> mean(trees$Height)
[1] 76
> sd(trees$Height)
[1] 6.371813
> range(trees$Height)
[1] 63 87
```

**Graphics**

**Histogram** plots the frequencies that data appears within certain ranges.

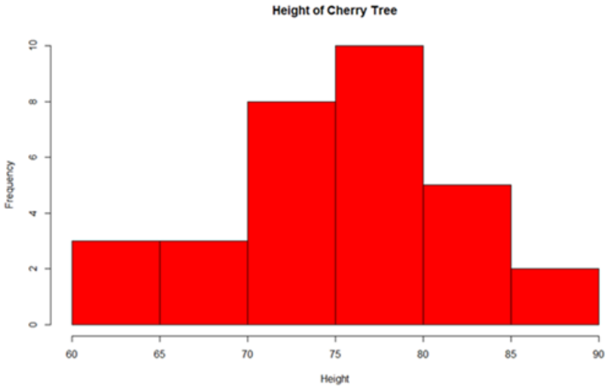
```
> data(trees)
```

***Add a title*****:** The “main” statement will give the plot an overall heading.

***Add axis labels*****:** Use “xlab” and “ylab” to label the X and Y axes, respectively.

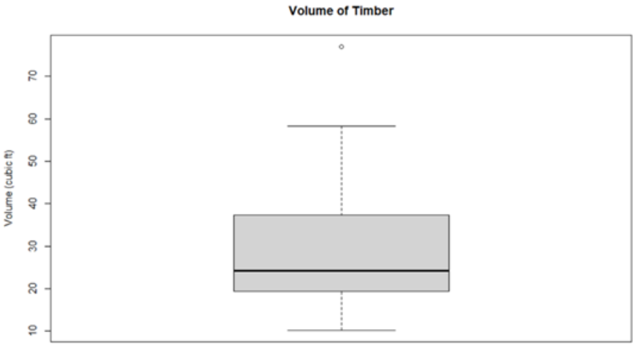
***Changing colors*****:** Use the col statement

```
hist(trees$Height, main="Height of Cherry Tree", xlab="Height",
ylab="Frequency", col="red")
```



A boxplot provides a graphical view of the median, quartiles, maximum, and minimum of a data set.

```
> boxplot(trees$Volume,main='Volume of Timber', ylab='Volume (cubic ft)')
```



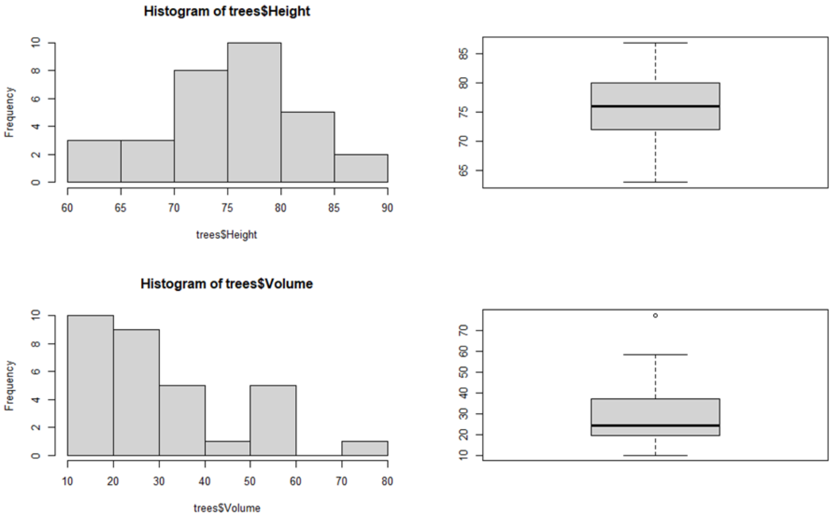
### *Partitioning the Graphics Window*

A useful facility before beginning is to divide a page into smaller pieces so that more than one figure can be displayed graphically.

**par:** used to set or query graphics parameters

```
par(mfrow=c(2,2))
# This will create a window of graphics with 2 rows and 2 columns.
# The windows are filled up row-wise.
# Use mfcol instead of mfrow to fill up column-wise.
```

```
> data(trees)
> par(mfrow=c(2,2))
> hist(trees$Height)
> boxplot(trees$Height)
> hist(trees$Volume)
> boxplot(trees$Volume)
> par(mfrow=c(1,1))
```



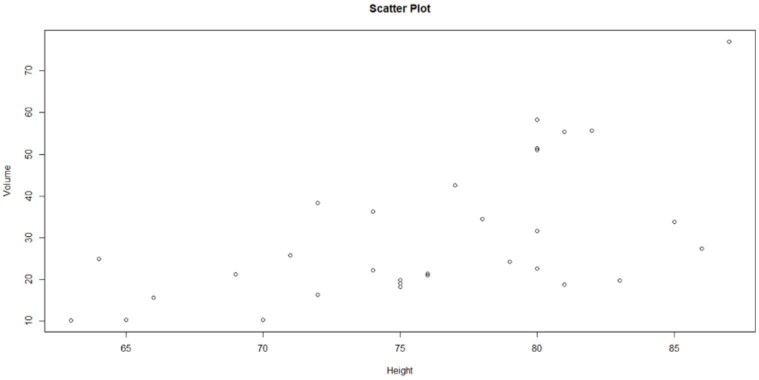
- Use layout()

Example: layout(matrix(1:4,2,2)) will partition the window into 4 equal parts

One can view the layout with layout show (n = 4)

A **scatter plot** provides a graphical view of the relationship between two sets of numbers.

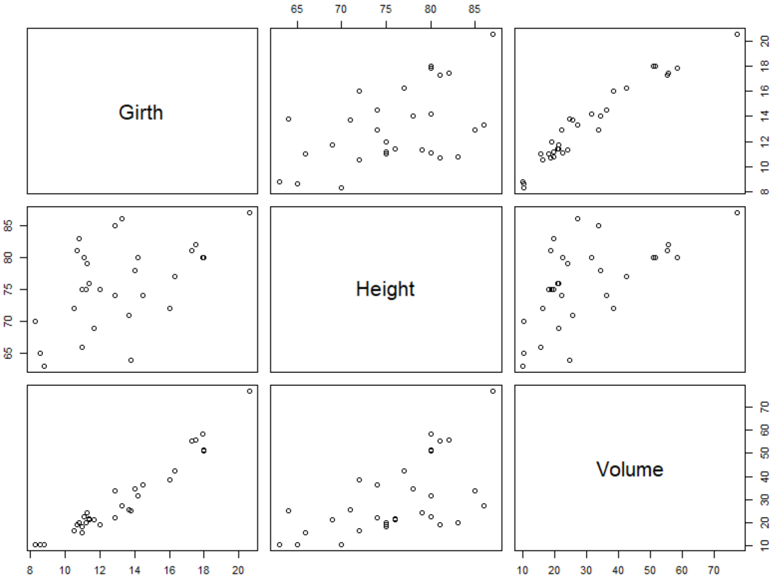
```
> plot(trees$Height, trees$Volume, xlab="Height", ylab="Volume",
main="Scatter Plot", pch=20)
```



parameter pch stands for ‘plotting character’.

```
> pairs(trees)
```

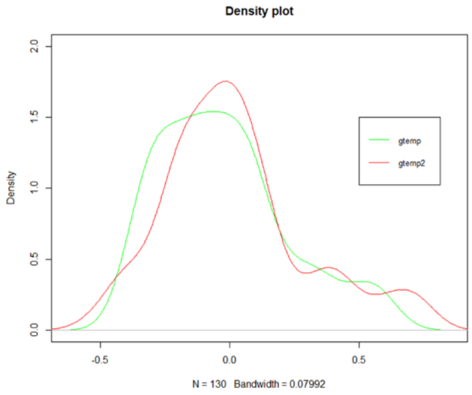
A matrix of scatterplots is produced.



**Density plot** is a representation of the distribution of a numeric variable that uses a kernel density estimate to show the probability density function of the variable.

In R Language we use the density() function which helps to compute kernel density estimates.

```
> plot(density(gtemp), ylim=c(0, 2), col = "green",main = "Density plot")
> lines(density(gtemp2), col="red")
> legend(0.5,1.5, cex=0.8, c("gtemp", "gtemp2"), col=c("green", "red"), lty=1:1)
```



## Writing functions

A function is a set of statements organized together to perform a specific task. R has a large number of in-built functions such as seq(), mean(), max(), sum(), etc. The user can create their own functions.

General form of the function:

```
func_name <- function(arg1, arg2, ...) {
Function body
}
```

func_name is the name of actual name of function.

The argument can be any type of object (like a scalar, a matrix, a data frame, a vector, a logical, etc)

### *Local vs global environment*

It's not necessarily to use return() at the end of your function. The reason you return an object is if you've saved the value of your statements into an object inside the function. In this case, the objects in the function are in a local environment and won't appear in your global environment.

```
fun1 <- function(x){
  2*x+3
}
```

```
> fun1(4)
[1] 11

fun2 <- function(x){
   y <- 2*x+3
}
> fun2(4)
> print(y)
Error in print(y) : object 'y' not found
```

We can return the value of y using return(y) at the end of the function.

```
fun2_1 <- function(x){
   y <- 2*x+3
   return(y)
}
> fun2_1(4)
[1] 11

fun3 <- function(x, y){
  z1 <- 2*x+y
  z2 <- x+2*y
  z3 <- 2*x+2*y
  z4 <- x/y
  return(c(z1, z2, z3, z4))
}
> fun3(1, 2)
[1] 4.0 5.0 6.0 0.5
```

If we need to return multiple objects from a function, we can use list() to list them together. To extract objects from output, use [[ ]] operator.

```
fun4 <- function(x, y){
  m1 <- mean(x)
  m2 <- mean(y)
  sd1 <- sd(x)
  sd2 <- sd(y)
  cor.xy <- cor(x, y)
  xy <- cbind(x, y)
  list(m1, m2, sd1, sd2, cor.xy, xy)
}

> x <- c(1, 4, 8, 11, 20, 23)
> y <- c(2, 6, 3, 8, 21, 29)
> fun4(x, y)
[[1]]
[1] 11.16667
[[2]]
[1] 11.5
[[3]]
[1] 8.750238
[[4]]
[1] 10.96814
[[5]]
[1] 0.9471335
[[6]]
      x  y
[1,]  1  2
[2,]  4  6
[3,]  8  3
[4,] 11  8
[5,] 20 21
[6,] 23 29
```

**for loops**

-The for loop is used when iterating through a list.

-The basic structure of the for loop:

```
for(index in list){
 commands
}

cars <- c("Toyota", "Ford", "Chevy")
for(I in cars) {
  print(i)
}
[1] "Toyota"
[1] "Ford"
[1] "Chevy"
```

**while loop**

The while loop is used when you want to keep iterating as long as a specific condition is satisfied. The basic structure of the while loop:

```
while(condition) {
  commands
}
i <- 3
while(i <= 6) {
  i <- i+1
  print(i)
}
[1] 4
[1] 5
[1] 6
[1] 7
```

**Ifelse function**

The ifelse function is very handy because it allows the user to specify the action taken for the test condition being true or false. Like the if statement the ifelse function can be included in any function or loop.

The basic structure of the ifelse function:

```
Ifelse(test, action.if.true, action.if.false)

> x <- seq(1:10)
> ifelse(x < 6, "T", "F")
[1] "T" "T" "T" "T" "T" "F" "F" "F" "F" "F"
```

**R Packages for Bioinformatics**

R packages are extensions to the R statistical programming language. R packages contain code, data, and documentation in a standardised collection format that can be installed by users of R. A large number of R packages are freely through CRAN (the Comprehensive R Archive Network; https://cran.r-project.org/) and Bioconductor set of R packages (www.bioconductor.org). Some well-known bioinformatics R packages are the Bioconductor set of R packages (www.bioconductor.org). Bioconductor is a free, open source and open development software project for the analysis and comprehension of genomic data.

**R Packages for analysis of biological sequence analysis and retrieval of genomic data**

- seqinr
- tidysq
- biomartr
- rentrez

**R packages for sequence alignment**

- Biostrings
- msa
- msaR
- ggmsa
- AlignStat

**R Packages for differential gene expression analysis of microarray data**

- amda
- maGUI
- maEndToEnd
- limma
- GEOlimma

**R packages for differential gene expression analysis of RNA-Seq data**

- edgeR
- DESeq2
- ideal
- DEvis

**R Packages for protein structure analysis**

- Bio3D
- Rpdb

- XLmap

**R packages for protein-protein interaction graphs**
- graph
- RBGL
- Rgraphviz
- crosstalkr
- igraph

**R Packages for proteomics data analysis**
- RforProteomcs
- protti
- Proteus
- DanteR
- MSstats
- MSqRob
- DAPAR

**R Packages for metagenomics data analysis**
- MicrobiomeExplorer
- matR
- MegaR

**R Packages for GWAS and genomic selection**
- statgenGWAS
- GWASTools
- BlueSNP
- rrBLUP
- lme4GS
- BWGS
- GSelection
- learnMET
- GAPIT

## Demonstration of an R package “GAPIT: Genomic Association and Prediction Integrated Tool”

GAPIT implemented a series of methods for Genome Wide Association (GWAS) and Genomic Selection (GS). The GWAS models include

- General Linear Model (GLM)
- Mixed Linear Model (MLM or Q+K)
- Compressed MLM (CMLM)
- Enriched CMLM
- SUPPER
- Multiple Loci Mixed Model (MLMM)
- FarmCPU
- BLINK

The GS models include

- gBLUP
- Compressed BLUP
- SUPER BLUP

GAPIT is an R package which can be freely downloaded from http://www.r-project.org or http://www.rstudio.com.

There are two sources to install GAPIT package.

***Zhiwu Zhang Lab website***

```
source("http://zzlab.net/GAPIT/GAPIT.library.R")
source("http://zzlab.net/GAPIT/gapit_functions.txt")
```

***GitHub***

```
install.packages("devtools")
devtools::install_github("jiabowang/GAPIT3",force=TRUE)
library(GAPIT3)
```

Help manual: https://zzlab.net/GAPIT/gapit_help_document.pdf

```
# Import data from Zhiwu Zhang Lab
myY <- read.table("http://zzlab.net/GAPIT/data/mdp_traits.txt", head = TRUE)
myGD=read.table(file="http://zzlab.net/GAPIT/data/mdp_numeric.txt",head=T)
myGM=read.table(file="http://zzlab.net/GAPIT/data/mdp_SNP_information.txt",
head=T)
# GWAS
myGAPIT=GAPIT(
  Y=myY[,c(1,2,3)], #fist column is ID
  GD=myGD,
  GM=myGM,
  PCA.total=3,
  model=c("FarmCPU", "Blink"),
  Multiple_analysis=TRUE)
```

**References**

Giorgi, F. M., Ceraolo, C. and Mercatelli, D. (2022). The R Language: An Engine for Bioinformatics and Data Science. *Life (Basel, Switzerland)*, **12(5)**, 648. https://doi.org/10.3390/life12050648

Ihaka, R. and Gentleman, R (1996). R: A Language for Data Analysis and Graphics. *Journal of Computational and Graphical Statistics*, **5**, 299–314. doi: 10.1080/10618600.1996.10474713

W. N. Venables, D. M. Smith and the R Core Team. *An Introduction to R. Notes on R: A Programming Environment for Data Analysis and Graphics*, Version 4.2.2 (2022-10-31), URL: https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf

https://en.wikipedia.org/wiki/R_(programming_language)

https://en.wikipedia.org/wiki/Bioconductor

https://www.cran.r-project.org/

https://www.bioconductor.org/

# Genome-Wide Association Studies

**Soumya Sharma**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

Genome-wide association study (GWAS) is a research strategy to find genetic variations that are statistically linked to a disease or a particular trait. The approach involves scanning the genomes of a large number of individuals in search of genetic variants that are more prevalent in persons with a particular disease or trait than in people without the disease or trait. These genomic variants are often utilised to look for neighbouring variants that are directly responsible for the disease or trait once they have been found.

Linkage disequilibrium (LD) between the markers being studied and the functional polymorphisms of the causal genes is the basis for GWAS. On the chromosome, loci that are physically close to one another are separated by recombination less frequently than loci that are farther apart. Gametic-phase disequilibrium, often known as LD, is the nonrandom connection of alleles at two loci. The SNPs close to the causal locus may have strong LD with the functional polymorphisms and hence be linked to the desired trait. These relationships are discovered through genome-wide association studies, which also highlight the genomic areas that contain the significant SNPs and the relevant genes.

Genome-wide association study (GWAS) attempts to predict association of specific traits (phenotype) with genetic variants (genotype) by statistical analysis at population level. Phenotypic information can be obtained by systematically measuring the phenotype (physical and physiological traits) that can be influenced by various genetic and environmental factors. Individual genotyping is usually done with microarrays for common variations or next-generation sequencing technologies like WES or WGS for rare variants. Due to the current expense of next-generation sequencing, microarray-based genotyping is the most frequently used approach for retrieving genotypes for GWAS. However  resequencing the entire genome has the ability to uncover almost all genetic variations. This genotypic information along with phenotypic data can be analysed to identify the genetic markers (SNPs, SSRs etc.), QTLs or candidate genes associated with a specific trait.

The input files for GWA studies usually include the genotype file i.e., marker information and the phenotype file i.e., trait information and also coded family relations between individuals. Following the data input, producing reliable GWAS results requires meticulous quality control.

**Testing for associations.**

The biometrical model underpins the genetic association theory. Depending on whether the phenotype is continuous (such as plant height, grain yield etc.) or binary (such as the presence or absence of disease), linear or logistic regression models are typically employed in GWAS to test for associations. To account for stratification and eliminate confounding effects from demographic characteristics, covariates such as age, sex, and ancestry are added, with the caveat that this may impair statistical power for binary traits in ascertained samples. Adding an additional individual-specific random effect term to linear or logistic mixed models to account for genetic relatedness among individuals might improve statistical power for genome discovery and boost control for stratification at the expense of increased complexity. Adding an additional individual-specific random effect term to linear or logistic mixed models to account for genetic relatedness between people might boost statistical power for genome discovery and increase control for stratification at the cost of more processing resources. When doing a GWAS, it's important to remember that genotypes of genetic variants that are physically close together aren't independent because they are in linkage disequilibrium; this test dependency should be taken into account as well.

The following equation depicts the linear regression model for testing the association between a marker and a trait:

$$Y \sim X\alpha + Z_s\beta_s + e$$
$$e \sim N(0, \sigma_e^2 I)$$

where, for each individual, Y is a vector of phenotype values, X is a matrix assigning records to phenotypes fixed effect, α is a corresponding vector of  fixed effects sizes (e.g., the mean, population structure effects, and age), $Z_s$ is a vector of genotype values for all individuals at genetic variations, $\beta_s$ is the corresponding fixed effect size of genetic variants, $\sigma_e^2$ measures residual variance and I is an identity matrix.

The underlying assumption is that if the marker will have effect on trait only if it is in linkage disequilibrium with an unseen QTL. The null hypothesis for the study asserts that marker has no effect on the trait, while the alternative hypothesis states that it does have an effect on the trait (as it is in LD with a QTL). If F > $F_{\alpha;\nu1;\nu2}$ where F is the F statistic obtained from the data and $F_{\alpha;\nu1;\nu2}$ is the value from a F distribution at α level of significance and ν1, ν2 degrees of freedom, the null hypothesis is rejected.

There are numerous statistical models to find associations between marker loci and a variety of traits, ranging from simple to highly complex. Accurate decoding of complex traits in diverse population requires more comprehensive statistical models which takes care of false positives arising from family relatedness and population structure, at the same time also keeps in check the number of false negatives due to over correction. Confounding effects due to population structure and kinship among individuals is taken into account by using these covariates in the statistical model. STRUCTURE (Pritchard et al., 2000), PCA (Price et al., 2006), and a discriminant analysis of principal components (DAPC) (Jombart et al., 2010) are methods for determining population organisation by using genetic markers. False positives arising due to common ancestry and family relatedness can be addressed by incorporating kinship matrix into the statistical model. One of the most often used methods for estimating family relatedness among individuals in a diverse population is identity-by-state (Loiselle et al., 1995).

Inclusion of population structure and a kinship matrix as covariates in mixed linear models (MLM) to reduce false positives is a widely used approach. Many MLM-based approaches have been presented since Yu et al. (2006) published the first MLM of association mapping (Zhang et al., 2010; Wang et al., 2014). All of these models are called single-locus models as they do a unidimensional genome scan by examining one marker at a time and then iterate the process for each marker in the dataset. But the true genetic model of complex traits that are governed by multiple loci at the same time cannot be explained by single locus models. Multilocus association mapping models have been suggested as a solution to this problem since they consider the input from all loci at the same time (Wang et al., 2016). One more constraint of MLM based models is increase in number of false negatives due to overfitting which may lead to omission of certain potentially valuable association (Liu et al., 2016). False negatives may arise during multiple comparison adjustments for evaluating statistical significance. Bonferroni correction (Holm, 1979) and false discovery rate (FDR) (Benjamini and Hochberg, 1995) are two commonly used multiple comparison approaches in association mapping for determining the significant threshold. Highly conservative standards can result in a high rate of false negatives. As a result, selection of a proper model and threshold are critical steps in detecting true trait associated markers that may be located inside or in high LD with genes that govern trait variation, while minimizing both false-positive and false-negative associations.

**Statistical models for GWAS**

Some popular models for GWAS include:

(1) analysis of variance (ANOVA)

(2) general linear model with principle component analysis (GLM + PCA) (Price et al., 2006),

(3) MLM with principle component analysis and Kinship matrix for family relatedness estimates (GLM+PCA+K) (Yu et al., 2006)

(4) compressed MLM (Zhang et al., 2010)

(5) enriched compressed MLM (Li et al., 2014)

(6) settlement of MLM under progressively exclusive relationship (SUPER) (Wang et al., 2014)

(7) multiple loci MLM (MLMM) (Segura et al., 2012)

(8) fixed and random model circulating probability unification (FarmCPU) (Liu et al., 2016).

Models from (1) to (6) are single locus models, while (7) and (8) are multilocus models.

Among these popular models of GWAS, the GLM and MLM are said to have a better control of false positives than ANOVA (Price et al., 2006; Yu et al., 2006). The GLM with PCA model is supposed to lower the number of false positives caused by population structure alone (Price et al., 2006). The kinship matrix is included in the MLM with PCA and K model, which is intended to reduce false positives caused by family relatedness (Yu et al., 2006). By controlling false positives, the MLM model is said to perform better than the GLM model alone (Yu et al., 2006). The benefit of MLM model in controlling false positives disappears when complex qualities are connected with population structure with considerable genetic divergence, The MLM approach does a good job of controlling P-value inflation, but it also produces false negatives, making it difficult to identify actual correlations (Zhang et al., 2010). The compressed MLM model (CMLM), which clusters individuals into groups and fits genetic values of groups as random effects in the model, was created to address this challenge (Zhang et al., 2010). When compared to traditional MLM methods, the CMLM method boosts statistical power (Zhang et al., 2010). Another option for dealing with P-value deflation caused by MLM is to adopt a SUPER model, in which just the linked genetic markers are utilised as pseudo–Quantitative Trait Nucleotides (QTNs) to determine kinship, rather than all of the markers (Wang et al., 2014). When a pseudo QTN is associated with the testing marker, it is not included in the kinship analysis. Between the pseudo QTNs and the testing marker, the SUPER model applies an LD threshold. When compared to using total kinship from all markers, this strategy improves statistical power. FarmCPU is a multilocus model that was created to reduce false positives while keeping false negatives to a minimum (Liu et al.,2016). To partially minimise the confusion between testing markers and kinship, the FarmCPU model use a modified MLM method called multiple loci linear mixed model (MLMM), which combines many markers simultaneously as covariates in a stepwise MLM. When compared to other models, this model is said to improve statistical power, computing efficiency, and the capacity to control false positives and false negatives (Liu et al., 2016).

Single-locus models, such as the general linear model (GLM) and the mixed linear model (MLM) require multiple tests that undergo a Bonferroni correction (Bradbury et al., 2007) for multiple comparison adjustments. The typical Bonferroni correction is often too conservative, which results in many important loci associated with the target traits being eliminated because they do not satisfy the stringent criterion of the significance test. The multi-locus models are better alternatives for GWASs because they do not require the Bonferroni correction, and thus more marker-trait associations may be identified. Recently, several new multi-locus GWAS models, such as multi-locus RMLM (mrMLM, Wang et al., 2016), fast multi-locus random-SNP-effect EMMA (FASTmrEMMA, Wen et al., 2017), and Iterative modified-Sure Independence Screening EM-Bayesian LASSO (ISIS EM-BLASSO, Tamba et al., 2017), have been developed.

### Representation of GWAS Results

GWAS results are typically represented as two types of p-value plots: genome-wide association plots (Manhattan plots) and quantile-quantile (QQ) plots. In Manhattan plot marker loci are represented as chromosomes and position on the chromosome in genomic order on x-axis and negative logarithm of their p values ($-\log_{10}P$) on y-axis (Fig1). The Manhattan plot resembles the Manhattan skyline because clusters of significant P values tend to ascend to the top due to local correlation of the genetic variants brought on by linkage.

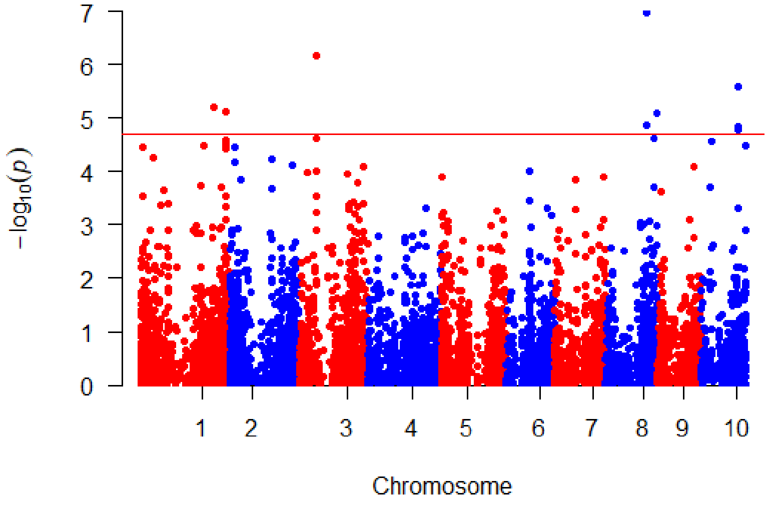


Fig 1: An illustration of a Manhattan plot depicting several strongly associated loci to the trait

Quantile-quantile plots (QQ plots) are widely used to display the proportion of significant results in relation to the projected number of significant results at a specific P value (Fig 2). The figure unambiguously demonstrated that, at levels more than P 0.001, more significant SNP were discovered in their analysis than would have been expected by chance.

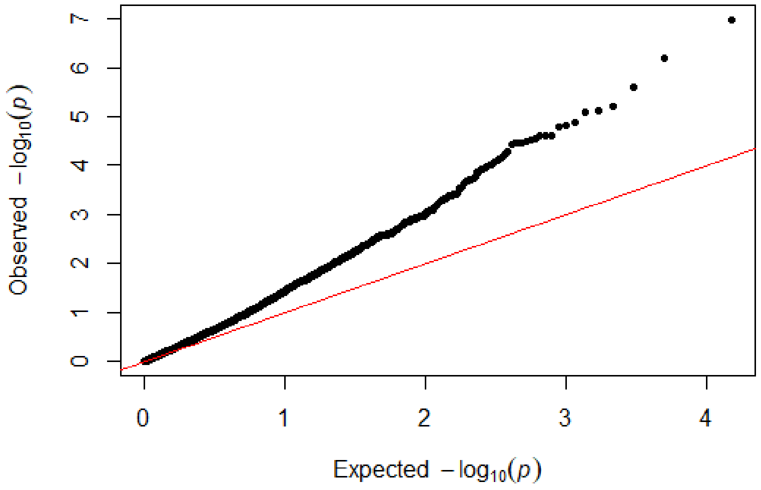


Fig 2: Quantile-quantile (QQ) plot. Comparison of GWAS P-values (black dotted line) to those expected for a null distribution (red line).

## References:

Benjamini, Y., Hochberg, Y. (1995). Controlling the false discovery rate, A practical and powerful approach to multiple testing. *J. R. Stat. Soc. Series. B. Stat. Methodol.* 57, 289–300. doi: 10.1111/j.2517-6161.1995.tb02031.x

Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. A., et al. (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nat. Genet.* 38 (8), 904–909. doi: 10.1038/ng1847

Zhang, Z., Ersoz, E., Lai, C. Q., Todhunter, R. J., Tiwari, H. K., Gore, M. A., et al. (2010). Mixed linear model approach adapted for genome-wide association studies. *Nat. Genet.* 42, 355–360. doi: 10.1038/ng.546

Kaler, A. S., Gillman, J. D., Beissinger, T., & Purcell, L. C. (2020). Comparing different statistical models and multiple testing corrections for association mapping in soybean and maize. Frontiers in plant science, 10, 1794.

Yu, J., Pressoir, G., Briggs, W. H., Vroh, B. I., Yamasaki, M., Doebley, J. F., et al. (2006). A unifed mixed-model method for association mapping that accounts for multiple levels of relatedness. *Nat. Genet.* 38, 203–208. doi: 10.1038/ng1702

Liu, X., Huang, M., Fan, B., Buckler, E. S., Zhang, Z. (2016). Iterative usage of fixed and random effect models for powerful and efficient genome-wide association studies. *PLoS Genet.* 12 (2), e1005767. doi: 10.1371/journal.pgen.1005767

Li, M., Liu, X., Bradbury, P., Yu, J., Zhang, Y.-M., Todhunter, R. J., et al. (2014). Enrichment of statistical power for genome-wide association studies. *BMC Biol.* 12, 73. doi: 10.1186/s12915-014-0073-5

Wang, Q., Tian, F., Pan, Y., Buckler, E. S., Zhang, Z. (2014). A SUPER powerful method for genome wide association study. *PLoS ONE* 9, e107684. doi: 10.1371/journal.pone.0107684

Segura, V., Vilhjálmsson, B. J., Platt, A., Korte, A., Seren, Ü., Long, Q., et al. (2012). An efficient multi-locus mixed-model approach for genome-wide association studies in structured populations. *Nat. Genet.* 44, 825–830. doi: 10.1038/ng.2314

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scand. J. Stat.* 6, 65–70.

Wen, Y. J., Zhang, H., Ni, Y. L., Huang, B., Zhang, J., Feng, J. Y., et al. (2018). Methodological implementation of mixed linear models in multi-locus genome-wide association studies. *Brief. Bioinform.* 19, 700–712. doi: 10.1093/bib/bbw145

Tamba, C. L., Ni, Y. L., Zhang, Y. M. (2017). Iterative sure independence screening EM-Bayesian LASSO algorithm for multi-locus genome-wide association studies. *PLoS Comput. Biol.* 13, e1005357. doi: 10.1371/journal.pcbi.1005357

# Hands-on Session for GWAS

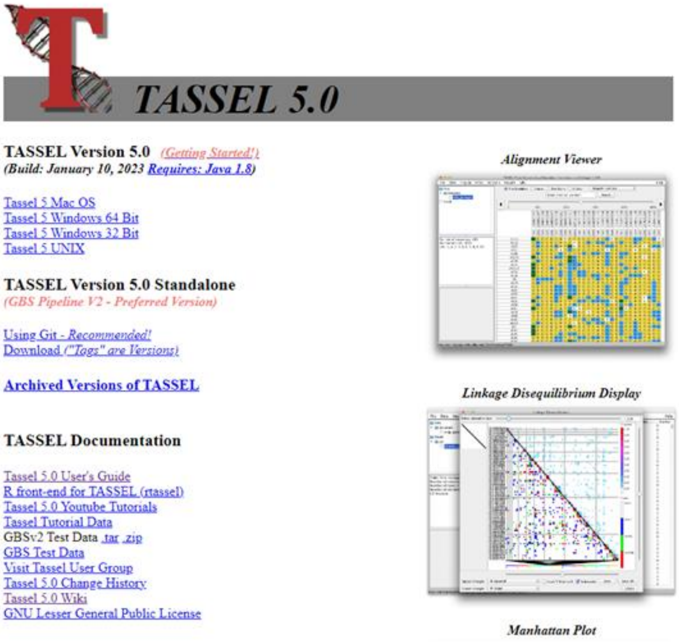
**Soumya Sharma**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

TASSEL also known as Trait Analysis by aSSociation, Evolution and Linkage is a powerful statistical software to conduct association mapping such as General Linear Model (GLM) and Mixed Linear Model (MLM). The GUI (graphical user interface) version of TASSEL is very well built for anyone who does not have a background or experience in working in command line. The following section demonstrates how to prepare input files and run association analysis in TASSEL in stepwise manner.

## 1. Download and install TASSEL software

Download and install the latest version of the TASSEL software at this link: https://www.maizegenetics.net/tassel



TASSEL 5.0

**TASSEL Version 5.0** *(Getting Started!)*
*(Build: January 10, 2023 Requires: Java 1.8)*

Tassel 5 Mac OS
Tassel 5 Windows 64 Bit
Tassel 5 Windows 32 Bit
Tassel 5 UNIX

**TASSEL Version 5.0 Standalone**
*(GBS Pipeline V2 - Preferred Version)*

Using Git - *Recommended!*
Download *("Tags" are Versions)*

**Archived Versions of TASSEL**

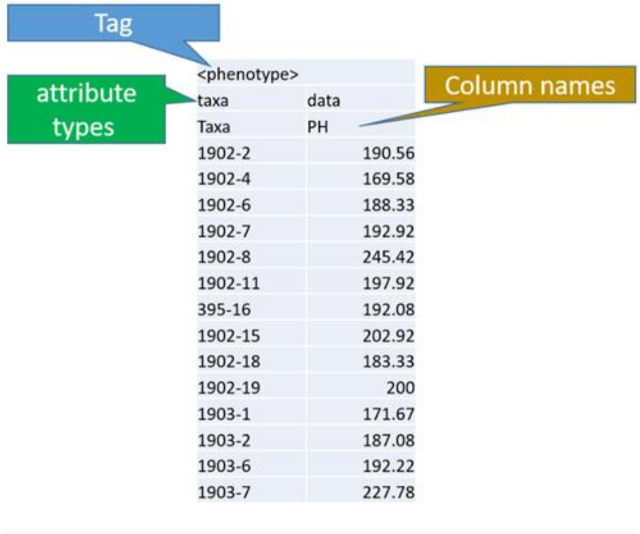
**TASSEL Documentation**

Tassel 5.0 User's Guide
R front-end for TASSEL (rtassel)
Tassel 5.0 Youtube Tutorials
Tassel Tutorial Data
GBSv2 Test Data .tar .zip
GBS Test Data
Visit Tassel User Group
Tassel 5.0 Change History
Tassel 5.0 Wiki
GNU Lesser General Public License

## 2. Preparing the Input files

### Phenotype file

Phenotype file can be prepared as shown below in the figure below

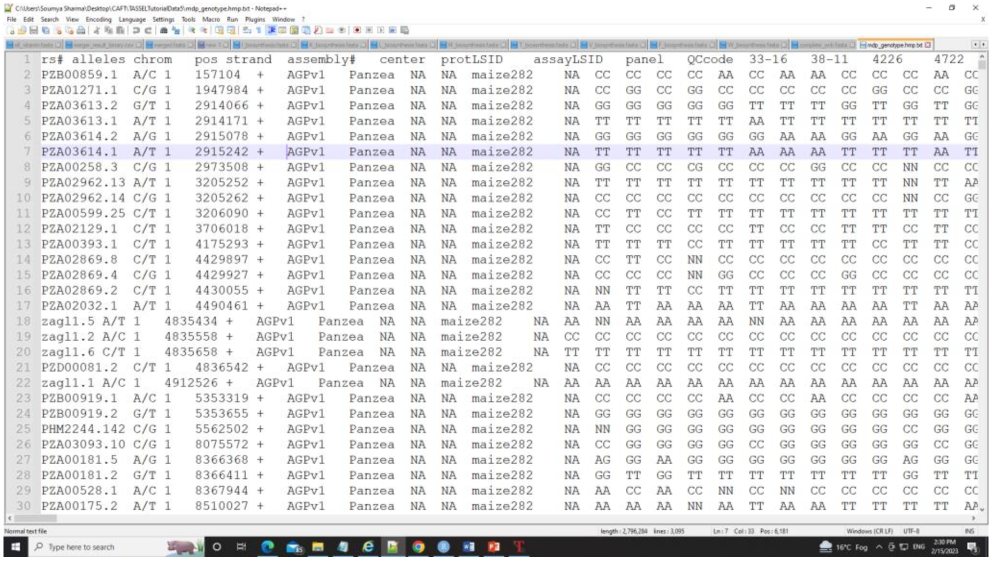

Tag
attribute types
Column names

| <phenotype> | |
|---|---|
| taxa | data |
| Taxa | PH |
| 1902-2 | 190.56 |
| 1902-4 | 169.58 |
| 1902-6 | 188.33 |
| 1902-7 | 192.92 |
| 1902-8 | 245.42 |
| 1902-11 | 197.92 |
| 395-16 | 192.08 |
| 1902-15 | 202.92 |
| 1902-18 | 183.33 |
| 1902-19 | 200 |
| 1903-1 | 171.67 |
| 1903-2 | 187.08 |
| 1903-6 | 192.22 |
| 1903-7 | 227.78 |

Please remember if your data has covariates such as sex, age or treatment, then, please categories them with header name factor.

## Genotype file

TASSEL supports various genotype file formats such as VCF (variant call format), .hmp.txt, and plink. We are using the hmp.txt version of the genotype file for this demonstration. The below screenshot of the hmp.txt genotype file.

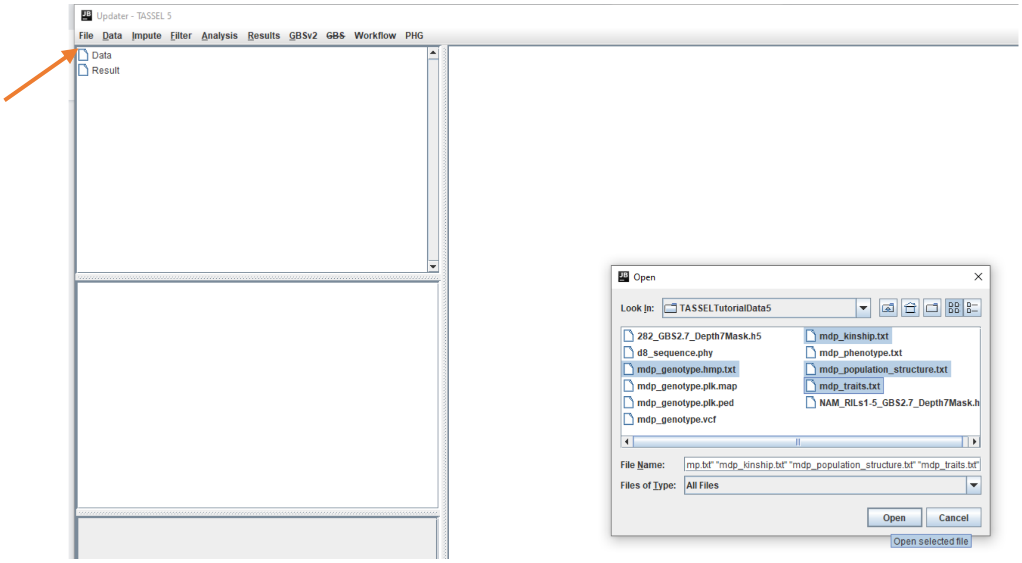

C:\Users\Soumya Sharma\Desktop\CAFT\TASSELTutorialData5\mdp_genotype.hmp.txt - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
mdp_genotype.hmp.txt
Normal text file
length : 2,796,284 lines : 3,095 Ln : 7 Col : 33 Pos : 6,181 Windows (CR LF) UTF-8 INS
Type here to search
16°C Fog ENG 2:30 PM 2/15/2023

```
rs# alleles chrom   pos strand  assembly#   center  protLSID    assayLSID   panel   QCcode  33-16   38-11   4226    4722
PZB00859.1  A/C 1   157104  +   AGPv1   Panzea  NA  NA  maize282    NA  CC  CC  CC  CC  AA  CC  AA  AA  CC  CC  CC  AA  CC
PZA01271.1  C/G 1   1947984 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  GG  CC  GG  CC  CC  CC  CC  CC  GG  CC  CC  GG
PZA03613.2  G/T 1   2914066 +   AGPv1   Panzea  NA  NA  maize282    NA  GG  GG  GG  GG  GG  TT  TT  TT  GG  TT  GG  TT  GG
PZA03613.1  A/T 1   2914171 +   AGPv1   Panzea  NA  NA  maize282    NA  TT  TT  TT  TT  TT  AA  TT  TT  TT  TT  TT  TT  TT
PZA03614.2  A/G 1   2915078 +   AGPv1   Panzea  NA  NA  maize282    NA  GG  GG  GG  GG  GG  GG  AA  AA  GG  AA  GG  AA  GG
PZA03614.1  A/T 1   2915242 +   AGPv1   Panzea  NA  NA  maize282    NA  TT  TT  TT  TT  TT  AA  AA  AA  TT  TT  TT  AA  TT
PZA00258.3  C/G 1   2973508 +   AGPv1   Panzea  NA  NA  maize282    NA  GG  CC  CC  CG  CC  CC  CC  GG  CC  CC  NN  CC  CC
PZA02962.13 A/T 1   3205252 +   AGPv1   Panzea  NA  NA  maize282    NA  TT  TT  TT  TT  TT  TT  TT  TT  TT  TT  NN  TT  AA
PZA02962.14 C/G 1   3205262 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  NN  CC  GG
PZA00599.25 C/T 1   3206090 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  TT  CC  TT  TT  TT  TT  TT  TT  TT  TT  TT  TT
PZA02129.1  C/T 1   3706018 +   AGPv1   Panzea  NA  NA  maize282    NA  TT  CC  CC  CC  CC  TT  CC  CC  TT  TT  CC  TT  CC
PZA00393.1  C/T 1   4175293 +   AGPv1   Panzea  NA  NA  maize282    NA  TT  TT  TT  CC  TT  TT  TT  TT  TT  CC  TT  TT  CC
PZA02869.8  C/T 1   4429897 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  TT  CC  NN  CC  CC  CC  CC  CC  CC  CC  CC  CC
PZA02869.4  C/G 1   4429927 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  CC  CC  NN  GG  CC  CC  CC  GG  CC  CC  CC  CC
PZA02869.2  C/T 1   4430055 +   AGPv1   Panzea  NA  NA  maize282    NA  NN  TT  TT  CC  TT  TT  TT  TT  TT  TT  TT  TT  TT
PZA02032.1  A/T 1   4490461 +   AGPv1   Panzea  NA  NA  maize282    NA  AA  TT  AA  AA  AA  TT  AA  AA  AA  AA  TT  AA  AA
zagl1.5 A/T 1   4835434 +   AGPv1   Panzea  NA  NA  maize282    NA  AA  NN  AA  AA  AA  AA  NN  AA  AA  AA  AA  AA  AA  AA
zagl1.2 A/C 1   4835558 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC
zagl1.6 C/T 1   4835658 +   AGPv1   Panzea  NA  NA  maize282    NA  TT  TT  TT  TT  TT  TT  TT  TT  TT  TT  TT  TT  TT  TT
PZD00081.2  C/T 1   4836542 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC  CC
zagl1.1 A/C 1   4912526 +   AGPv1   Panzea  NA  NA  maize282    NA  AA  AA  AA  AA  AA  AA  AA  AA  AA  AA  AA  AA  AA  AA
PZB00919.1  A/C 1   5353319 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  CC  CC  CC  AA  CC  CC  AA  CC  CC  CC  CC  AA
PZB00919.2  G/T 1   5353655 +   AGPv1   Panzea  NA  NA  maize282    NA  GG  GG  GG  GG  GG  GG  GG  GG  GG  GG  GG  GG  GG
PHM2244.142 C/G 1   5562502 +   AGPv1   Panzea  NA  NA  maize282    NA  NN  GG  GG  GG  GG  GG  GG  GG  GG  GG  CC  GG  GG
PZA03093.10 C/G 1   8075572 +   AGPv1   Panzea  NA  NA  maize282    NA  CC  GG  GG  GG  GG  CC  GG  GG  GG  GG  GG  CC  GG
PZA00181.5  A/G 1   8366368 +   AGPv1   Panzea  NA  NA  maize282    NA  AG  GG  AA  GG  GG  GG  GG  GG  GG  GG  AG  GG  GG
PZA00181.2  G/T 1   8366411 +   AGPv1   Panzea  NA  NA  maize282    NA  GG  TT  GG  TT  TT  TT  TT  TT  TT  TT  GG  TT  TT
PZA00528.1  A/C 1   8367944 +   AGPv1   Panzea  NA  NA  maize282    NA  AA  CC  AA  CC  NN  CC  NN  CC  CC  CC  CC  CC  CC
PZA00175.2  A/T 1   8510027 +   AGPv1   Panzea  NA  NA  maize282    NA  AA  AA  AA  NN  AA  TT  AA  AA  TT  TT  TT  TT  AA
```

## 3. Importing phenotype and genotype files

Import the files by following the steps shown below.

Start Tassel -> go to “file” menu -> select “open” -> specify the “folder” where files are located -> choose the “files” to open holding CTRL button -> click on “open”


Updater - TASSEL 5
File Data Impute Filter Analysis Results GBSv2 GBS Workflow PHG
Data
Result
Open
Open
Cancel
Open selected file

Look In: TASSELTutorialData5

282_GBS2.7_Depth7Mask.h5
d8_sequence.phy
mdp_genotype.hmp.txt
mdp_genotype.plk.map
mdp_genotype.plk.ped
mdp_genotype.vcf
mdp_kinship.txt
mdp_phenotype.txt
mdp_population_structure.txt
mdp_traits.txt
NAM_RILs1-5_GBS2.7_Depth7Mask.h

File Name: mp.txt" "mdp_kinship.txt" "mdp_population_structure.txt" "mdp_traits.txt"

Files of Type: All Files

### 4. Phenotype distribution plot

It is always a wise idea to look at the phenotype distribution by plotting to check for any outliers.

Select the “phenotype” file -> go to “Results” -> go to “Charts” -> select graph type as “Histogram” -> select the trait under “Series 1”

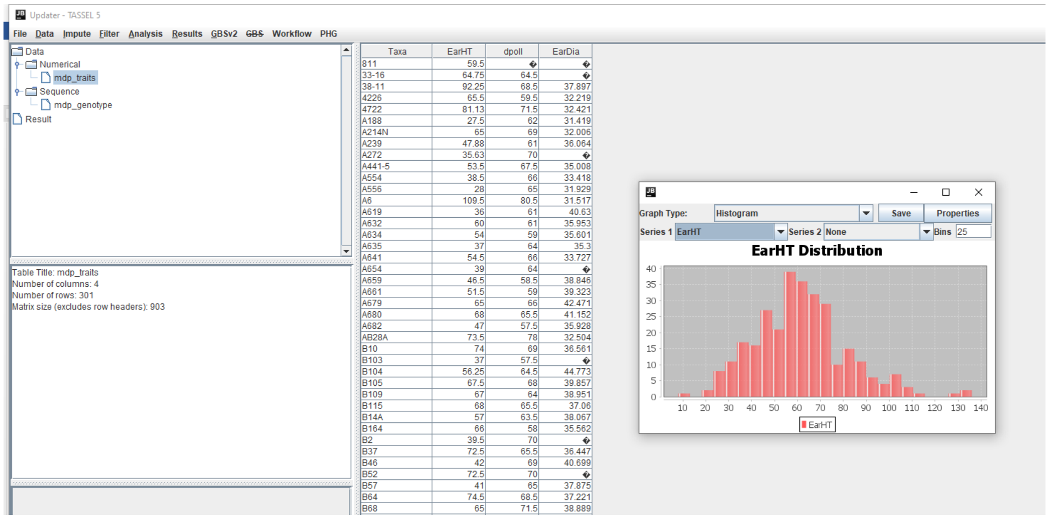


Table Title: mdp_traits
Number of columns: 4
Number of rows: 301
Matrix size (excludes row headers): 903

| Taxa | EarHT | dpoll | EarDia |
|---|---|---|---|
| 811 | 59.5 | � | � |
| 33-16 | 64.75 | 64.5 | � |
| 38-11 | 92.25 | 68.5 | 37.897 |
| 4226 | 65.5 | 59.5 | 32.219 |
| 4722 | 81.13 | 71.5 | 32.421 |
| A188 | 27.5 | 62 | 31.419 |
| A214N | 65 | 69 | 32.006 |
| A239 | 47.88 | 61 | 36.064 |
| A272 | 35.63 | 70 | � |
| A441-5 | 53.5 | 67.5 | 35.008 |
| A554 | 38.5 | 66 | 33.418 |
| A556 | 28 | 65 | 31.929 |
| A6 | 109.5 | 80.5 | 31.517 |
| A619 | 36 | 61 | 40.63 |
| A632 | 60 | 61 | 35.953 |
| A634 | 54 | 59 | 35.601 |
| A635 | 37 | 64 | 35.3 |
| A641 | 54.5 | 66 | 33.727 |
| A654 | 39 | 64 | � |
| A659 | 46.5 | 58.5 | 38.846 |
| A661 | 51.5 | 59 | 39.323 |
| A679 | 65 | 66 | 42.471 |
| A680 | 68 | 65.5 | 41.152 |
| A682 | 47 | 57.5 | 35.928 |
| AB28A | 73.5 | 78 | 32.504 |
| B10 | 74 | 69 | 36.561 |
| B103 | 37 | 57.5 | � |
| B104 | 56.25 | 64.5 | 44.773 |
| B105 | 67.5 | 68 | 39.857 |
| B109 | 67 | 64 | 38.951 |
| B115 | 68 | 65.5 | 37.06 |
| B14A | 57 | 63.5 | 38.067 |
| B164 | 66 | 58 | 35.562 |
| B2 | 39.5 | 70 | � |
| B37 | 72.5 | 65.5 | 36.447 |
| B46 | 42 | 69 | 40.699 |
| B52 | 72.5 | 70 | � |
| B57 | 41 | 65 | 37.875 |
| B64 | 74.5 | 68.5 | 37.221 |
| B68 | 65 | 71.5 | 38.889 |

Graph Type: Histogram

Series 1 EarHT Series 2 None Bins 25

### 5. Genotype summary analysis

Next crucial step is to look at the genotype data by simply following the steps shown.

Select genotype data -> go to “Data” menu -> click “Geno Summary”

The output will be as shown in the figure below. The arrow depicts missing genotypic data to see if it requires to be imputed.

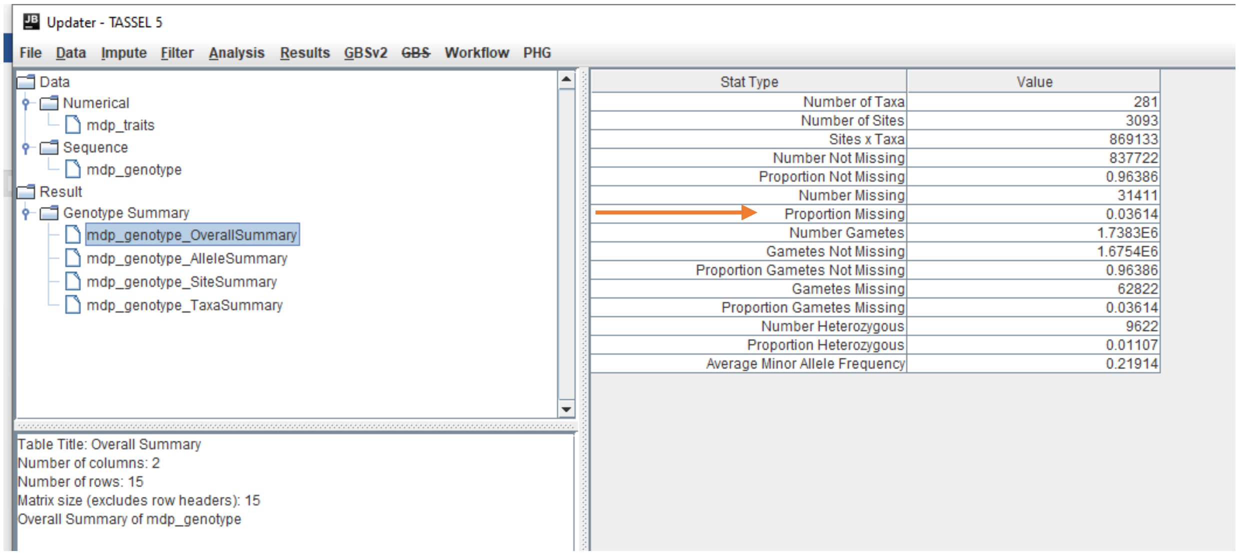


Table Title: Overall Summary
Number of columns: 2
Number of rows: 15
Matrix size (excludes row headers): 15
Overall Summary of mdp_genotype

| Stat Type | Value |
|---|---|
| Number of Taxa | 281 |
| Number of Sites | 3093 |
| Sites x Taxa | 869133 |
| Number Not Missing | 837722 |
| Proportion Not Missing | 0.96386 |
| Number Missing | 31411 |
| Proportion Missing | 0.03614 |
| Number Gametes | 1.7383E6 |
| Gametes Not Missing | 1.6754E6 |
| Proportion Gametes Not Missing | 0.96386 |
| Gametes Missing | 62822 |
| Proportion Gametes Missing | 0.03614 |
| Number Heterozygous | 9622 |
| Proportion Heterozygous | 0.01107 |
| Average Minor Allele Frequency | 0.21914 |

## Minor allele frequency distribution

Select genotype _SiteSummary -> go to “Results” -> click on “Charts” -> select “Minor Allele Frequency” under “Series 1”

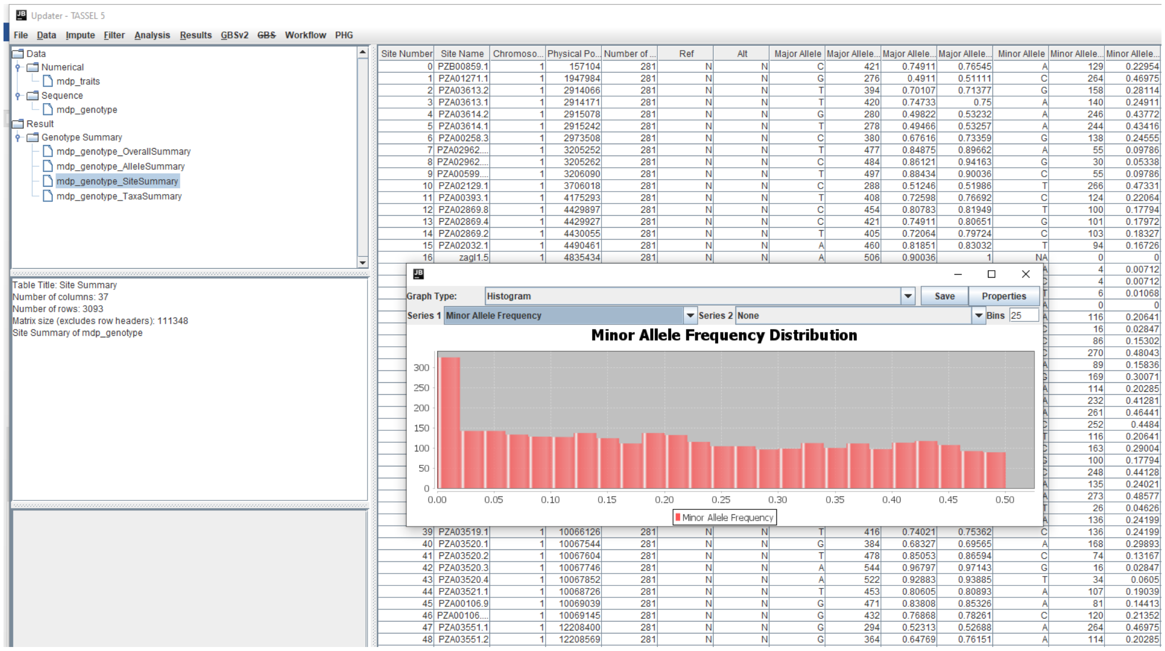


Table Title: Site Summary
Number of columns: 37
Number of rows: 3093
Matrix size (excludes row headers): 111348
Site Summary of mdp_genotype

Graph Type: Histogram
Series 1 Minor Allele Frequency
Series 2 None
Bins 25

| Site Number | Site Name | Chromoso... | Physical Po... | Number of ... | Ref | Alt | Major Allele | Major Allele... | Major Allele... | Major Allele... | Minor Allele | Minor Allele... | Minor Allele... |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | PZB00859.1 | 1 | 157104 | 281 | N | N | C | 421 | 0.74911 | 0.76545 | A | 129 | 0.22954 |
| 1 | PZA01271.1 | 1 | 1947984 | 281 | N | N | G | 276 | 0.4911 | 0.51111 | C | 264 | 0.46975 |
| 2 | PZA03613.2 | 1 | 2914066 | 281 | N | N | T | 394 | 0.70107 | 0.71377 | G | 158 | 0.28114 |
| 3 | PZA03613.1 | 1 | 2914171 | 281 | N | N | T | 420 | 0.74733 | 0.75 | A | 140 | 0.24911 |
| 4 | PZA03614.2 | 1 | 2915078 | 281 | N | N | G | 280 | 0.49822 | 0.53232 | A | 246 | 0.43772 |
| 5 | PZA03614.1 | 1 | 2915242 | 281 | N | N | T | 278 | 0.49466 | 0.53257 | A | 244 | 0.43416 |
| 6 | PZA00258.3 | 1 | 2973508 | 281 | N | N | C | 380 | 0.67616 | 0.73359 | G | 138 | 0.24555 |
| 7 | PZA02962.... | 1 | 3205252 | 281 | N | N | T | 477 | 0.84875 | 0.89662 | A | 55 | 0.09786 |
| 8 | PZA02962.... | 1 | 3205262 | 281 | N | N | C | 484 | 0.86121 | 0.94163 | G | 30 | 0.05338 |
| 9 | PZA00599.... | 1 | 3206090 | 281 | N | N | T | 497 | 0.88434 | 0.90036 | C | 55 | 0.09786 |
| 10 | PZA02129.1 | 1 | 3706018 | 281 | N | N | C | 288 | 0.51246 | 0.51986 | T | 266 | 0.47331 |
| 11 | PZA00393.1 | 1 | 4175293 | 281 | N | N | T | 408 | 0.72598 | 0.76692 | C | 124 | 0.22064 |
| 12 | PZA02869.8 | 1 | 4429897 | 281 | N | N | C | 454 | 0.80783 | 0.81949 | T | 100 | 0.17794 |
| 13 | PZA02869.4 | 1 | 4429927 | 281 | N | N | C | 421 | 0.74911 | 0.80651 | G | 101 | 0.17972 |
| 14 | PZA02869.2 | 1 | 4430055 | 281 | N | N | T | 405 | 0.72064 | 0.79724 | C | 103 | 0.18327 |
| 15 | PZA02032.1 | 1 | 4490461 | 281 | N | N | A | 460 | 0.81851 | 0.83032 | T | 94 | 0.16726 |
| 16 | zagl1.5 | 1 | 4835434 | 281 | N | N | A | 506 | 0.90036 | 1 | NA | 0 | 0 |
|  |  |  |  |  |  |  |  |  |  |  | A | 4 | 0.00712 |
|  |  |  |  |  |  |  |  |  |  |  | C | 4 | 0.00712 |
|  |  |  |  |  |  |  |  |  |  |  | T | 6 | 0.01068 |
|  |  |  |  |  |  |  |  |  |  |  | A | 0 | 0 |
|  |  |  |  |  |  |  |  |  |  |  | A | 116 | 0.20641 |
|  |  |  |  |  |  |  |  |  |  |  | C | 16 | 0.02847 |
|  |  |  |  |  |  |  |  |  |  |  | C | 86 | 0.15302 |
|  |  |  |  |  |  |  |  |  |  |  | C | 270 | 0.48043 |
|  |  |  |  |  |  |  |  |  |  |  | A | 89 | 0.15836 |
|  |  |  |  |  |  |  |  |  |  |  | G | 169 | 0.30071 |
|  |  |  |  |  |  |  |  |  |  |  | A | 114 | 0.20285 |
|  |  |  |  |  |  |  |  |  |  |  | A | 232 | 0.41281 |
|  |  |  |  |  |  |  |  |  |  |  | A | 261 | 0.46441 |
|  |  |  |  |  |  |  |  |  |  |  | C | 252 | 0.4484 |
|  |  |  |  |  |  |  |  |  |  |  | T | 116 | 0.20641 |
|  |  |  |  |  |  |  |  |  |  |  | C | 163 | 0.29004 |
|  |  |  |  |  |  |  |  |  |  |  | G | 100 | 0.17794 |
|  |  |  |  |  |  |  |  |  |  |  | C | 248 | 0.44128 |
|  |  |  |  |  |  |  |  |  |  |  | A | 135 | 0.24021 |
|  |  |  |  |  |  |  |  |  |  |  | A | 273 | 0.48577 |
|  |  |  |  |  |  |  |  |  |  |  | T | 26 | 0.04626 |
|  |  |  |  |  |  |  |  |  |  |  | A | 136 | 0.24199 |
| 39 | PZA03519.1 | 1 | 10066126 | 281 | N | N | T | 416 | 0.74021 | 0.75362 | C | 136 | 0.24199 |
| 40 | PZA03520.1 | 1 | 10067544 | 281 | N | N | G | 384 | 0.68327 | 0.69565 | A | 168 | 0.29893 |
| 41 | PZA03520.2 | 1 | 10067604 | 281 | N | N | T | 478 | 0.85053 | 0.86594 | C | 74 | 0.13167 |
| 42 | PZA03520.3 | 1 | 10067746 | 281 | N | N | A | 544 | 0.96797 | 0.97143 | G | 16 | 0.02847 |
| 43 | PZA03520.4 | 1 | 10067852 | 281 | N | N | A | 522 | 0.92883 | 0.93885 | T | 34 | 0.0605 |
| 44 | PZA03521.1 | 1 | 10068726 | 281 | N | N | T | 453 | 0.80605 | 0.80893 | A | 107 | 0.19039 |
| 45 | PZA00106.9 | 1 | 10069039 | 281 | N | N | G | 471 | 0.83808 | 0.85326 | A | 81 | 0.14413 |
| 46 | PZA00106.... | 1 | 10069145 | 281 | N | N | G | 432 | 0.76868 | 0.78261 | C | 120 | 0.21352 |
| 47 | PZA03551.1 | 1 | 12208400 | 281 | N | N | G | 294 | 0.52313 | 0.52688 | A | 264 | 0.46975 |
| 48 | PZA03551.2 | 1 | 12208569 | 281 | N | N | G | 364 | 0.64769 | 0.76151 | A | 114 | 0.20285 |

## Proportion of heterozygous in the samples to check for selfed samples.

Select genotype_TaxaSummary -> go to “Results” -> click on “Charts” -> select “Proportion Heterozygous” under “Series 1”

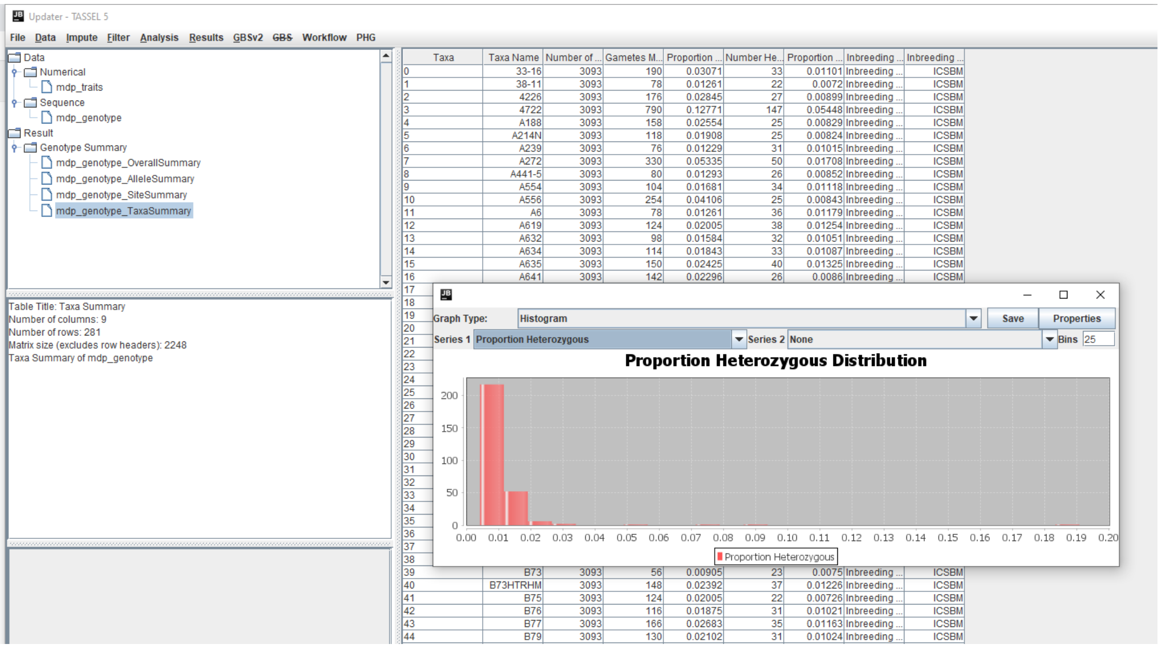


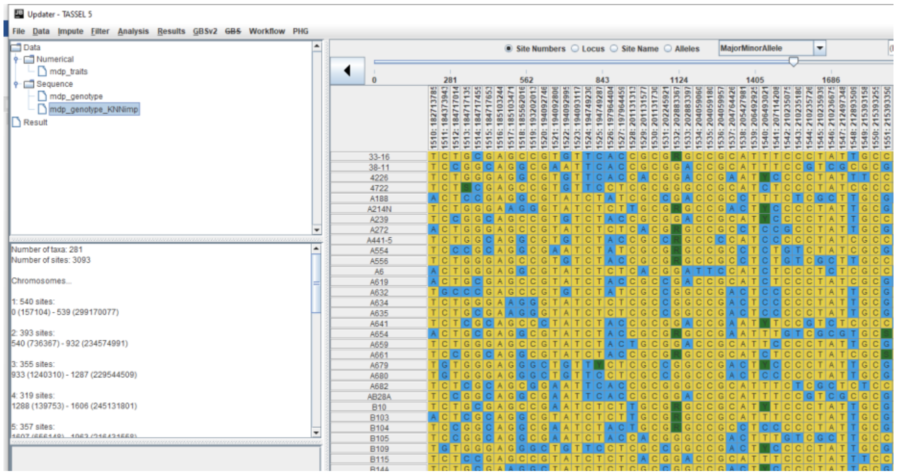
Table Title: Taxa Summary
Number of columns: 9
Number of rows: 281
Matrix size (excludes row headers): 2248
Taxa Summary of mdp_genotype

Graph Type: Histogram
Series 1 Proportion Heterozygous
Series 2 None

| Taxa | Taxa Name | Number of ... | Gametes M... | Proportion ... | Number He... | Proportion ... | Inbreeding ... | Inbreeding ... |
|---|---|---|---|---|---|---|---|---|
| 0 | 33-16 | 3093 | 190 | 0.03071 | 33 | 0.01101 | Inbreeding ... | ICSBM |
| 1 | 38-11 | 3093 | 78 | 0.01261 | 22 | 0.0072 | Inbreeding ... | ICSBM |
| 2 | 4226 | 3093 | 176 | 0.02845 | 27 | 0.00899 | Inbreeding ... | ICSBM |
| 3 | 4722 | 3093 | 790 | 0.12771 | 147 | 0.05448 | Inbreeding ... | ICSBM |
| 4 | A188 | 3093 | 158 | 0.02554 | 25 | 0.00829 | Inbreeding ... | ICSBM |
| 5 | A214N | 3093 | 118 | 0.01908 | 25 | 0.00824 | Inbreeding ... | ICSBM |
| 6 | A239 | 3093 | 76 | 0.01229 | 31 | 0.01015 | Inbreeding ... | ICSBM |
| 7 | A272 | 3093 | 330 | 0.05335 | 50 | 0.01708 | Inbreeding ... | ICSBM |
| 8 | A441-5 | 3093 | 80 | 0.01293 | 26 | 0.00852 | Inbreeding ... | ICSBM |
| 9 | A554 | 3093 | 104 | 0.01681 | 34 | 0.01118 | Inbreeding ... | ICSBM |
| 10 | A556 | 3093 | 254 | 0.04106 | 25 | 0.00843 | Inbreeding ... | ICSBM |
| 11 | A6 | 3093 | 78 | 0.01261 | 36 | 0.01179 | Inbreeding ... | ICSBM |
| 12 | A619 | 3093 | 124 | 0.02005 | 38 | 0.01254 | Inbreeding ... | ICSBM |
| 13 | A632 | 3093 | 98 | 0.01584 | 32 | 0.01051 | Inbreeding ... | ICSBM |
| 14 | A634 | 3093 | 114 | 0.01843 | 33 | 0.01087 | Inbreeding ... | ICSBM |
| 15 | A635 | 3093 | 150 | 0.02425 | 40 | 0.01325 | Inbreeding ... | ICSBM |
| 16 | A641 | 3093 | 142 | 0.02296 | 26 | 0.0086 | Inbreeding ... | ICSBM |
| 17 | | | | | | | | |
| 18 | | | | | | | | |
| 19 | | | | | | | | |
| 20 | | | | | | | | |
| 21 | | | | | | | | |
| 22 | | | | | | | | |
| 23 | | | | | | | | |
| 24 | | | | | | | | |
| 25 | | | | | | | | |
| 26 | | | | | | | | |
| 27 | | | | | | | | |
| 28 | | | | | | | | |
| 29 | | | | | | | | |
| 30 | | | | | | | | |
| 31 | | | | | | | | |
| 32 | | | | | | | | |
| 33 | | | | | | | | |
| 34 | | | | | | | | |
| 35 | | | | | | | | |
| 36 | | | | | | | | |
| 37 | | | | | | | | |
| 38 | | | | | | | | |
| 39 | B73 | 3093 | 56 | 0.00905 | 23 | 0.0075 | Inbreeding ... | ICSBM |
| 40 | B73HTRHM | 3093 | 148 | 0.02392 | 37 | 0.01226 | Inbreeding ... | ICSBM |
| 41 | B75 | 3093 | 124 | 0.02005 | 22 | 0.00726 | Inbreeding ... | ICSBM |
| 42 | B76 | 3093 | 116 | 0.01875 | 31 | 0.01021 | Inbreeding ... | ICSBM |
| 43 | B77 | 3093 | 166 | 0.02683 | 35 | 0.01163 | Inbreeding ... | ICSBM |
| 44 | B79 | 3093 | 130 | 0.02102 | 31 | 0.01024 | Inbreeding ... | ICSBM |

## 6. Imputation of missing values

Select genotype file -> go to “impute” -> click on “LD KNNi imputation” -> set parameters ->click “okay”



Number of taxa: 281
Number of sites: 3093

Chromosomes...

1: 540 sites:
0 (157104) - 539 (299170077)

2: 393 sites:
540 (736367) - 932 (234574991)

3: 355 sites:
933 (1240310) - 1287 (229544509)

4: 319 sites:
1288 (139753) - 1606 (245131801)

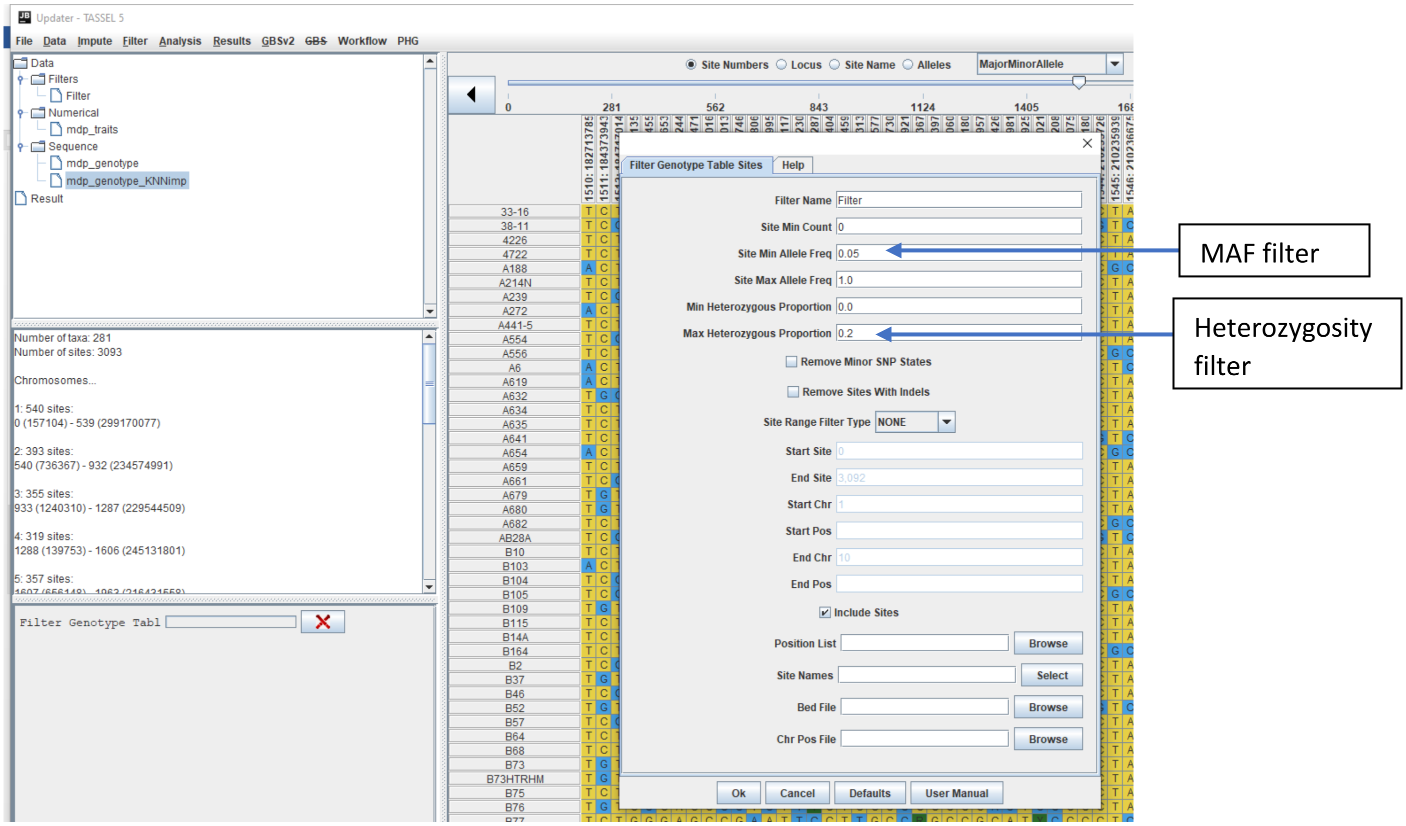
5: 357 sites:

## 7. Filter Markers based on Minor allele frequency (MAF)

Steps to filter markers based on Minor allele frequency (MAF) are shown below:

0.05 Minor allele Frequency (set filter thresholds for rare alleles)

Select genotype file -> go to “filter” -> click on “Filter Genotype Table Sites” -> set parameters -> click “OK”



Number of taxa: 281
Number of sites: 3093

Chromosomes...

1: 540 sites:
0 (157104) - 539 (299170077)

2: 393 sites:
540 (736367) - 932 (234574991)

3: 355 sites:
933 (1240310) - 1287 (229544509)

4: 319 sites:
1288 (139753) - 1606 (245131801)

5: 357 sites:

Filter Name: Filter
Site Min Count: 0
Site Min Allele Freq: 0.05
Site Max Allele Freq: 1.0
Min Heterozygous Proportion: 0.0
Max Heterozygous Proportion: 0.2
□ Remove Minor SNP States
□ Remove Sites With Indels
Site Range Filter Type: NONE
Start Site: 0
End Site: 3,092
Start Chr: 1
Start Pos:
End Chr: 10
End Pos:
☑ Include Sites
Position List:
Site Names:
Bed File:
Chr Pos File:

MAF filter

Heterozygosity filter

## Conduct GWAS analysis

## 8. Principal component analysis (PCA)

PCA output can be used as the covariate in the GLM or MLM to correct for population structure. Please follow the steps shown below:

Select genotype file -> go to “Analysis” -> go to “Relatedness” -> click on “PCA”-> set parameters -> click “ok”

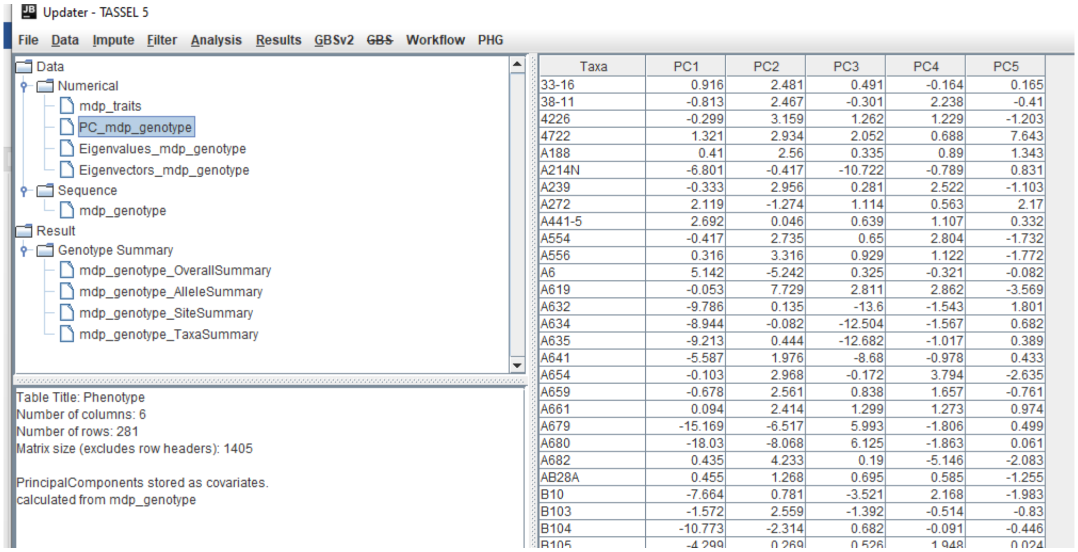


Table Title: Phenotype
Number of columns: 6
Number of rows: 281
Matrix size (excludes row headers): 1405

PrincipalComponents stored as covariates.
calculated from mdp_genotype

| Taxa | PC1 | PC2 | PC3 | PC4 | PC5 |
|---|---|---|---|---|---|
| 33-16 | 0.916 | 2.481 | 0.491 | -0.164 | 0.165 |
| 38-11 | -0.813 | 2.467 | -0.301 | 2.238 | -0.41 |
| 4226 | -0.299 | 3.159 | 1.262 | 1.229 | -1.203 |
| 4722 | 1.321 | 2.934 | 2.052 | 0.688 | 7.643 |
| A188 | 0.41 | 2.56 | 0.335 | 0.89 | 1.343 |
| A214N | -6.801 | -0.417 | -10.722 | -0.789 | 0.831 |
| A239 | -0.333 | 2.956 | 0.281 | 2.522 | -1.103 |
| A272 | 2.119 | -1.274 | 1.114 | 0.563 | 2.17 |
| A441-5 | 2.692 | 0.046 | 0.639 | 1.107 | 0.332 |
| A554 | -0.417 | 2.735 | 0.65 | 2.804 | -1.732 |
| A556 | 0.316 | 3.316 | 0.929 | 1.122 | -1.772 |
| A6 | 5.142 | -5.242 | 0.325 | -0.321 | -0.082 |
| A619 | -0.053 | 7.729 | 2.811 | 2.862 | -3.569 |
| A632 | -9.786 | 0.135 | -13.6 | -1.543 | 1.801 |
| A634 | -8.944 | -0.082 | -12.504 | -1.567 | 0.682 |
| A635 | -9.213 | 0.444 | -12.682 | -1.017 | 0.389 |
| A641 | -5.587 | 1.976 | -8.68 | -0.978 | 0.433 |
| A654 | -0.103 | 2.968 | -0.172 | 3.794 | -2.635 |
| A659 | -0.678 | 2.561 | 0.838 | 1.657 | -0.761 |
| A661 | 0.094 | 2.414 | 1.299 | 1.273 | 0.974 |
| A679 | -15.169 | -6.517 | 5.993 | -1.806 | 0.499 |
| A680 | -18.03 | -8.068 | 6.125 | -1.863 | 0.061 |
| A682 | 0.435 | 4.233 | 0.19 | -5.146 | -2.083 |
| AB28A | 0.455 | 1.268 | 0.695 | 0.585 | -1.255 |
| B10 | -7.664 | 0.781 | -3.521 | 2.168 | -1.983 |
| B103 | -1.572 | 2.559 | -1.392 | -0.514 | -0.83 |
| B104 | -10.773 | -2.314 | 0.682 | -0.091 | -0.446 |
| B105 | -4.299 | 0.269 | 0.526 | 1.948 | 0.024 |

## 9. Intersecting the files

Intersect the genotype, phenotype and PCA files by following the steps below:

Select genotype, phenotype and PCA files simultaneously by holding 'CTRL' button -> go to "Data" -> click on "Intersect join"

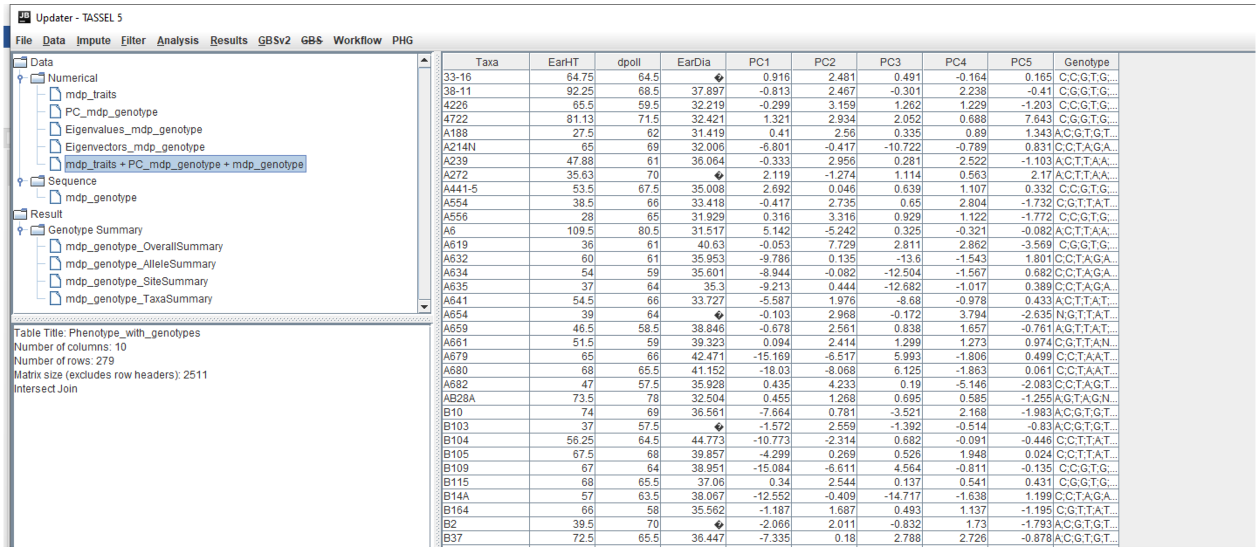

Updater - TASSEL 5
File Data Impute Filter Analysis Results GBSv2 ~~GBS~~ Workflow PHG
Data
Numerical
mdp_traits
PC_mdp_genotype
Eigenvalues_mdp_genotype
Eigenvectors_mdp_genotype
mdp_traits + PC_mdp_genotype + mdp_genotype
Sequence
mdp_genotype
Result
Genotype Summary
mdp_genotype_OverallSummary
mdp_genotype_AlleleSummary
mdp_genotype_SiteSummary
mdp_genotype_TaxaSummary

Table Title: Phenotype_with_genotypes
Number of columns: 10
Number of rows: 279
Matrix size (excludes row headers): 2511
Intersect Join

| Taxa | EarHT | dpoll | EarDia | PC1 | PC2 | PC3 | PC4 | PC5 | Genotype |
|---|---|---|---|---|---|---|---|---|---|
| 33-16 | 64.75 | 64.5 | � | 0.916 | 2.481 | 0.491 | -0.164 | 0.165 | C;C;G;T;G;... |
| 38-11 | 92.25 | 68.5 | 37.897 | -0.813 | 2.467 | -0.301 | 2.238 | -0.41 | C;G;G;T;G;... |
| 4226 | 65.5 | 59.5 | 32.219 | -0.299 | 3.159 | 1.262 | 1.229 | -1.203 | C;C;G;T;G;... |
| 4722 | 81.13 | 71.5 | 32.421 | 1.321 | 2.934 | 2.052 | 0.688 | 7.643 | C;G;G;T;G;... |
| A188 | 27.5 | 62 | 31.419 | 0.41 | 2.56 | 0.335 | 0.89 | 1.343 | A;C;G;T;G;T... |
| A214N | 65 | 69 | 32.006 | -6.801 | -0.417 | -10.722 | -0.789 | 0.831 | C;C;T;A;G;A... |
| A239 | 47.88 | 61 | 36.064 | -0.333 | 2.956 | 0.281 | 2.522 | -1.103 | A;C;T;T;A;A;... |
| A272 | 35.63 | 70 | � | 2.119 | -1.274 | 1.114 | 0.563 | 2.17 | A;C;T;T;A;A;... |
| A441-5 | 53.5 | 67.5 | 35.008 | 2.692 | 0.046 | 0.639 | 1.107 | 0.332 | C;C;G;T;G;... |
| A554 | 38.5 | 66 | 33.418 | -0.417 | 2.735 | 0.65 | 2.804 | -1.732 | C;G;T;T;A;T... |
| A556 | 28 | 65 | 31.929 | 0.316 | 3.316 | 0.929 | 1.122 | -1.772 | C;C;G;T;G;... |
| A6 | 109.5 | 80.5 | 31.517 | 5.142 | -5.242 | 0.325 | -0.321 | -0.082 | A;C;T;T;A;A;... |
| A619 | 36 | 61 | 40.63 | -0.053 | 7.729 | 2.811 | 2.862 | -3.569 | C;G;G;T;G;... |
| A632 | 60 | 61 | 35.953 | -9.786 | 0.135 | -13.6 | -1.543 | 1.801 | C;C;T;A;G;A... |
| A634 | 54 | 59 | 35.601 | -8.944 | -0.082 | -12.504 | -1.567 | 0.682 | C;C;T;A;G;A... |
| A635 | 37 | 64 | 35.3 | -9.213 | 0.444 | -12.682 | -1.017 | 0.389 | C;C;T;A;G;A... |
| A641 | 54.5 | 66 | 33.727 | -5.587 | 1.976 | -8.68 | -0.978 | 0.433 | A;C;T;T;A;T;... |
| A654 | 39 | 64 | � | -0.103 | 2.968 | -0.172 | 3.794 | -2.635 | N;G;T;T;A;T... |
| A659 | 46.5 | 58.5 | 38.846 | -0.678 | 2.561 | 0.838 | 1.657 | -0.761 | A;G;T;T;A;T;... |
| A661 | 51.5 | 59 | 39.323 | 0.094 | 2.414 | 1.299 | 1.273 | 0.974 | C;G;T;T;A;N... |
| A679 | 65 | 66 | 42.471 | -15.169 | -6.517 | 5.993 | -1.806 | 0.499 | C;C;T;A;A;T... |
| A680 | 68 | 65.5 | 41.152 | -18.03 | -8.068 | 6.125 | -1.863 | 0.061 | C;C;T;A;A;T... |
| A682 | 47 | 57.5 | 35.928 | 0.435 | 4.233 | 0.19 | -5.146 | -2.083 | C;C;T;A;G;T... |
| AB28A | 73.5 | 78 | 32.504 | 0.455 | 1.268 | 0.695 | 0.585 | -1.255 | A;G;T;A;G;N... |
| B10 | 74 | 69 | 36.561 | -7.664 | 0.781 | -3.521 | 2.168 | -1.983 | A;C;G;T;G;T... |
| B103 | 37 | 57.5 | � | -1.572 | 2.559 | -1.392 | -0.514 | -0.83 | A;C;G;T;G;T... |
| B104 | 56.25 | 64.5 | 44.773 | -10.773 | -2.314 | 0.682 | -0.091 | -0.446 | C;C;T;T;A;T... |
| B105 | 67.5 | 68 | 39.857 | -4.299 | 0.269 | 0.526 | 1.948 | 0.024 | C;C;T;T;A;T... |
| B109 | 67 | 64 | 38.951 | -15.084 | -6.611 | 4.564 | -0.811 | -0.135 | C;C;G;T;G;... |
| B115 | 68 | 65.5 | 37.06 | 0.34 | 2.544 | 0.137 | 0.541 | 0.431 | C;G;G;T;G;... |
| B14A | 57 | 63.5 | 38.067 | -12.552 | -0.409 | -14.717 | -1.638 | 1.199 | C;C;T;A;G;A... |
| B164 | 66 | 58 | 35.562 | -1.187 | 1.687 | 0.493 | 1.137 | -1.195 | C;G;T;T;A;T... |
| B2 | 39.5 | 70 | � | -2.066 | 2.011 | -0.832 | 1.73 | -1.793 | A;C;G;T;G;T... |
| B37 | 72.5 | 65.5 | 36.447 | -7.335 | 0.18 | 2.788 | 2.726 | -0.878 | A;C;G;T;G;T... |

## 10. Running General Linear Model (GLM)

Run the GLM analysis by selecting the intersected files following the steps below:

Select the intersect joined file "mdp_traits + PC_mdp_genotype + mdp_genotype" -> go to "Analysis" -> go to "association" -> click on "GLM" -> set parameters -> click "ok"

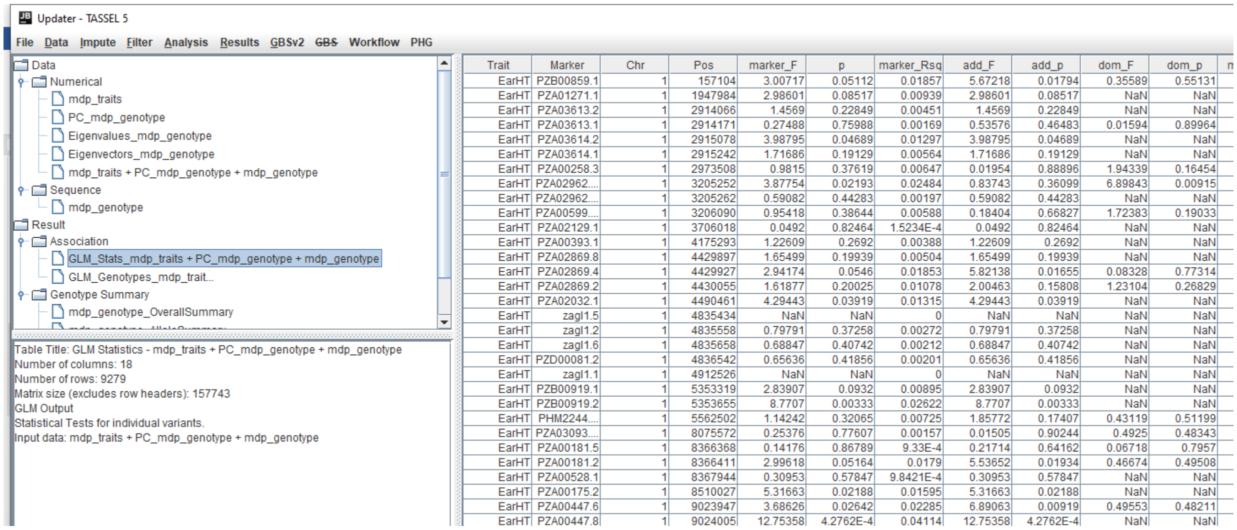


Table Title: GLM Statistics - mdp_traits + PC_mdp_genotype + mdp_genotype
Number of columns: 18
Number of rows: 9279
Matrix size (excludes row headers): 157743
GLM Output
Statistical Tests for individual variants.
Input data: mdp_traits + PC_mdp_genotype + mdp_genotype

| Trait | Marker | Chr | Pos | marker_F | p | marker_Rsq | add_F | add_p | dom_F | dom_p |
|---|---|---|---|---|---|---|---|---|---|---|
| EarHT | PZB00859.1 | 1 | 157104 | 3.00717 | 0.05112 | 0.01857 | 5.67218 | 0.01794 | 0.35589 | 0.55131 |
| EarHT | PZA01271.1 | 1 | 1947984 | 2.98601 | 0.08517 | 0.00939 | 2.98601 | 0.08517 | NaN | NaN |
| EarHT | PZA03613.2 | 1 | 2914066 | 1.4569 | 0.22849 | 0.00451 | 1.4569 | 0.22849 | NaN | NaN |
| EarHT | PZA03613.1 | 1 | 2914171 | 0.27488 | 0.75988 | 0.00169 | 0.53576 | 0.46483 | 0.01594 | 0.89964 |
| EarHT | PZA03614.2 | 1 | 2915078 | 3.98795 | 0.04689 | 0.01297 | 3.98795 | 0.04689 | NaN | NaN |
| EarHT | PZA03614.1 | 1 | 2915242 | 1.71686 | 0.19129 | 0.00564 | 1.71686 | 0.19129 | NaN | NaN |
| EarHT | PZA00258.3 | 1 | 2973508 | 0.9815 | 0.37619 | 0.00647 | 0.01954 | 0.88896 | 1.94339 | 0.16454 |
| EarHT | PZA02962.... | 1 | 3205252 | 3.87754 | 0.02193 | 0.02484 | 0.83743 | 0.36099 | 6.89843 | 0.00915 |
| EarHT | PZA02962.... | 1 | 3205262 | 0.59082 | 0.44283 | 0.00197 | 0.59082 | 0.44283 | NaN | NaN |
| EarHT | PZA00599.... | 1 | 3206090 | 0.95418 | 0.38644 | 0.00588 | 0.18404 | 0.66827 | 1.72383 | 0.19033 |
| EarHT | PZA02129.1 | 1 | 3706018 | 0.0492 | 0.82464 | 1.5234E-4 | 0.0492 | 0.82464 | NaN | NaN |
| EarHT | PZA00393.1 | 1 | 4175293 | 1.22609 | 0.2692 | 0.00388 | 1.22609 | 0.2692 | NaN | NaN |
| EarHT | PZA02869.8 | 1 | 4429897 | 1.65499 | 0.19939 | 0.00504 | 1.65499 | 0.19939 | NaN | NaN |
| EarHT | PZA02869.4 | 1 | 4429927 | 2.94174 | 0.0546 | 0.01853 | 5.82138 | 0.01655 | 0.08328 | 0.77314 |
| EarHT | PZA02869.2 | 1 | 4430055 | 1.61877 | 0.20025 | 0.01078 | 2.00463 | 0.15808 | 1.23104 | 0.26829 |
| EarHT | PZA02032.1 | 1 | 4490461 | 4.29443 | 0.03919 | 0.01315 | 4.29443 | 0.03919 | NaN | NaN |
| EarHT | zagl1.5 | 1 | 4835434 | NaN | NaN | 0 | NaN | NaN | NaN | NaN |
| EarHT | zagl1.2 | 1 | 4835558 | 0.79791 | 0.37258 | 0.00272 | 0.79791 | 0.37258 | NaN | NaN |
| EarHT | zagl1.6 | 1 | 4835658 | 0.68847 | 0.40742 | 0.00212 | 0.68847 | 0.40742 | NaN | NaN |
| EarHT | PZD00081.2 | 1 | 4836542 | 0.65636 | 0.41856 | 0.00201 | 0.65636 | 0.41856 | NaN | NaN |
| EarHT | zagl1.1 | 1 | 4912526 | NaN | NaN | 0 | NaN | NaN | NaN | NaN |
| EarHT | PZB00919.1 | 1 | 5353319 | 2.83907 | 0.0932 | 0.00895 | 2.83907 | 0.0932 | NaN | NaN |
| EarHT | PZB00919.2 | 1 | 5353655 | 8.7707 | 0.00333 | 0.02622 | 8.7707 | 0.00333 | NaN | NaN |
| EarHT | PHM2244.... | 1 | 5562502 | 1.14242 | 0.32065 | 0.00725 | 1.85772 | 0.17407 | 0.43119 | 0.51199 |
| EarHT | PZA03093.... | 1 | 8075572 | 0.25376 | 0.77607 | 0.00157 | 0.01505 | 0.90244 | 0.4925 | 0.48343 |
| EarHT | PZA00181.5 | 1 | 8366368 | 0.14176 | 0.86789 | 9.33E-4 | 0.21714 | 0.64162 | 0.06718 | 0.7957 |
| EarHT | PZA00181.2 | 1 | 8366411 | 2.99618 | 0.05164 | 0.0179 | 5.53652 | 0.01934 | 0.46674 | 0.49508 |
| EarHT | PZA00528.1 | 1 | 8367944 | 0.30953 | 0.57847 | 9.8421E-4 | 0.30953 | 0.57847 | NaN | NaN |
| EarHT | PZA00175.2 | 1 | 8510027 | 5.31663 | 0.02188 | 0.01595 | 5.31663 | 0.02188 | NaN | NaN |
| EarHT | PZA00447.6 | 1 | 9023947 | 3.68626 | 0.02642 | 0.02285 | 6.89063 | 0.00919 | 0.49553 | 0.48211 |
| EarHT | PZA00447.8 | 1 | 9024005 | 12.75358 | 4.2762E-4 | 0.04114 | 12.75358 | 4.2762E-4 | NaN | NaN |

The output of the GLM analysis is produced under the Result node. The GLM association test can be evaluated by plotting Q-Q plot and the Manhattan plot as shown below.

Select the association analysis output file -> go to "Results" -> click on "Manhattan plot"-> select the trait

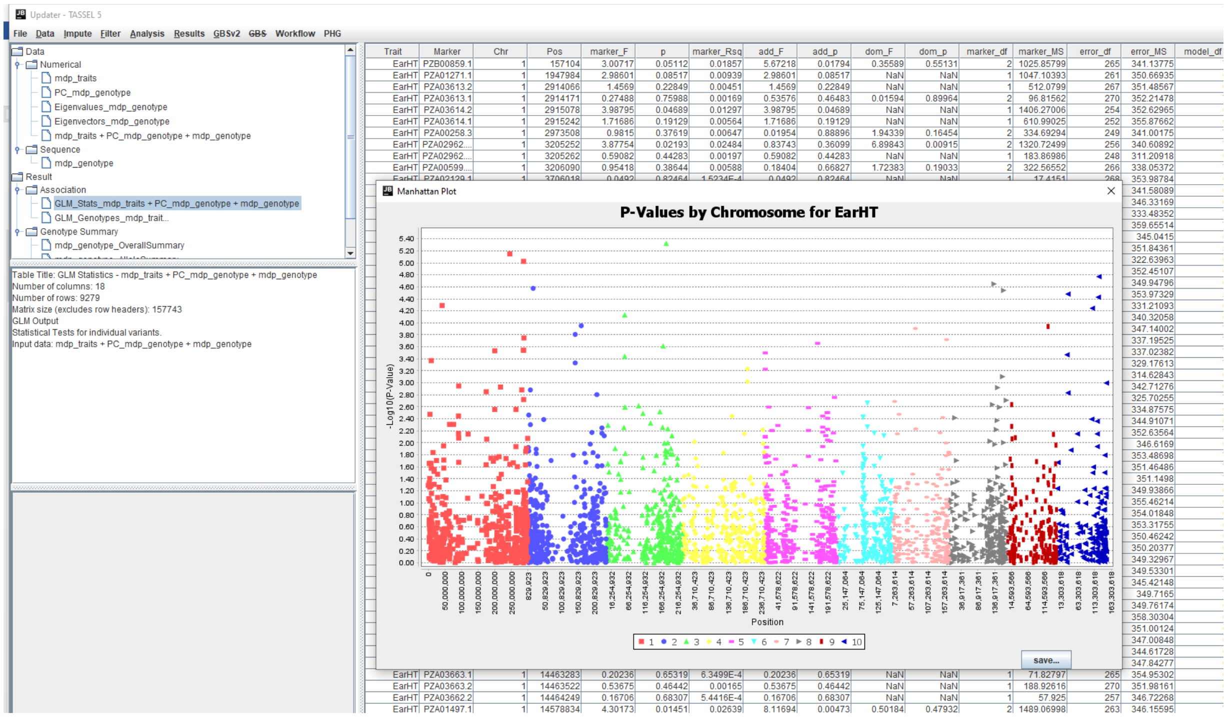


Table Title: GLM Statistics - mdp_traits + PC_mdp_genotype + mdp_genotype
Number of columns: 18
Number of rows: 9279
Matrix size (excludes row headers): 157743
GLM Output
Statistical Tests for individual variants.
Input data: mdp_traits + PC_mdp_genotype + mdp_genotype

| Trait | Marker | Chr | Pos | marker_F | p | marker_Rsq | add_F | add_p | dom_F | dom_p | marker_df | marker_MS | error_df | error_MS | model_df |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EarHT | PZB00859.1 | 1 | 157104 | 3.00717 | 0.05112 | 0.01857 | 5.67218 | 0.01794 | 0.35589 | 0.55131 | 2 | 1025.85799 | 265 | 341.13775 | |
| EarHT | PZA01271.1 | 1 | 1947984 | 2.98601 | 0.08517 | 0.00939 | 2.98601 | 0.08517 | NaN | NaN | 1 | 1047.10393 | 261 | 350.66935 | |
| EarHT | PZA03613.2 | 1 | 2914066 | 1.4569 | 0.22849 | 0.00451 | 1.4569 | 0.22849 | NaN | NaN | 1 | 512.0799 | 267 | 351.48567 | |
| EarHT | PZA03613.1 | 1 | 2914171 | 0.27488 | 0.75988 | 0.00169 | 0.53576 | 0.46483 | 0.01594 | 0.89964 | 2 | 96.81562 | 270 | 352.21478 | |
| EarHT | PZA03614.2 | 1 | 2915078 | 3.98795 | 0.04689 | 0.01297 | 3.98795 | 0.04689 | NaN | NaN | 1 | 1406.27006 | 254 | 352.62965 | |
| EarHT | PZA03614.1 | 1 | 2915242 | 1.71686 | 0.19129 | 0.00564 | 1.71686 | 0.19129 | NaN | NaN | 1 | 610.99025 | 252 | 355.87662 | |
| EarHT | PZA00258.3 | 1 | 2973508 | 0.9815 | 0.37619 | 0.00647 | 0.01954 | 0.88896 | 1.94339 | 0.16454 | 2 | 334.69294 | 249 | 341.00175 | |
| EarHT | PZA02962.... | 1 | 3205252 | 3.87754 | 0.02193 | 0.02484 | 0.83743 | 0.36099 | 6.89843 | 0.00915 | 2 | 1320.72499 | 256 | 340.60892 | |
| EarHT | PZA02962.... | 1 | 3205262 | 0.59082 | 0.44283 | 0.00197 | 0.59082 | 0.44283 | NaN | NaN | 1 | 183.86986 | 248 | 311.20918 | |
| EarHT | PZA00599.... | 1 | 3206090 | 0.95418 | 0.38644 | 0.00588 | 0.18404 | 0.66827 | 1.72383 | 0.19033 | 2 | 322.56552 | 266 | 338.05372 | |
| EarHT | PZA02129.1 | 1 | 3706018 | 0.0492 | 0.82464 | 1.5234E-4 | 0.0492 | 0.82464 | NaN | NaN | 1 | 17.4151 | 268 | 353.98784 | |
| | | | | | | | | | | | | | | 341.58089 | |
| | | | | | | | | | | | | | | 346.33169 | |
| | | | | | | | | | | | | | | 333.48352 | |
| | | | | | | | | | | | | | | 359.65514 | |
| | | | | | | | | | | | | | | 345.0415 | |
| | | | | | | | | | | | | | | 351.84361 | |
| | | | | | | | | | | | | | | 322.63963 | |
| | | | | | | | | | | | | | | 352.45107 | |
| | | | | | | | | | | | | | | 349.94796 | |
| | | | | | | | | | | | | | | 353.97329 | |
| | | | | | | | | | | | | | | 331.21093 | |
| | | | | | | | | | | | | | | 340.32058 | |
| | | | | | | | | | | | | | | 347.14002 | |
| | | | | | | | | | | | | | | 337.19525 | |
| | | | | | | | | | | | | | | 337.02382 | |
| | | | | | | | | | | | | | | 329.17613 | |
| | | | | | | | | | | | | | | 314.62843 | |
| | | | | | | | | | | | | | | 342.71276 | |
| | | | | | | | | | | | | | | 325.70255 | |
| | | | | | | | | | | | | | | 334.87575 | |
| | | | | | | | | | | | | | | 344.91071 | |
| | | | | | | | | | | | | | | 352.63564 | |
| | | | | | | | | | | | | | | 346.6169 | |
| | | | | | | | | | | | | | | 353.48698 | |
| | | | | | | | | | | | | | | 351.46486 | |
| | | | | | | | | | | | | | | 351.1498 | |
| | | | | | | | | | | | | | | 349.93866 | |
| | | | | | | | | | | | | | | 355.46214 | |
| | | | | | | | | | | | | | | 354.01848 | |
| | | | | | | | | | | | | | | 353.31755 | |
| | | | | | | | | | | | | | | 350.46242 | |
| | | | | | | | | | | | | | | 350.20377 | |
| | | | | | | | | | | | | | | 349.32967 | |
| | | | | | | | | | | | | | | 349.53301 | |
| | | | | | | | | | | | | | | 345.42148 | |
| | | | | | | | | | | | | | | 349.7165 | |
| | | | | | | | | | | | | | | 349.76174 | |
| | | | | | | | | | | | | | | 358.30304 | |
| | | | | | | | | | | | | | | 351.00124 | |
| | | | | | | | | | | | | | | 347.00848 | |
| | | | | | | | | | | | | | | 344.61728 | |
| | | | | | | | | | | | | | | 347.84277 | |
| EarHT | PZA03663.1 | 1 | 14463283 | 0.20236 | 0.65319 | 6.3499E-4 | 0.20236 | 0.65319 | NaN | NaN | 1 | 71.82797 | 265 | 354.95302 | |
| EarHT | PZA03663.2 | 1 | 14463522 | 0.53675 | 0.46442 | 0.00165 | 0.53675 | 0.46442 | NaN | NaN | 1 | 188.92616 | 270 | 351.98161 | |
| EarHT | PZA03662.2 | 1 | 14464249 | 0.16706 | 0.68307 | 5.4416E-4 | 0.16706 | 0.68307 | NaN | NaN | 1 | 57.925 | 257 | 346.72268 | |
| EarHT | PZA01497.1 | 1 | 14578834 | 4.30173 | 0.01451 | 0.02639 | 8.11694 | 0.00473 | 0.50184 | 0.47932 | 2 | 1489.06998 | 263 | 346.15595 | |

Select the association analysis output file -> go to "Results" -> click on "QQ plot"-> select the trait -> click "okay"

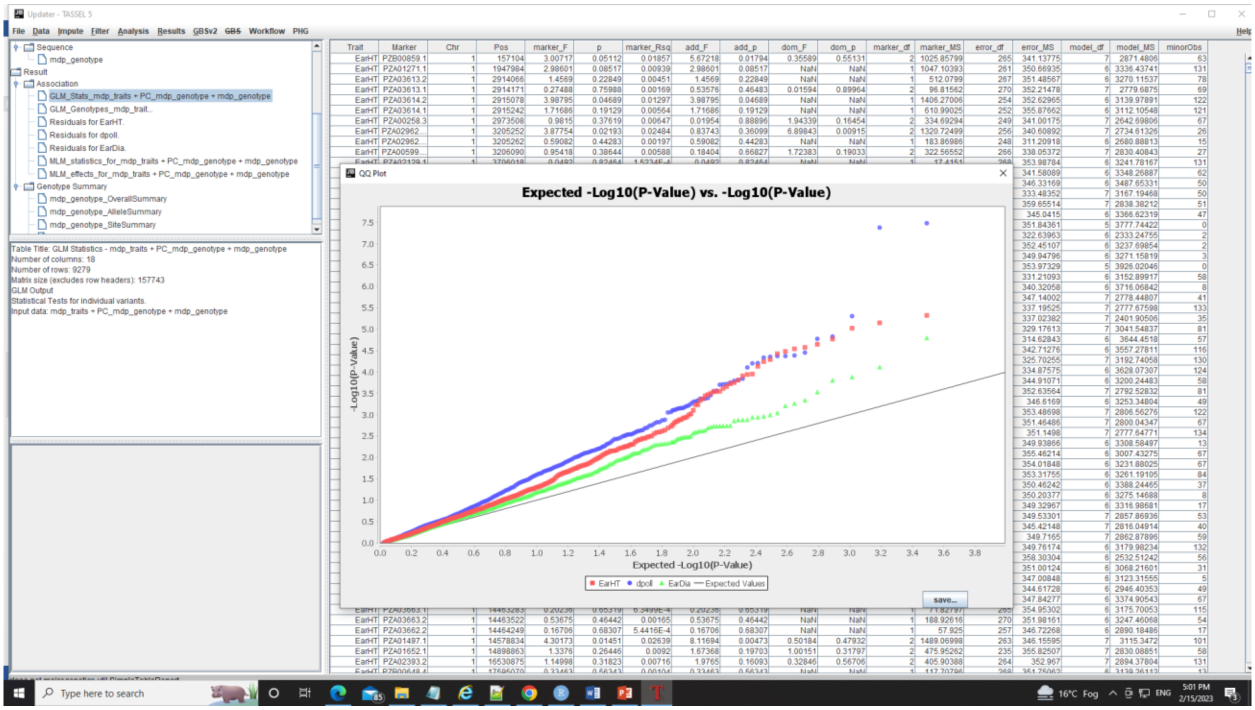

Updater - TASSEL 5
File Data Impute Filter Analysis Results GBSv2 GBS Workflow PHG
Help
Sequence
mdp_genotype
Result
Association
GLM_Stats_mdp_traits + PC_mdp_genotype + mdp_genotype
GLM_Genotypes_mdp_trait...
Residuals for EarHT.
Residuals for dpoll.
Residuals for EarDia.
MLM_statistics_for_mdp_traits + PC_mdp_genotype + mdp_genotype
MLM_effects_for_mdp_traits + PC_mdp_genotype + mdp_genotype
Genotype Summary
mdp_genotype_OverallSummary
mdp_genotype_AlleleSummary
mdp_genotype_SiteSummary
QQ Plot
Expected -Log10(P-Value) vs. -Log10(P-Value)
-Log10(P-Value)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8 2.0 2.2 2.4 2.6 2.8 3.0 3.2 3.4 3.6 3.8
Expected -Log10(P-Value)
■ EarHT ● dpoll ▲ EarDia — Expected Values
save...
Type here to search
16°C Fog ENG
5:01 PM
2/15/2023

Table Title: GLM Statistics - mdp_traits + PC_mdp_genotype + mdp_genotype
Number of columns: 18
Number of rows: 9279
Matrix size (excludes row headers): 157743
GLM Output
Statistical Tests for individual variants.
Input data: mdp_traits + PC_mdp_genotype + mdp_genotype

| Trait | Marker | Chr | Pos | marker_F | p | marker_Rsq | add_F | add_p | dom_F | dom_p | marker_df | marker_MS | error_df | error_MS | model_df | model_MS | minorObs |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EarHT | PZB00859.1 | 1 | 157104 | 3.00717 | 0.05112 | 0.01857 | 5.67218 | 0.01794 | 0.35589 | 0.55131 | 2 | 1025.85799 | 265 | 341.13775 | 7 | 2871.4806 | 63 |
| EarHT | PZA01271.1 | 1 | 1947984 | 2.98601 | 0.08517 | 0.00939 | 2.98601 | 0.08517 | NaN | NaN | 1 | 1047.10393 | 261 | 350.66935 | 6 | 3336.43741 | 131 |
| EarHT | PZA03613.2 | 1 | 2914066 | 1.4569 | 0.22849 | 0.00451 | 1.4569 | 0.22849 | NaN | NaN | 1 | 512.0799 | 267 | 351.48567 | 6 | 3270.11537 | 78 |
| EarHT | PZA03613.1 | 1 | 2914171 | 0.27488 | 0.75988 | 0.00169 | 0.53576 | 0.46483 | 0.01594 | 0.89964 | 2 | 96.81562 | 270 | 352.21478 | 7 | 2779.6875 | 69 |
| EarHT | PZA03614.2 | 1 | 2915078 | 3.98795 | 0.04689 | 0.01297 | 3.98795 | 0.04689 | NaN | NaN | 1 | 1406.27006 | 254 | 352.62965 | 6 | 3139.97891 | 122 |
| EarHT | PZA03614.1 | 1 | 2915242 | 1.71686 | 0.19129 | 0.00564 | 1.71686 | 0.19129 | NaN | NaN | 1 | 610.99025 | 252 | 355.87662 | 6 | 3112.10548 | 121 |
| EarHT | PZA00258.3 | 1 | 2973508 | 0.9815 | 0.37619 | 0.00647 | 0.01954 | 0.88896 | 1.94339 | 0.16454 | 2 | 334.69294 | 249 | 341.00175 | 7 | 2642.69806 | 67 |
| EarHT | PZA02962.... | 1 | 3205252 | 3.87754 | 0.02193 | 0.02484 | 0.83743 | 0.36099 | 6.89843 | 0.00915 | 2 | 1320.72499 | 256 | 340.60892 | 7 | 2734.61326 | 26 |
| EarHT | PZA02962.... | 1 | 3205262 | 0.59082 | 0.44283 | 0.00197 | 0.59082 | 0.44283 | NaN | NaN | 1 | 183.86986 | 248 | 311.20918 | 6 | 2680.88813 | 15 |
| EarHT | PZA00599.... | 1 | 3206090 | 0.95418 | 0.38644 | 0.00588 | 0.18404 | 0.66827 | 1.72383 | 0.19033 | 2 | 322.56552 | 266 | 338.05372 | 7 | 2830.40843 | 27 |
| EarHT | PZA03663.2 | 1 | 14463522 | 0.53675 | 0.46442 | 0.00165 | 0.53675 | 0.46442 | NaN | NaN | 1 | 188.92616 | 270 | 351.98161 | 6 | 3247.46068 | 54 |
| EarHT | PZA03662.2 | 1 | 14464249 | 0.16706 | 0.68307 | 5.4416E-4 | 0.16706 | 0.68307 | NaN | NaN | 1 | 57.925 | 257 | 346.72268 | 6 | 2890.18486 | 17 |
| EarHT | PZA01497.1 | 1 | 14578834 | 4.30173 | 0.01451 | 0.02639 | 8.11694 | 0.00473 | 0.50184 | 0.47932 | 2 | 1489.06998 | 263 | 346.15595 | 7 | 3115.3472 | 101 |
| EarHT | PZA01652.1 | 1 | 14898863 | 1.3376 | 0.26446 | 0.0092 | 1.67368 | 0.19703 | 1.00151 | 0.31797 | 2 | 475.95262 | 235 | 355.82507 | 7 | 2830.08851 | 58 |
| EarHT | PZA02393.2 | 1 | 16530875 | 1.14998 | 0.31823 | 0.00716 | 1.9765 | 0.16093 | 0.32846 | 0.56706 | 2 | 405.90388 | 264 | 352.967 | 7 | 2894.37804 | 131 |

## 11. Mixed Linear Model (MLM)

## Calculating Kinship matrix

Follow the below steps to calcuate the kinship matrix:

Select genotype file -> go to "Analysis" ->go to "Relatedness" -> click on "kinship" -> set parameters -> click "ok"

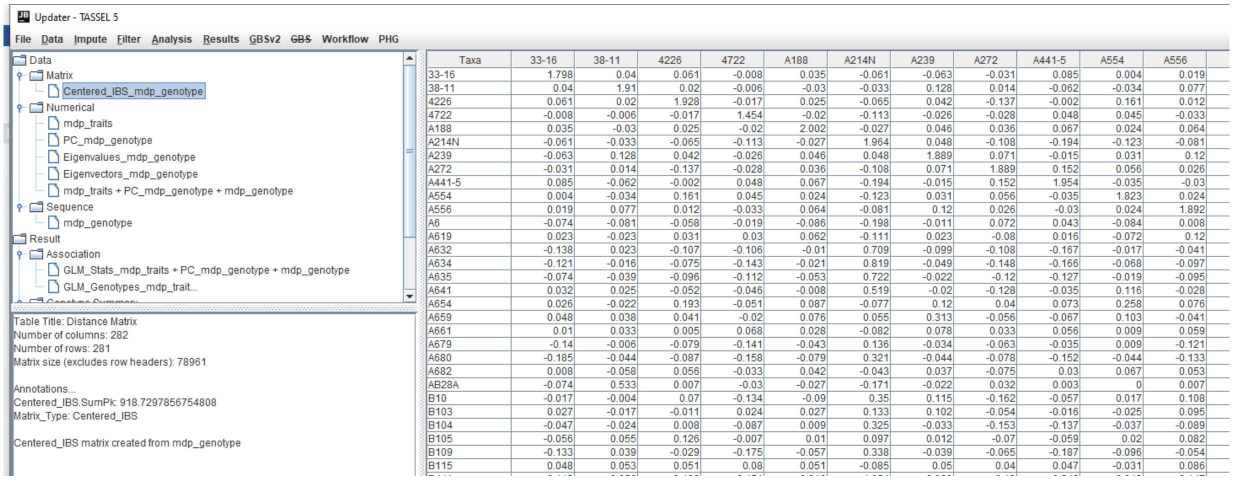

Updater - TASSEL 5
File Data Impute Filter Analysis Results GBSv2 GBS Workflow PHG
Data
Matrix
Centered_IBS_mdp_genotype
Numerical
mdp_traits
PC_mdp_genotype
Eigenvalues_mdp_genotype
Eigenvectors_mdp_genotype
mdp_traits + PC_mdp_genotype + mdp_genotype
Sequence
mdp_genotype
Result
Association
GLM_Stats_mdp_traits + PC_mdp_genotype + mdp_genotype
GLM_Genotypes_mdp_trait...

Table Title: Distance Matrix
Number of columns: 282
Number of rows: 281
Matrix size (excludes row headers): 78961

Annotations...
Centered_IBS.SumPk: 918.7297856754808
Matrix_Type: Centered_IBS

Centered_IBS matrix created from mdp_genotype

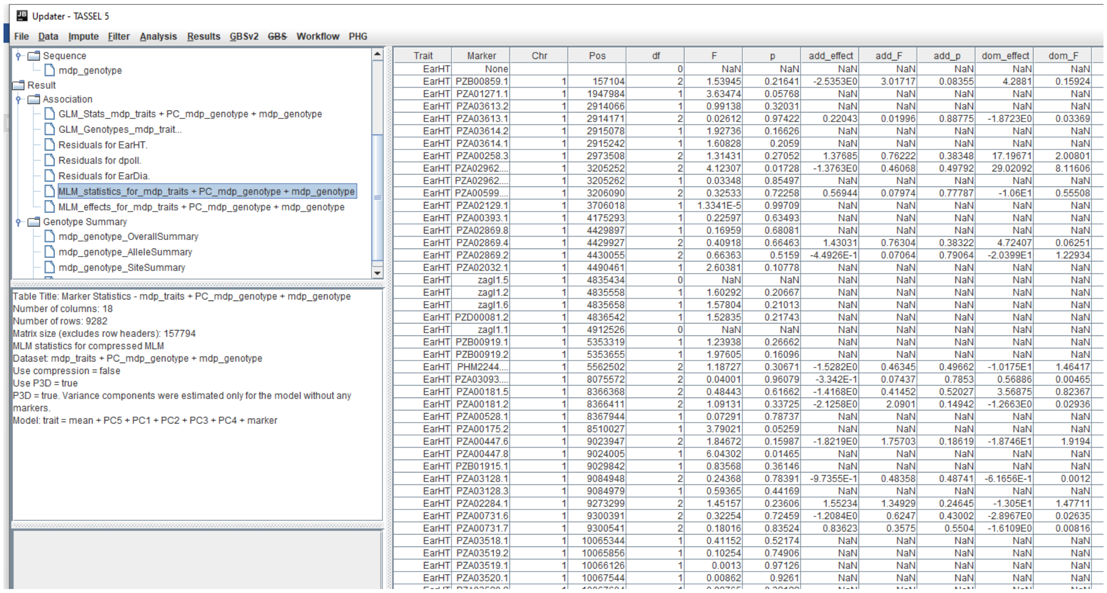
| Taxa | 33-16 | 38-11 | 4226 | 4722 | A188 | A214N | A239 | A272 | A441-5 | A554 | A556 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 33-16 | 1.798 | 0.04 | 0.061 | -0.008 | 0.035 | -0.061 | -0.063 | -0.031 | 0.085 | 0.004 | 0.019 |
| 38-11 | 0.04 | 1.91 | 0.02 | -0.006 | -0.03 | -0.033 | 0.128 | 0.014 | -0.062 | -0.034 | 0.077 |
| 4226 | 0.061 | 0.02 | 1.928 | -0.017 | 0.025 | -0.065 | 0.042 | -0.137 | -0.002 | 0.161 | 0.012 |
| 4722 | -0.008 | -0.006 | -0.017 | 1.454 | -0.02 | -0.113 | -0.026 | -0.028 | 0.048 | 0.045 | -0.033 |
| A188 | 0.035 | -0.03 | 0.025 | -0.02 | 2.002 | -0.027 | 0.046 | 0.036 | 0.067 | 0.024 | 0.064 |
| A214N | -0.061 | -0.033 | -0.065 | -0.113 | -0.027 | 1.964 | 0.048 | -0.108 | -0.194 | -0.123 | -0.081 |
| A239 | -0.063 | 0.128 | 0.042 | -0.026 | 0.046 | 0.048 | 1.889 | 0.071 | -0.015 | 0.031 | 0.12 |
| A272 | -0.031 | 0.014 | -0.137 | -0.028 | 0.036 | -0.108 | 0.071 | 1.889 | 0.152 | 0.056 | 0.026 |
| A441-5 | 0.085 | -0.062 | -0.002 | 0.048 | 0.067 | -0.194 | -0.015 | 0.152 | 1.954 | -0.035 | -0.03 |
| A554 | 0.004 | -0.034 | 0.161 | 0.045 | 0.024 | -0.123 | 0.031 | 0.056 | -0.035 | 1.823 | 0.024 |
| A556 | 0.019 | 0.077 | 0.012 | -0.033 | 0.064 | -0.081 | 0.12 | 0.026 | -0.03 | 0.024 | 1.892 |
| A6 | -0.074 | -0.081 | -0.058 | -0.019 | -0.086 | -0.198 | -0.011 | 0.072 | 0.043 | -0.084 | 0.008 |
| A619 | 0.023 | -0.023 | 0.031 | 0.03 | 0.062 | -0.111 | 0.023 | -0.08 | 0.016 | -0.072 | 0.12 |
| A632 | -0.138 | 0.023 | -0.107 | -0.106 | -0.01 | 0.709 | -0.099 | -0.108 | -0.167 | -0.017 | -0.041 |
| A634 | -0.121 | -0.016 | -0.075 | -0.143 | -0.021 | 0.819 | -0.049 | -0.148 | -0.166 | -0.068 | -0.097 |
| A635 | -0.074 | -0.039 | -0.096 | -0.112 | -0.053 | 0.722 | -0.022 | -0.12 | -0.127 | -0.019 | -0.095 |
| A641 | 0.032 | 0.025 | -0.052 | -0.046 | -0.008 | 0.519 | -0.02 | -0.128 | -0.035 | 0.116 | -0.028 |
| A654 | 0.026 | -0.022 | 0.193 | -0.051 | 0.087 | -0.077 | 0.12 | 0.04 | 0.073 | 0.258 | 0.076 |
| A659 | 0.048 | 0.038 | 0.041 | -0.02 | 0.076 | 0.055 | 0.313 | -0.056 | -0.067 | 0.103 | -0.041 |
| A661 | 0.01 | 0.033 | 0.005 | 0.068 | 0.028 | -0.082 | 0.078 | 0.033 | 0.056 | 0.009 | 0.059 |
| A679 | -0.14 | -0.006 | -0.079 | -0.141 | -0.043 | 0.136 | -0.034 | -0.063 | -0.035 | 0.009 | -0.121 |
| A680 | -0.185 | -0.044 | -0.087 | -0.158 | -0.079 | 0.321 | -0.044 | -0.078 | -0.152 | -0.044 | -0.133 |
| A682 | 0.008 | -0.058 | 0.056 | -0.033 | 0.042 | -0.043 | 0.037 | -0.075 | 0.03 | 0.067 | 0.053 |
| AB28A | -0.074 | 0.533 | 0.007 | -0.03 | -0.027 | -0.171 | -0.022 | 0.032 | 0.003 | 0 | 0.007 |
| B10 | -0.017 | -0.004 | 0.07 | -0.134 | -0.09 | 0.35 | 0.115 | -0.162 | -0.057 | 0.017 | 0.108 |
| B103 | 0.027 | -0.017 | -0.011 | 0.024 | 0.027 | 0.133 | 0.102 | -0.054 | -0.016 | -0.025 | 0.095 |
| B104 | -0.047 | -0.024 | 0.008 | -0.087 | 0.009 | 0.325 | -0.033 | -0.153 | -0.137 | -0.037 | -0.089 |
| B105 | -0.056 | 0.055 | 0.126 | -0.007 | 0.01 | 0.097 | 0.012 | -0.07 | -0.059 | 0.02 | 0.082 |
| B109 | -0.133 | 0.039 | -0.029 | -0.175 | -0.057 | 0.338 | -0.039 | -0.065 | -0.187 | -0.096 | -0.054 |
| B115 | 0.048 | 0.053 | 0.051 | 0.08 | 0.051 | -0.085 | 0.05 | 0.04 | 0.047 | -0.031 | 0.086 |

## Running Mixed Linear Model (MLM)

MLM model includes the PCA and the kinship matrix i.e. MLM (PCA+K).

Therefore, once the Kinship matrix has been calculated, MLM can be now be conducted by following below steps:

Select the intersect joined file “mdp_traits + PC_mdp_genotype + mdp_genotype” and kinship file simultaneously by holding ‘CTRL’ button -> go to “Analysis” -> go to “Association” -> click on “MLM” -> set parameters -> click “okay”


Updater - TASSEL 5
File Data Impute Filter Analysis Results GBSv2 GBS Workflow PHG
Sequence
mdp_genotype
Result
Association
GLM_Stats_mdp_traits + PC_mdp_genotype + mdp_genotype
GLM_Genotypes_mdp_trait...
Residuals for EarHT.
Residuals for dpoll.
Residuals for EarDia.
MLM_statistics_for_mdp_traits + PC_mdp_genotype + mdp_genotype
MLM_effects_for_mdp_traits + PC_mdp_genotype + mdp_genotype
Genotype Summary
mdp_genotype_OverallSummary
mdp_genotype_AlleleSummary
mdp_genotype_SiteSummary

```
Table Title: Marker Statistics - mdp_traits + PC_mdp_genotype + mdp_genotype
Number of columns: 18
Number of rows: 9282
Matrix size (excludes row headers): 157794
MLM statistics for compressed MLM
Dataset: mdp_traits + PC_mdp_genotype + mdp_genotype
Use compression = false
Use P3D = true
P3D = true. Variance components were estimated only for the model without any
markers.
Model: trait = mean + PC5 + PC1 + PC2 + PC3 + PC4 + marker
```

| Trait | Marker | Chr | Pos | df | F | p | add_effect | add_F | add_p | dom_effect | dom_F |
|---|---|---|---|---|---|---|---|---|---|---|---|
| EarHT | None | | | 0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZB00859.1 | 1 | 157104 | 2 | 1.53945 | 0.21641 | -2.5353E0 | 3.01717 | 0.08355 | 4.2881 | 0.15924 |
| EarHT | PZA01271.1 | 1 | 1947984 | 1 | 3.63474 | 0.05768 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA03613.2 | 1 | 2914066 | 1 | 0.99138 | 0.32031 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA03613.1 | 1 | 2914171 | 2 | 0.02612 | 0.97422 | 0.22043 | 0.01996 | 0.88775 | -1.8723E0 | 0.03369 |
| EarHT | PZA03614.2 | 1 | 2915078 | 1 | 1.92736 | 0.16626 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA03614.1 | 1 | 2915242 | 1 | 1.60828 | 0.2059 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA00258.3 | 1 | 2973508 | 2 | 1.31431 | 0.27052 | 1.37685 | 0.76222 | 0.38348 | 17.19671 | 2.00801 |
| EarHT | PZA02962.... | 1 | 3205252 | 2 | 4.12307 | 0.01728 | -1.3763E0 | 0.46068 | 0.49792 | 29.02092 | 8.11606 |
| EarHT | PZA02962.... | 1 | 3205262 | 1 | 0.03348 | 0.85497 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA00599.... | 1 | 3206090 | 2 | 0.32533 | 0.72258 | 0.56944 | 0.07974 | 0.77787 | -1.06E1 | 0.55508 |
| EarHT | PZA02129.1 | 1 | 3706018 | 1 | 1.3341E-5 | 0.99709 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA00393.1 | 1 | 4175293 | 1 | 0.22597 | 0.63493 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA02869.8 | 1 | 4429897 | 1 | 0.16959 | 0.68081 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA02869.4 | 1 | 4429927 | 2 | 0.40918 | 0.66463 | 1.43031 | 0.76304 | 0.38322 | 4.72407 | 0.06251 |
| EarHT | PZA02869.2 | 1 | 4430055 | 2 | 0.66363 | 0.5159 | -4.4926E-1 | 0.07064 | 0.79064 | -2.0399E1 | 1.22934 |
| EarHT | PZA02032.1 | 1 | 4490461 | 1 | 2.60381 | 0.10778 | NaN | NaN | NaN | NaN | NaN |
| EarHT | zagl1.5 | 1 | 4835434 | 0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| EarHT | zagl1.2 | 1 | 4835558 | 1 | 1.60292 | 0.20667 | NaN | NaN | NaN | NaN | NaN |
| EarHT | zagl1.6 | 1 | 4835658 | 1 | 1.57804 | 0.21013 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZD00081.2 | 1 | 4836542 | 1 | 1.52835 | 0.21743 | NaN | NaN | NaN | NaN | NaN |
| EarHT | zagl1.1 | 1 | 4912526 | 0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZB00919.1 | 1 | 5353319 | 1 | 1.23938 | 0.26662 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZB00919.2 | 1 | 5353655 | 1 | 1.97605 | 0.16096 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PHM2244.... | 1 | 5562502 | 2 | 1.18727 | 0.30671 | -1.5282E0 | 0.46345 | 0.49662 | -1.0175E1 | 1.46417 |
| EarHT | PZA03093.... | 1 | 8075572 | 2 | 0.04001 | 0.96079 | -3.342E-1 | 0.07437 | 0.7853 | 0.56886 | 0.00465 |
| EarHT | PZA00181.5 | 1 | 8366368 | 2 | 0.48443 | 0.61662 | -1.4168E0 | 0.41452 | 0.52027 | 3.56875 | 0.82367 |
| EarHT | PZA00181.2 | 1 | 8366411 | 2 | 1.09131 | 0.33725 | -2.1258E0 | 2.0901 | 0.14942 | -1.2663E0 | 0.02936 |
| EarHT | PZA00528.1 | 1 | 8367944 | 1 | 0.07291 | 0.78737 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA00175.2 | 1 | 8510027 | 1 | 3.79021 | 0.05259 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA00447.6 | 1 | 9023947 | 2 | 1.84672 | 0.15987 | -1.8219E0 | 1.75703 | 0.18619 | -1.8746E1 | 1.9194 |
| EarHT | PZA00447.8 | 1 | 9024005 | 1 | 6.04302 | 0.01465 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZB01915.1 | 1 | 9029842 | 1 | 0.83568 | 0.36146 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA03128.1 | 1 | 9084948 | 2 | 0.24368 | 0.78391 | -9.7355E-1 | 0.48358 | 0.48741 | -6.1656E-1 | 0.0012 |
| EarHT | PZA03128.3 | 1 | 9084979 | 1 | 0.59365 | 0.44169 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA02284.1 | 1 | 9273299 | 2 | 1.45157 | 0.23606 | 1.55234 | 1.34929 | 0.24645 | -1.305E1 | 1.47711 |
| EarHT | PZA00731.6 | 1 | 9300391 | 2 | 0.32254 | 0.72459 | -1.2084E0 | 0.6247 | 0.43002 | -2.8967E0 | 0.02635 |
| EarHT | PZA00731.7 | 1 | 9300541 | 2 | 0.18016 | 0.83524 | 0.83623 | 0.3575 | 0.5504 | -1.6109E0 | 0.00816 |
| EarHT | PZA03518.1 | 1 | 10065344 | 1 | 0.41152 | 0.52174 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA03519.2 | 1 | 10065856 | 1 | 0.10254 | 0.74906 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA03519.1 | 1 | 10066126 | 1 | 0.0013 | 0.97126 | NaN | NaN | NaN | NaN | NaN |
| EarHT | PZA03520.1 | 1 | 10067544 | 1 | 0.00862 | 0.9261 | NaN | NaN | NaN | NaN | NaN |

Plot the output (MLM stats file in the Results branch following the steps shown for GLM).

## 12. Exporting results

One may export the results in .txt format by the following the below steps:

Select the file -> go to “File” -> click on “ Save As” ->browse the folder to save the file -> name the file ->click “okay”

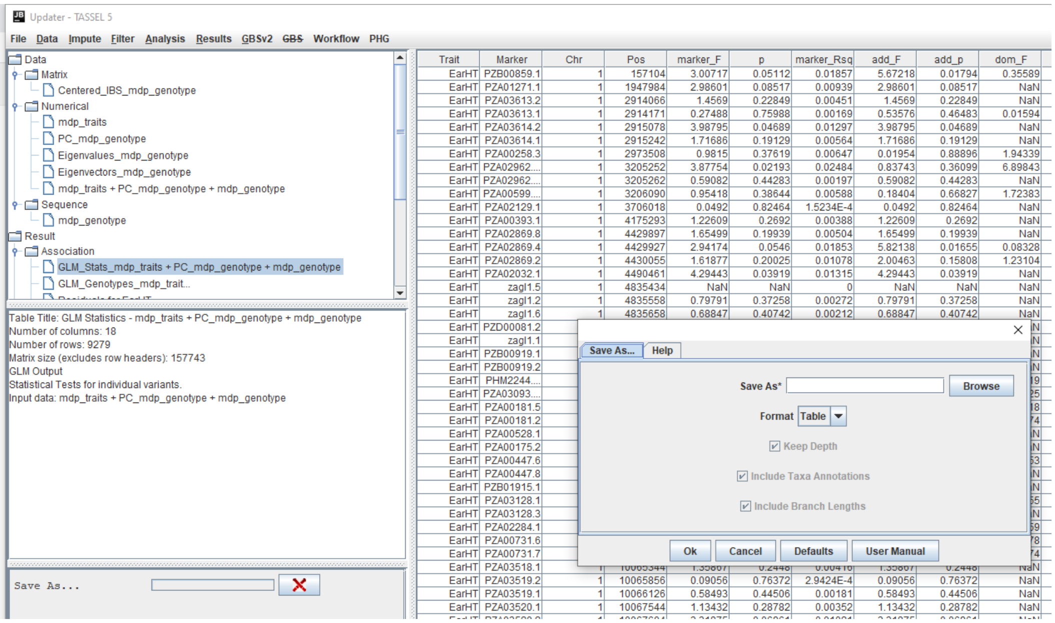

Updater - TASSEL 5
File Data Impute Filter Analysis Results GBSv2 ~~GBS~~ Workflow PHG
Data
Matrix
Centered_IBS_mdp_genotype
Numerical
mdp_traits
PC_mdp_genotype
Eigenvalues_mdp_genotype
Eigenvectors_mdp_genotype
mdp_traits + PC_mdp_genotype + mdp_genotype
Sequence
mdp_genotype
Result
Association
GLM_Stats_mdp_traits + PC_mdp_genotype + mdp_genotype
GLM_Genotypes_mdp_trait...

Table Title: GLM Statistics - mdp_traits + PC_mdp_genotype + mdp_genotype
Number of columns: 18
Number of rows: 9279
Matrix size (excludes row headers): 157743
GLM Output
Statistical Tests for individual variants.
Input data: mdp_traits + PC_mdp_genotype + mdp_genotype

Save As...

| Trait | Marker | Chr | Pos | marker_F | p | marker_Rsq | add_F | add_p | dom_F |
|---|---|---|---|---|---|---|---|---|---|
| EarHT | PZB00859.1 | 1 | 157104 | 3.00717 | 0.05112 | 0.01857 | 5.67218 | 0.01794 | 0.35589 |
| EarHT | PZA01271.1 | 1 | 1947984 | 2.98601 | 0.08517 | 0.00939 | 2.98601 | 0.08517 | NaN |
| EarHT | PZA03613.2 | 1 | 2914066 | 1.4569 | 0.22849 | 0.00451 | 1.4569 | 0.22849 | NaN |
| EarHT | PZA03613.1 | 1 | 2914171 | 0.27488 | 0.75988 | 0.00169 | 0.53576 | 0.46483 | 0.01594 |
| EarHT | PZA03614.2 | 1 | 2915078 | 3.98795 | 0.04689 | 0.01297 | 3.98795 | 0.04689 | NaN |
| EarHT | PZA03614.1 | 1 | 2915242 | 1.71686 | 0.19129 | 0.00564 | 1.71686 | 0.19129 | NaN |
| EarHT | PZA00258.3 | 1 | 2973508 | 0.9815 | 0.37619 | 0.00647 | 0.01954 | 0.88896 | 1.94339 |
| EarHT | PZA02962.... | 1 | 3205252 | 3.87754 | 0.02193 | 0.02484 | 0.83743 | 0.36099 | 6.89843 |
| EarHT | PZA02962.... | 1 | 3205262 | 0.59082 | 0.44283 | 0.00197 | 0.59082 | 0.44283 | NaN |
| EarHT | PZA00599.... | 1 | 3206090 | 0.95418 | 0.38644 | 0.00588 | 0.18404 | 0.66827 | 1.72383 |
| EarHT | PZA02129.1 | 1 | 3706018 | 0.0492 | 0.82464 | 1.5234E-4 | 0.0492 | 0.82464 | NaN |
| EarHT | PZA00393.1 | 1 | 4175293 | 1.22609 | 0.2692 | 0.00388 | 1.22609 | 0.2692 | NaN |
| EarHT | PZA02869.8 | 1 | 4429897 | 1.65499 | 0.19939 | 0.00504 | 1.65499 | 0.19939 | NaN |
| EarHT | PZA02869.4 | 1 | 4429927 | 2.94174 | 0.0546 | 0.01853 | 5.82138 | 0.01655 | 0.08328 |
| EarHT | PZA02869.2 | 1 | 4430055 | 1.61877 | 0.20025 | 0.01078 | 2.00463 | 0.15808 | 1.23104 |
| EarHT | PZA02032.1 | 1 | 4490461 | 4.29443 | 0.03919 | 0.01315 | 4.29443 | 0.03919 | NaN |
| EarHT | zagl1.5 | 1 | 4835434 | NaN | NaN | 0 | NaN | NaN | NaN |
| EarHT | zagl1.2 | 1 | 4835558 | 0.79791 | 0.37258 | 0.00272 | 0.79791 | 0.37258 | NaN |
| EarHT | zagl1.6 | 1 | 4835658 | 0.68847 | 0.40742 | 0.00212 | 0.68847 | 0.40742 | NaN |
| EarHT | PZD00081.2 | | | | | | | | N |
| EarHT | zagl1.1 | | | | | | | | N |
| EarHT | PZB00919.1 | | | | | | | | N |
| EarHT | PZB00919.2 | | | | | | | | N |
| EarHT | PHM2244.... | | | | | | | | 19 |
| EarHT | PZA03093.... | | | | | | | | 25 |
| EarHT | PZA00181.5 | | | | | | | | 18 |
| EarHT | PZA00181.2 | | | | | | | | 74 |
| EarHT | PZA00528.1 | | | | | | | | N |
| EarHT | PZA00175.2 | | | | | | | | N |
| EarHT | PZA00447.6 | | | | | | | | 53 |
| EarHT | PZA00447.8 | | | | | | | | N |
| EarHT | PZB01915.1 | | | | | | | | N |
| EarHT | PZA03128.1 | | | | | | | | 55 |
| EarHT | PZA03128.3 | | | | | | | | N |
| EarHT | PZA02284.1 | | | | | | | | 59 |
| EarHT | PZA00731.6 | | | | | | | | 78 |
| EarHT | PZA00731.7 | | | | | | | | 74 |
| EarHT | PZA03518.1 | 1 | 10065344 | 1.35867 | 0.2448 | 0.00416 | 1.35867 | 0.2448 | NaN |
| EarHT | PZA03519.2 | 1 | 10065856 | 0.09056 | 0.76372 | 2.9424E-4 | 0.09056 | 0.76372 | NaN |
| EarHT | PZA03519.1 | 1 | 10066126 | 0.58493 | 0.44506 | 0.00181 | 0.58493 | 0.44506 | NaN |
| EarHT | PZA03520.1 | 1 | 10067544 | 1.13432 | 0.28782 | 0.00352 | 1.13432 | 0.28782 | NaN |

Save As*

Format Table

☑ Keep Depth

☑ Include Taxa Annotations

☑ Include Branch Lengths

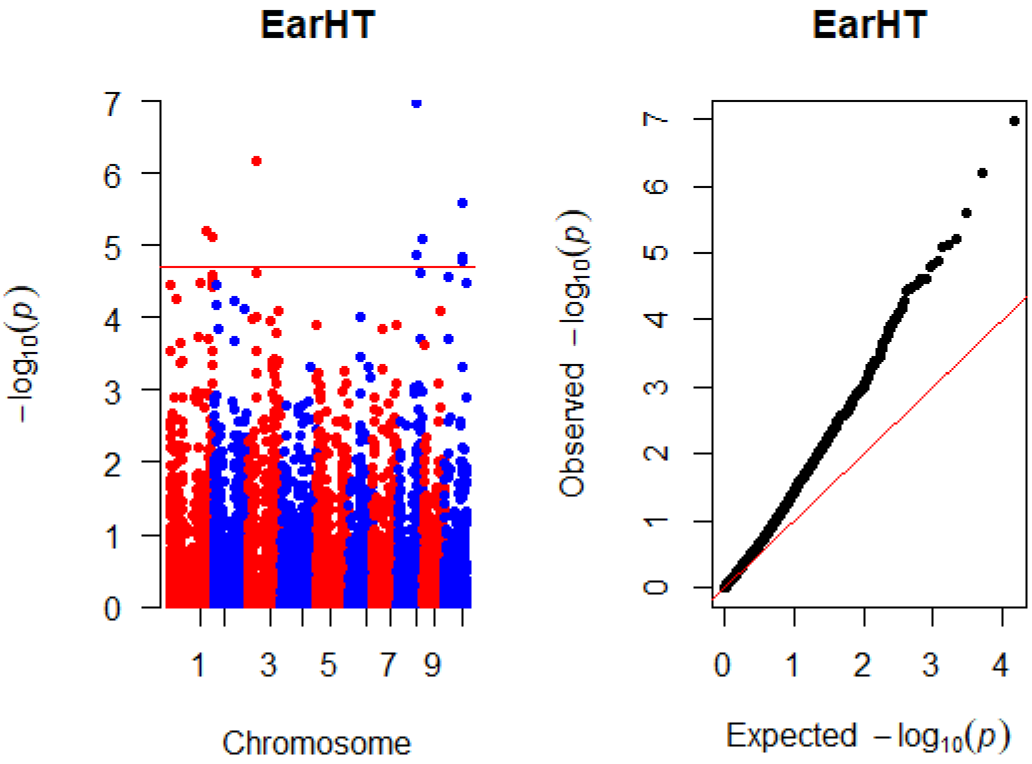
## 13. Plotting GWAS results in R using qqman package

**The R code to plot GWAS result using QQMAN package is below:**

```
library(qqman)

library(dplyr)

# import TASSEL results

# note

TASSEL_MLM_Out <- read.table("mlm_out.txt", header = T, sep = "\t")

# Number of traits

head(unique(TASSEL_MLM_Out$Trait))

# note: for each plot trait name must be specificed

# first trait as example (i.e., EarHT)

Trait1 <-  TASSEL_MLM_Out %>% filter(.$Trait == "EarHT")

# Bonferroni correction threshold

nmrk <- nrow(Trait1)

(GWAS_Bonn_corr_threshold <- -log10(0.05 / nmrk))

# Manhattan plot

(Mann_plot <- manhattan(
  TASSEL_MLM_Out,
  chr = "Chr",
  bp = "Pos",
  snp = "Marker",
  p = "p",
  col = c("red", "blue"),
  annotateTop = T,
  genomewideline = GWAS_Bonn_corr_threshold,
  suggestiveline = F
)
)
# QQ plot
QQ_plot <- qq(TASSEL_MLM_Out$p)
# Manhattan and Q-Q plot arranged in 1 rows and 2 columns
old_par <- par()
par(mfrow=c(1,2))
(Mann_plot <- manhattan(
  TASSEL_MLM_Out,
  chr = "Chr",
  bp = "Pos",
  snp = "Marker",
  p = "p",
  col = c("red", "blue"),
  annotateTop = T,
  genomewideline = GWAS_Bonn_corr_threshold,
  suggestiveline = F,
  main = "EarHT" # trait name
)
```

```
)
(QQ_plot <- qq(TASSEL_MLM_Out$p,  main = "EarHT" ))
```

**The output plot will be as shown below:**



## Reference:

Bradbury, P. J., Zhang, Z., Kroon, D. E., Casstevens, T. M., Ramdoss, Y., & Buckler, E. S. (2007). TASSEL: software for association mapping of complex traits in diverse samples. *Bioinformatics*, *23*(19), 2633-2635.

# An Introduction to Quantitative Trait Loci (QTL) Mapping

**Neeraj Budhlakoti, D. C. Mishra and G. K. Jha**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi, India**

## Background

Quantitative traits exhibit continuous variation due to a combination of polygenic inheritance and environmental influences. Polygenes contribute individually with small effects on the phenotype of the trait, but the cumulative impact of all polygenes affecting a particular trait is significant. Initially postulated to have only additive effects, polygenes are now known to demonstrate dominance and epistatic effects.

This approach aims to identify genomic regions linked to the expression of quantitative traits, referred to as quantitative trait loci (QTL). A QTL can encompass one or more genes influencing the relevant quantitative trait. To conduct QTL analysis, it is crucial to assess the phenotypes of the mapping population at multiple locations. Relying on a single location for evaluation may lead to an underestimation of the total number of QTLs governing the traits in question. Main effect QTLs directly impact the expression of the traits, while epistatic QTLs interact with main effect QTLs, influencing the overall trait phenotype. A major QTL is characterized by explaining 10% or more of the phenotypic variance for the trait, while a QTL with a smaller effect size is termed a minor QTL. The phenotypic effect of a stable QTL remains relatively consistent across environments, making it detectable across different conditions. In contrast, an unstable QTL exhibits variable behavior in different environments.

Typically, major QTLs demonstrate stable expression across various environments, whereas minor QTLs are more susceptible to environmental variations. Metabolic QTLs (mQTLs) govern metabolic traits, such as the rates of diverse metabolic reactions and the levels of metabolites. mQTLs typically exhibit epistatic interactions and possess moderate phenotypic effects. Generally, metabolic traits display lower heritability compared to gene expression levels, and it's noteworthy that eQTLs and mQTLs for a specific trait do not co-localize.Quantitative variation in the cellular content of specific proteins is orchestrated by Protein Quantity QTLs (pQTLs), which have been mapped in various plant species, including maize and wheat. In the case of wheat, pQTLs are distributed throughout the genome, with some affecting proteins associated with membranes. Studies aimed at identifying and mapping eQTLs, mQTLs, and pQTLs that control molecular traits collectively form the field of genetical genomics. This interdisciplinary field contributes to our understanding of the intricate relationships between genetic variations and the regulation of molecular processes in diverse biological systems.

**General Procedure for QTL Mapping:**

The general procedure for Quantitative Trait Loci (QTL) mapping involves a series of methodical steps, each integral to the process. There are four fundamental prerequisites for successful QTL mapping:

- ✓ **Creation of a Suitable Mapping Population:** This involves selecting two homozygous lines with contrasting phenotypes for the trait of interest and crossing them to generate an appropriate mapping population. Preferably, this population should be a doubled haploid (DH) or recombinant inbred line (RIL) population.
- ✓ **Construction of a Dense Marker Linkage Map:** The next step is phenotyping, where the mapping population is assessed for the target trait through replicated trials, ideally over various locations and years.
- ✓ **Reliable Phenotypic Evaluation:** Both parent lines of the mapping population are screened with a large number of genetic markers covering the entire genome to identify polymorphic markers.
- ✓ **Utilization of Appropriate Software for QTL Detection and Mapping:** The entire mapping population is then genotyped using these polymorphic markers.
- ✓ Subsequent steps include:
- ✓ **Linkage Map Construction:** The marker genotype data are utilized to construct a framework linkage map for the population. This map displays the sequence of the markers and the genetic distances between them, measured in centimorgans (cM).
- ✓ **Association Analysis between Marker Genotypes and Trait Phenotypes:** The final step involves analyzing the marker genotype data alongside the trait phenotype data to detect correlations between marker genotypes and the trait phenotype.

This methodology is primarily based on bi-parental populations and is essential for identifying the genetic basis of various traits in species, paving the way for advanced genetic research and breeding programs.

### Methods for QTL Detection and Mapping:

Quantitative Trait Loci (QTL) mapping methods must navigate three critical challenges to ensure accuracy and reliability in their findings:

1. **Inference of QTL Genotypes**: Unlike observable physical traits, the QTL genotypes of different individuals in a population are not directly observable. Hence, these genotypes must be deduced or inferred, often through indirect means such as the analysis of genetic markers.
2. **Selection of an Appropriate Genetic Model**: Given the potential for thousands of loci across the whole genome, selecting an appropriate genetic model for QTL analysis is a complex task. This selection is crucial because the model influences how the data is interpreted and the accuracy of the mapping results. The challenge lies in choosing from a vast array of possible models, each with its own assumptions and implications.
3. **Correlation of Loci on the Same Chromosome**: Loci that are located on the same chromosome tend to be correlated due to linkage. This correlation makes it challenging to separate and individually analyze the effect of each locus, as their effects on the trait may be intertwined.

   To address these issues, QTL analysis methodologies have been developed and can be broadly categorized into two main groups:

   a) **Single QTL Mapping**

   b) **Multiple QTL Mapping**

### Single QTL Mapping

Single QTL mapping methods focus on detecting one Quantitative Trait Locus (QTL) at a time. These approaches do not account for the potential presence of other QTLs in the genome that may also influence the target trait. The two primary methods in this category are:

- ✓ **Single-Marker Analysis (SMA):** Single-marker analysis, also known as single-point analysis, represents the simplest and earliest method used in QTL detection. In this approach, each marker is individually tested for its association with the target trait. A significant difference in the trait between different genotypes at the marker locus suggests that the marker is linked to a QTL influencing the trait. This process is repeated for every marker locus evaluated in the mapping population. The extent of the phenotypic difference between the genotype classes of the marker gives an estimate of the effect caused by substituting a single allele at the QTL locus. A commonly used statistical package for SMA is R/qtl.

- ✓ **Simple Interval Mapping (SIM):** Simple Interval Mapping, initially proposed by Lander and Botstein in 1989, leverages the information from a linkage map. This method assesses the association between trait values and the genotype of a hypothetical QTL (target QTL) at various points between pairs of adjacent marker loci (the target interval). The presence of a putative QTL is inferred if the log of odds (LOD) score exceeds a predetermined critical threshold. Lander and Botstein developed formulas for calculating significance levels appropriate for interval mapping, taking into account factors like genome size, number of chromosomes, number of marker intervals, and the desired overall false positive rate. SIM has become a widely used approach due to its accessibility through statistical packages such as MAPMAKER/QTL.

**Multiple QTL Mapping**

Multiple QTL mapping (MQM) combines multiple regression analysis with SIM to include all the significant QTLs in the genetic model used for mapping (Jansen 1994).

MQM offers the following advantages:

(1) Consideration of other QTLs affecting the trait tends to reduce residual variation

(2) Increase the QTL detection power,

(3) Linked QTLs can be detected as separate QTLs,

(4) The estimates of QTL effects are more reliable than those with single QTL methods

(5) QTL- QTL interaction can be detected. But when too many markers are included as cofactors in the model, the QTL detection power tends to decline in comparison to SIM.

The main multiple QTL mapping methods include

**(1) Composite interval mapping**

**(2) Multiple interval mapping**

**(3) Bayesian multiple QTL mapping**

- ✓ **Composite Interval Mapping (CIM)**

  CIM merges the techniques of interval mapping and multiple regression analysis, as established by Jansen in 1994 and Zeng in 1994. It effectively manages the influence of QTLs found in different marker intervals, both within the same chromosome and across others, enhancing the accuracy of QTL identification. The process begins with an analysis of individual markers, followed by the development of a multi-QTL model using either stepwise or forward regression methods. The model initially incorporates the marker with the highest LOD score, followed by the addition and reevaluation of the marker with the next highest score for its significance.

- ✓ **Multiple Interval Mapping (MIM)**

  Developed by Kao et al. in 1999, MIM facilitates the simultaneous mapping of QTLs across various marker intervals. This method, considered simpler than CIM, maps multiple QTLs at the same time. The genetic model in MIM encompasses the quantity, positions, and interactions (epistasis) of the QTLs.

- ✓ **Bayesian Multiple QTL Mapping**

  This method, designed for identifying multiple QTLs, regards the number of QTLs as a variable subject to random change. It employs a reversible-jump Markov Chain Monte Carlo (MCMC) method for precise modeling, as proposed by Satgopan et al. in 1996 and Banerjee et al. in 2008. Bayesian QTL mapping starts with a chosen prior distribution, from which a posterior distribution is derived to make inferences. Both the CIM and Bayesian methods use maximum likelihood functions for analysis. These methods have been integrated into various software tools, including QTL Cartographer, FlexQTL, INTERQTL, and R/QTLBIM.

**LOD Score and LOD Score Threshold**

The Logarithm of the Odds (LOD) score is a crucial metric in identifying the most probable location of a Quantitative Trait Locus (QTL) in relation to the linkage map. An empirical threshold for the LOD score can be determined using a permutation test, as outlined by Churchill and Doerge in 1994. In this approach, while the marker genotypes of the sample population remain constant, their corresponding trait phenotype values are randomly rearranged. This method helps in assessing the significance of the LOD score by comparing it against a distribution generated from these random shuffles, providing a robust means to discern the true association of a QTL with a specific trait.

***Test of Significance***

LOD > 3 is the significance threshold – 1 in 1,000 the loci are not linked

$$\text{Odds} = \frac{Probability\ of\ Sucess}{Probability\ of\ Failure} = \frac{p}{1-p}$$

Odds = 1 → Equal chance of success and failure

Odds < 1 → Lower chance of success

Odds > 1 → Higher chance of success

### QTL Confidence/Support Interval

The location of a Quantitative Trait Locus (QTL) on a linkage map is typically represented by a bar adjacent to the map. When QTLs associated with different traits are found in the same region, they are indicated by placing additional bars next to each other. The length of these bars symbolizes a range known as the confidence interval or support interval. This interval signifies the probable area where the QTL is situated. It stretches on both sides of the point where the peak of the Logarithm of the Odds (LOD) score is observed, encapsulating the region within which the QTL is most likely to be found.

### Advantages of QTL Mapping

1. QTL mapping detects and map each QTL to short genomic region and identify markers flanking the QTL regions, which can subsequently be used in molecular breeding. Finely mapped QTL facilitates cloning of the genes located in some QTL regions and understanding their functions.
2. QTL analysis provides an estimate of the phenotypic variation explained by a QTL. It helps the breeders in selecting QTL for deployment for crop improvement.

### Disadvantages of QTL mapping

1. The genetic variation for quantitative traits in the bi-parental mapping population used for QTL mapping is limited to the variation present in the parents used. Similarly, alleles studied are also limited to two only.
2. Mapping resolution is low due to limited meiotic cycles. QTL is often mapped to a large genomic region which usually harbors hundreds of genes posing difficulty in identifying the target gene.
3. QTL mapping is difficult in perennial crops; it needs special approach.
4. Identified QTL needs validation which incurs extra cost and time.

### Commonly used Software for QTL mapping

A large number of QTL analysis software is available. For SMA, simple statistical package can work. However, for CIM, MIM, ICIM, etc. different software with suitable algorithm would be required. Name of a few commonly used software are:

**Table 1: Tools and packages for QTL Mapping**

| Tool Name | Description | Interface | URL | References |
|---|---|---|---|---|
| QTL IciMapping | Integrated Software for Building Genetic Linkage Maps and Mapping Quantitative Trait Genes | Written in C# and runs on Windows XP/Vista/7/10, with .NET Framework 4.0. | https://isbreedingen.caas.cn/software/qtllcimapping/294607.htm | Meng *at al.,* 2015 |
| solQTL | Major tool for Solanaceae researchers to perform QTL analysis and dynamically crosslink to relevant genome annotation and genetic expression | Command-line interface (R based) | http://solgenomics.net/qtl/ | Tecle *et al*., 2010 |
| QTL Cartographer | Identifies and maps quantitative trait loci (QTL) in inbred cross populations | Windows menu-driven stand-alone | https://brcwebportal.cos.ncsu.edu/qtlcart/WQTLCart.htm | Wang *et al*., 2012 |
| MapMaker/QTL: | It's widely used in genetic research for analyzing recombination between different markers and for mapping various genetic traits. |  | http://hpcio.cit.nih.gov/lserver/MAPMAKER_Q TL.html | Lander *et al.,* 1987 |
| R/QTL | It is an extensible, interactive environment for mapping quantitative trait loci (QTL) in experimental populations. |  | http://www.rqtl.org | Broman *et al*., 2003 |

### Conclusion and Future Prospects

With the advancement of molecular biological tools, improved techniques, and a deeper understanding of the genome, the concept of Quantitative Trait Loci (QTL) is evolving. The definition of a 'trait' has expanded from the traditional whole-organism phenotype to include more specific phenotypes, such as the quantity of RNA transcript from a particular gene expression (e-QTL) or the amount of protein produced from a specific gene (Protein QTL). The challenge of limited molecular markers or sparsely populated maps has been overcome by leveraging genomic sequences or Single Nucleotide Polymorphisms (SNPs). Similarly, advancements in phenotyping techniques, including proteomics and metabolomics, are addressing the challenges associated with capturing complex trait variations.

Genome-wide Association Studies (GWAS) have gained significant popularity, complementing QTL mapping. Together, QTL mapping and GWAS offer the potential to achieve the ultimate goal: identifying individual genes or nucleotides that contribute to the target phenotype. This integrated approach represents a powerful strategy in the contemporary era of genomics, enabling a more precise understanding of the genetic basis of complex traits and facilitating targeted improvements in various fields, including agriculture and medicine.

## References:

Banerjee S, Yandell BS, Yi N (2008) Bayesian quantitative trait loci mapping for multiple traits. *Genetics*, 179:2275–2289.

Broman, K. W., Wu, H., Sen, Ś., & Churchill, G. A. (2003). R/qtl: QTL mapping in experimental crosses. Bioinformatics, 19(7), 889-890.

Churchill GA, Deorge RW (1994) Empirical threshold values for quantitative trait mapping. *Genetics*, 138:963–971.

Jansen RC (1994) High resolution of quantitative traits into multiple loci via interval mapping. *Genetics*, 136:1447–1455.

Kao CH, Zeng Z-B, Teasdale RD (1999) Multiple interval mapping. *Genetics*, 152:1203–1216.

Lander ES, Botstein D (1989) Mapping mendelian factors underlying quantitative traits using RFLP linkage maps. *Genetics*, 121:185–199.

Lander, E. S., Green, P., Abrahamson, J., Barlow, A., Daly, M. J., Lincoln, S. E., & Newberg, L. A. (1987). MAPMAKER: An interactive computer package for constructing primary genetic linkage maps of experimental and natural populations. Genomics, 1(2), 174-181.

Meng, L., Li, H., Zhang, L., & Wang, J. (2015). QTL IciMapping: Integrated software for genetic linkage map construction and quantitative trait locus mapping in biparental populations. The Crop Journal, 3(3), 269-283.

Satgopan JM, Yandell BS, Newton MA et al (1996) A Bayesian approach to detect quantitative trait loci using Markov chain Monte Carlo. *Genetics*, 144:805–816.

Tecle IY, Menda N, Buels RM, van der Knaap E, & Mueller LA. solQTL: a tool for QTL analysis, visualization and linking to genomes at SGN database. BMC Bioinformatics. 2010; 11(1):525.

Wang, S., Basten, C. J., Zeng, Z. B. (2012). Windows QTL Cartographer 2.5. Department of Statistics, North Carolina State University, Raleigh, NC.

Zeng Z-B (1994) Precision mapping of quantitative trait loci. *Genetics*, 136:1457–1468.

# Genomic Selection: Concept, Methods and Challenges

**Neeraj Budhlakoti[1], Anil Rai[2] and D. C. Mishra[1]**

**[1]ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

**[2]Indian Council of Agricultural Research, New Delhi**

## Abstract

Since the inception of the theory and conceptual framework of genomic selection (GS), extensive research has been done on evaluating its efficiency for utilization in crop improvement. Though marker-assisted selection has proven its potential for improvement of qualitative traits that are controlled by one to few genes with large effects, its role in improving quantitative traits that are controlled by several genes with small effects is limited. In this regard, GS that utilizes genomic-estimated breeding values of individuals obtained from genome-wide markers to choose candidates for the next breeding cycle is a powerful approach to improve quantitative traits. In the past 20 years, GS has been widely adopted in animal breeding programs globally because of its potential to improve selection accuracy, minimize phenotyping, reduce cycle time and increase genetic gains. Improved statistical models that leverage the genomic information to increase the prediction accuracies are critical for the effectiveness of GS-enabled breeding programs.

**Keywords:** *GEBVs, GS, LD, MAS, QTL, SNP.*

## Introduction

As it is known earlier selection based on phenotypic data has been successfully used in past. As abundance of DNA and marker data, trend slightly shifted to marker assisted selection (MAS). MAS is an indirect selection process where a trait of interest is selected, not based on the trait itself, but on a marker linked to it. MAS has been shown to be efficient and effective for traits that are associated with one or a few major genes with large effect but does not perform as well when it is used for selection of polygenic traits (Bernardo 2008).As most economic traits are influenced by many genes, tracking a small number of these through DNA markers will only explain a small proportion of the genetic variance. In addition, individual genes are likely to have small effects and so a large amount of data is needed to accurately estimate their effects. To overcome these difficulties, Meuwissen et al. (2001) proposed a variant of MAS that they called genomic selection. The key features of this method are that markers covering the whole genome are used so that potentially all the genetic variance is explained by the markers and the markers are assumed to be in linkage disequilibrium (LD) with the Quantitative trait loci (QTL), so that the number of effects per QTL to be estimated is small.

Any successful GS program, starts with forming a training population in such a way that individuals/lines/variety are genotyped for genomic markers distributed over entire genome and should be representative of whole population. The training individuals are further subjected to extensive phenotyping for underlying trait of interest. The information of individual genotype and phenotype is used for identification and building of suitable statistical model using phenotype as a response and genotype as independent variable whereas part of training data can also be used for validation of fitted model. Genomic Estimated Breeding Values (GEBVs) of the individuals of the breeding population (where only information of genotyped individuals is available with no phenotypic records) is being calculated using their genotyped information where marker effect are estimated from developed model. Ultimately individuals/line/variety from the breeding population can be selected based on superiority of their estimated value of GEBVs.

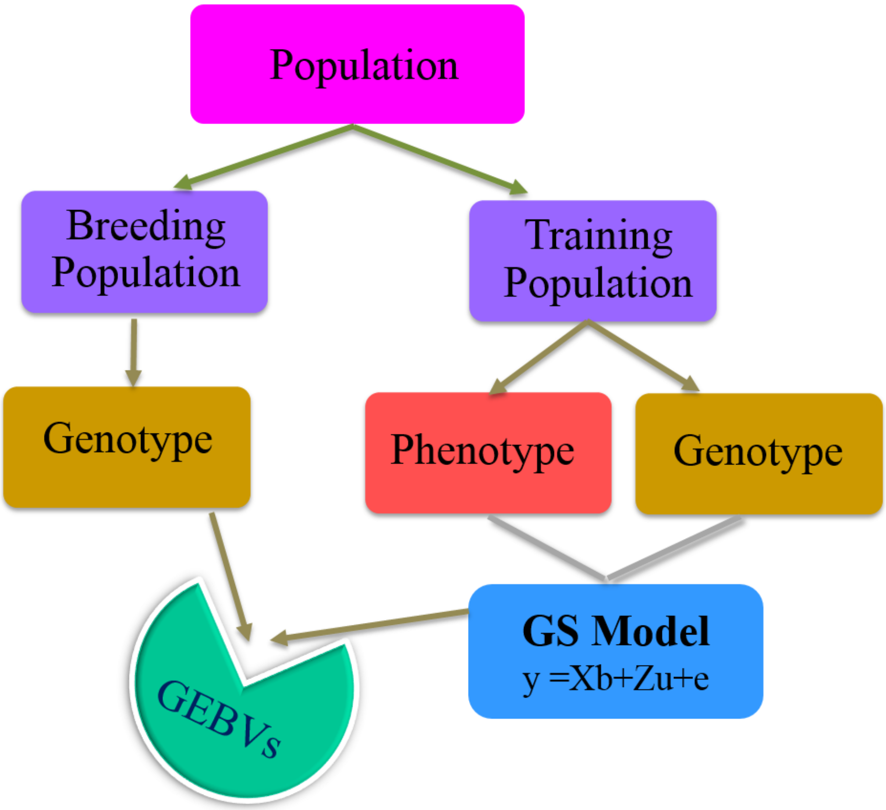


**Fig. 1**: Basic schema of genomic selection process

The major limitation to the implementation of genomic selection has been the large number of markers required and the cost of genotyping these markers are very high. Recently both these limitations have been overcome in most livestock and plant species following the sequencing of the livestock genomes, the subsequent availability of hundreds of thousands of single nucleotide polymorphisms (SNP), and dramatic improvements in development of SNP genotyping technology. Various regression methods have been developed for predicting phenotype. Methods are based on analysis of data consist of genotype and phenotype information. These methods are primarily based on linear models, which are easy to interpret and able to fit to the data without over fitting. However, the relationship between breeding value and genetic markers is likely to be more complex than a simple linear relationship, particularly when large numbers of SNPs are fitted simultaneously in the model. To answer these issues, model-free or so-called nonparametric methods which side-step linearity and require lesser genetic assumptions have gained more attention (Gianola et al, 2006).

## Statistical model for Genomic Selection

Process of selecting the suitable individuals in GS starts with a simple linear model sometime also called as least squares regression or ordinary least squares regression (OLS).

$$Y = 1_n \mu + X\beta + \varepsilon$$

where, $\mathbf{Y} = n \times 1$ vector of observations; μ is the mean; $\boldsymbol{\beta} = p \times 1$ vector of marker effects; $\varepsilon = n \times 1$ vector of random residual effects; $\boldsymbol{X} =$ design matrix of order $n \times p$ (where each row represents the genotype/individuals/lines (n) and column corresponds to marker (p)), $\varepsilon \sim N(0, \sigma_e^2)$.

One major problem in linear models using several thousands of genome-wide markers is that number of markers (p) exceed the number of observations (n) i.e. genotype/individuals/lines and this creates the problem of over-parameterization (large ‘p’ and small ‘n’ problem (p>>n)). Using a subset of the significant markers can be an alternative for dealing with large ‘p’ and small ‘n’ problem. Meuwissen et al. (2001) used a modification of the least squares regression for GS. They performed least squares regression analysis on each maker separately with following model

$$Y = X_j \beta_j + \varepsilon$$

where,

$X_j$ = $j^{th}$column of the design matrix of marker

$\beta_j$ = genetic effect of $j^{th}$ marker

Marker with significant effects are selected using the log likelihood of this model and those are further used for estimation of breeding values. However, it has to be noted that some crucial or key information may be lost by selection based on subset of markers.

Hence, an efficient solution for the over-parameterization problem in linear models is using ridge regression (RR), which is a penalized regression-based approach (Meuwissen et al., 2001). It also solves the problems of multicollinearity at the same time (i.e. correlated predictors e.g. SNP or markers). RR shrinks the coefficients of correlated predictors equally towards zero and solves the regression problem using $\ell 2$ penalized least squares. Here, the goal is to derive an estimator of parameter $\beta$ with smaller variance than the least square estimator. Similar to RR, least absolute shrinkage and selection operator (LASSO) (Tibshirani, 1996; Usai et al., 2009) is other variant of penalized regression, which uses the $\ell_1$ penalized least squares criterion to obtain a sparse solution. LASSO sometime may not work well highly correlated predictors (e.g. SNPs in high linkage disequilibrium) (Ogutu et al., 2012). The elastic net (ENET) is an extension of the lasso that is robust to extreme correlations among the predictors (Friedman et al., 2010) and it is a compromise between $\ell 1$ penalty (lasso) and $\ell 2$ penalty (ridge regression) (Zou and Hastie, 2005).

The RR model considers that each marker contribute to equal variance, which is not the case for all traits. Therefore, the variance of the markers based on the trait genetic architecture has to be modeled. For this purpose, several Bayesian models have been proposed where it is assumed that there is some prior distribution of marker effects. Further, inferences about model parameters are obtained on the basis of posterior distributions of the marker effects. There are several variants of Bayesian models for genomic prediction such as Bayes A, Bayes B, Bayes Cπ and Bayes Dπ (Meuwissen et al., 2001; Habier et al., 2011) and other derivatives e.g. Bayesian LASSO, Bayesian ridge regression (BRR). Besides the marker-based models, the best linear unbiased prediction (BLUP), is one of the most commonly used genomic prediction method. There are many variants of BLUP available for this purpose e.g. genomic BLUP (GBLUP), single-step GBLUP (ssGBLUP), ridge regression BLUP (RRBLUP), GBLUP with linear ridge kernel regression (rrGBLUP), of which is GBLUP is very frequently used. While the BLUP has been used in other plant and animal breeding studies traditionally for various purposes (Henderson et al., 1959), the GBLUP uses the genomic relationships calculated using markers instead of the conventional pedigree-based BLUP which uses the pedigree relationships to obtain the GEBVs of the lines or individuals (Meuwissen et al., 2001).

The genomic prediction models discussed so far perform well for traits with additive genetic architecture but their performance becomes very poor in case of epistatic genetic architectures. Hence, Gianola et al. (2006) first used nonparametric and semiparametric methods for modeling complex genetic architecture. Subsequently, several statistical methods were implemented to model both main and epistatic effects for genomic selection (Xu, 2007; Cai et al., 2011; Legarra and Reverter, 2018). There are several nonparametric methods have been studied in relation to genomic selection e.g. NW (Nadaraya-Watson) estimator (Gianola et al., 2006), RKHS (Reproductive Kernel Hilbert Space) (Gianola et al., 2006), SVM (support vector machine) (Maenhout et al., 2007; Long et al., 2011), ANN (Artificial Neural Network) (Gianola et al., 2011) and RF (Random Forest) (Holliday et al., 2012) among them nonparametric methods SVM, NN and RF are based on machine learning approach.

Methods discussed earlier in this section are based on genomic information where information is available for single-trait i.e. single-trait genomic selection (STGS). As performance of STGS based methods may be affected significantly in case of pleiotropy i.e., one gene linked to multiple traits. A mutation in a pleiotropic gene may have an effect on several traits simultaneously. It was also observed that low heritability traits can borrow information from correlated traits and consequently achieve higher prediction accuracy can be achieved. Also STGS based methods considers the information of each trait independently. Hence we may lose crucial information which may ultimately result in poor genomic prediction accuracy. Now-a-days we are also getting data on multiple traits, so multi-trait genomic selection (MTGS) based methods may provide more accurate GEBVs and subsequently the higher prediction accuracy. Several MTGS based methods have been studied in relation to GS e.g. Multivariate mixed model approach (Jia and Jannink, 2012; Klápště et al., 2020), Bayesian multi-trait model (Jia and Jannink, 2012; Cheng et al., 2018), MRCE (Multivariate Regression with Covariance Estimation)(Rothman et al., 2010), cGGM (conditional Gaussian Graphical Models) (Chiquet et al., 2017). Jia et al. (2012) presented three multivariate linear models (i.e., GBLUP, Bayes A, and Bayes Cπ) and compared them to uni-variate models and a detailed comparison of various STGS and MTGS based methods has also been studied by Budhlakoti et al. (2019). A brief structure of different STGS and MTGS based methods used in GS studies are given in Fig. 2.

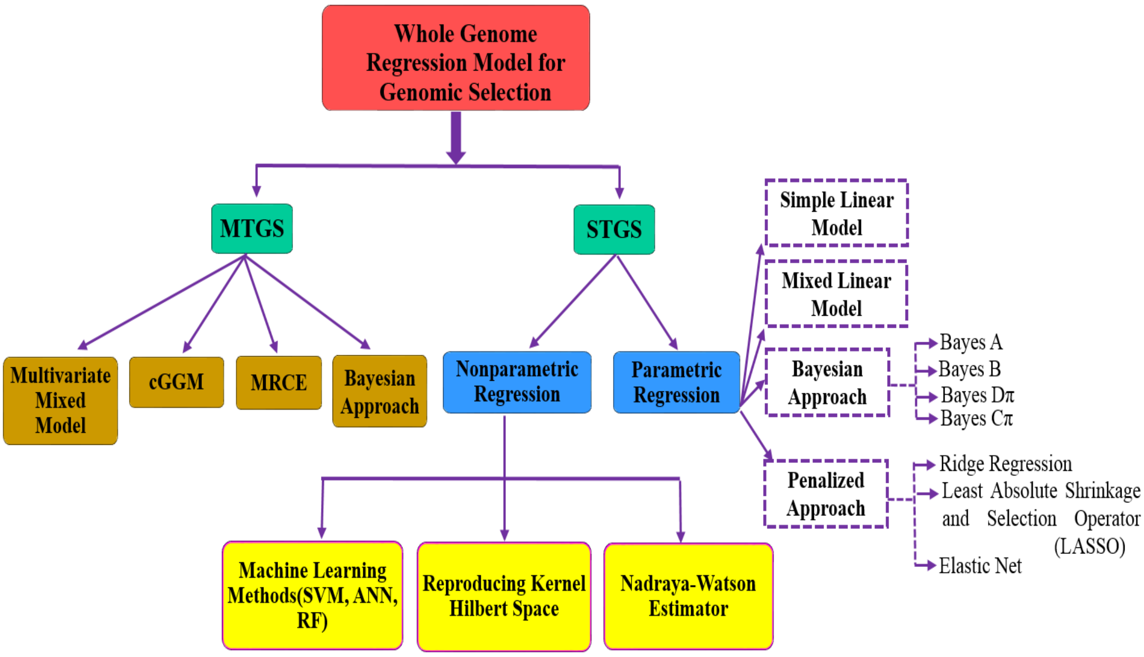


**Fig. 2:** Overall summary of the most commonly used models in Genomic Selection

## Tools and packages to implement Genomic Selection

Several tools and packages have been developed for the evaluation of genomic prediction and implementation of GS, some of which are discussed below.

| Tools/Packages | Description | URL | Reference |
|---|---|---|---|
| GMStool | It is a genome-wide association study (GWAS)-based tool for genomic prediction using genome-wide marker data | https://github.com/JaeYoonKim72/GMStool | Jeong et al. (2020) |
| rrBLUP | R package based on BLUP models its and other derivatives | https://CRAN.R-project.org/package=rrBLUP | Endelman, (2011) |
| BWGS | It has a wide choice of totally 15 parametric and nonparametric statistical models for estimation of GEBV for selection candidates. | https://CRAN.R-project.org/package=BWGS | Charmet et al. (2020) |
| BGLR | This package is an extension of the BLR package (Perezand Campos, 2014) and can be used to implement several Bayesian models | https://CRAN.R-project.org/package=BGLR | Perez and Campos, (2014) |
| GenSel | Used for estimation of molecular marker–based breeding values of animals for trait under evaluation | https://github.com/austin-putz/GenSel | Fernando and |

|  |  |  |  |
|---|---|---|---|
|  |  |  | Garrick, (2009) |
| lme4GS | This package can be used for fitting mixed models with covariance structures with user defined parameter | https://github.com/perpdgo/lme4GS | Caamal-Pat et al. (2021) |
| GSelection | Package comprises of a set of functions to select the important markers and estimates the GEBV of selection candidates using an integrated model framework | https:// CRAN.R-project.org/package =GSelection | Majumdar et al. (2019) |
| STGS | It is a comprehensive package which gives a single-step solution for genomic selection based on most commonly used statistical methods (i.e., RR, BLUP, LASSO, SVM, ANN, and RF). | https://CRAN.Rproject. org/package=STGS | Budhlakoti et al. (2019a) |
| MTGS | MTGS is a comprehensive package which gives a single-step solution for genomic selection using various MTGS-based methods (MRCE, MLASSO, i.e., multivariate LASSO, and KMLASSO, i.e., kernelized multivariate LASSO). | https://CRAN.R-project.org/ package=MTGS | Budhlakoti et al. (2019) |

## Issues and challenges in genomic selection

Genomic selection is a powerful tool for plant and animal breeding, but it also presents a number of challenges and issues. Some of the key challenges and issues in genomic selection include:

1. Data quality and quantity: Genomic selection requires large amounts of high-quality genomic data. However, obtaining this data can be challenging, especially in species with complex genomes or limited genomic resources.

2. Genetic diversity: Genomic selection works best when there is a large amount of genetic diversity in the population. However, in some species, there may be limited genetic diversity, which can limit the effectiveness of genomic selection.

3. Phenotyping: In order to train genomic selection models, accurate and consistent phenotypic data is required. However, phenotyping can be time-consuming, expensive, and difficult to standardize.

4. Trait heritability: The effectiveness of genomic selection depends on the heritability of the trait being selected. Some traits may have low heritability, making it difficult to accurately predict their values using genomic data.

5. Statistical model used: The choice of statistical model used in genomic selection is important because it can impact the accuracy of the predictions and the efficiency of the analysis. Some of the key concerns related to the type of statistical model used in genomic selection include:

   i. Overfitting: Overfitting can occur when a model is too complex for the data, leading to high accuracy in the training set but poor performance on new data. This can be a concern in genomic selection, particularly when using models with a large number of parameters or when the sample size is small.

   ii. Model assumptions: Different statistical models have different assumptions about the data, and violating these assumptions can lead to biased or inaccurate predictions. For example, linear regression assumes that the residuals are normally distributed and homoscedastic, and violating these assumptions can lead to poor performance.

   iii. Scalability: Some statistical models are computationally intensive and may not be feasible for very large datasets. This can be a concern in genomic selection, particularly as the amount of genomic data continues to grow.

   iv. Interpretability: Some statistical models are more interpretable than others, which can be important for understanding the biological basis of the trait being predicted. For example, linear regression models can provide insight into which genomic regions are associated with the trait, while more complex models may be more difficult to interpret.

   v. Incorporation of external information: Some statistical models can incorporate external information, such as gene annotation or pathway information, to improve predictions. However, the quality and relevance of this external information can impact the performance of the model.

6. Integration with traditional breeding: Genomic selection is most effective when it is integrated with traditional breeding methods. However, this can be challenging, especially in species with long breeding cycles or complex genetic architectures.

## Conclusion and perspectives

Genomic selection has improved genetic gains in plant and animal breeding research over the past two decades. Advances in cheaper next-generation sequencing technologies have resulted in the availability of high-density SNP genotyping chips and completely sequenced crop and animal genomes, boosting the predictive ability of a genomic selection model. However, there is still scope for improvement in the methodology of genomic selection, such as imputation of missing genotypic value and implementation of GxE interaction, to successfully implement it in breeding programs. Regular updating of the training set and evaluation under controlled conditions is necessary for better performance. To achieve fruitful outcomes, a structured program is needed that includes human resource development, advanced data recording methodologies, and trait phenotyping.

## Reference

Bernardo, R. (2008). Molecular markers and selection for complex traits in plants: Learning from the last 20 years. Crop Science 48, 1649–1664. doi:10.2135/CROPSCI2008.03.0131.

Budhlakoti, N., Mishra, D. C., Rai, A., Lal, S. B., Chaturvedi, K. K., and Kumar, R. R. (2019). A Comparative Study of Single-Trait and Multi-Trait Genomic Selection. Journal of Computational Biology 26, 1100–1112. doi:10.1089/CMB.2019.0032.

Budhlakoti, N, Mishra, D. C., Rai, A. and Chaturvedi, K.K. (2019a) Package ‘STGS’, 1-11.

Budhlakoti, N., Mishra, D. C., and Rai, A. (2019b). Package ‘MTGS’, 1–6.

Caamal-Pat, D., Pérez-Rodríguez, P., Crossa, J., Velasco-Cruz, C., Pérez-Elizalde, S., and Vázquez-Peña, M. (2021). lme4GS: An R-Package for Genomic Selection. Frontiers in Genetics 12, 982. doi:10.3389/FGENE.2021.680569/BIBTEX.

Cai, X., Huang, A., and Xu, S. (2011). Fast empirical Bayesian LASSO for multiple quantitative trait locus mapping. BMC Bioinformatics 12, 1–13. doi:10.1186/1471-2105-12-211/FIGURES/5.

Charmet, G., Tran, L. G., Auzanneau, J., Rincent, R., and Bouchet, S. (2020). BWGS: A R package for genomic selection and its application to a wheat breeding programme. PLOS ONE 15, e0222733. doi:10.1371/JOURNAL.PONE.0222733.

Cheng, H., Kizilkaya, K., Zeng, J., Garrick, D., and Fernando, R. (2018). Genomic prediction from multiple-trait Bayesian regression methods using mixture priors. Genetics 209, 89–103. doi:10.1534/GENETICS.118.300650/-/DC1.

Chiquet, J., Mary-Huard, T., St´, S., and Robin, S. (2017). Structured regularization for conditional Gaussian graphical models. Statistics and Computing 27, 789-804.

Endelman, J. B. (2011). Ridge Regression and Other Kernels for Genomic Selection with R Package rrBLUP. The Plant Genome 4, 250–255. doi:10.3835/PLANTGENOME2011.08.0024.

Fernando, R. and Garrick, D. (2009). GenSel- User Manual for a portfolio of Genomic Selection related Analyses. (http://taurus.ansci.iastate.edu/Site/Welcome_files/GenSel%20 Manual%20v2.pdf)

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of statistical software 33, 1.

Gianola, D., Fernando, R. L., and Stella, A. (2006). Genomic-Assisted Prediction of Genetic Value With Semiparametric Procedures. Genetics 173, 1761. doi:10.1534/GENETICS.105.049510.

Gianola, D., Okut, H., Weigel, K. A., and Rosa, G. J. M. (2011). Predicting complex quantitative traits with Bayesian neural networks: a case study with Jersey cows and wheat. BMC Genet. 12, 87. doi:10.1186/1471-2156-12-87.

Habier, D., Fernando, R. L., Kizilkaya, K., and Garrick, D. J. (2011). Extension of the bayesian alphabet for genomic selection. BMC Bioinformatics 12, 1–12. doi:10.1186/1471-2105-12-186/FIGURES/2.

Henderson, C. R., Kempthorne, O., Searle, S. R. and von Krosigk, C. M. (1959). The estimation of environmental and genetic trends from records subject to culling. Biometrics, 15: 192.

Holliday, J. A., Wang, T., and Aitken, S. (2012). Predicting Adaptive Phenotypes From Multilocus Genotypes in Sitka Spruce (Picea sitchensis) Using Random Forest. doi:10.1534/g3.112.002733.

Jeong, S., Kim, J. Y., and Kim, N. (2020). GMStool: GWAS-based marker selection tool for genomic prediction from genomic data. Scientific Reports 10, 1–12. doi:10.1038/s41598-020-76759-y.

Jia, Y., and Jannink, J. L. (2012). Multiple-Trait Genomic Selection Methods Increase Genetic Value Prediction Accuracy. Genetics 192, 1513. doi:10.1534/GENETICS.112.144246.

Klápště, J., Dungey, H. S., Telfer, E. J., Suontama, M., Graham, N. J., Li, Y., et al. (2020). Marker Selection in Multivariate Genomic Prediction Improves Accuracy of Low Heritability Traits. Front. Genet. 11, 499094. doi:10.3389/FGENE.2020.499094/FULL.

Legarra, A., and Reverter, A. (2018). Semi-parametric estimates of population accuracy and bias of predictions of breeding values and future phenotypes using the LR method 01 Mathematical Sciences 0104 Statistics. Genetics Selection Evolution 50, 1–18. doi:10.1186/S12711-018-0426-6/FIGURES/3.

Long, N., Gianola, D., Rosa, G. J. M., and Weigel, K. A. (2011). Application of support vector regression to genome-assisted prediction of quantitative traits. Theor. Appl. Genet. 123, 1065–1074. doi:10.1007/S00122-011-1648-Y.

Maenhout, S., De Baets, B., Haesaert, G., and Van Bockstaele, E. (2007). Support vector machine regression for the prediction of maize hybrid performance. Theor. Appl. Genet. 115, 1003–1013. doi:10.1007/s00122-007-0627-9.

Majumdar, S. G., Rai, A., and Mishra, D. C. (2019). Package ‘GSelection’, 1–14.Available at: https://rdrr.io/cran/GSelection/man/GSelection-package.html.

Meuwissen, T. H. E., Hayes, B. J., and Goddard, M. E. (2001). Prediction of total genetic value using genome-wide dense marker maps. Genetics 157, 1819–1829. doi:10.1093/GENETICS/157.4.1819.

Ogutu, J. O., Schulz-Streeck, T., and Piepho, H. P. (2012). Genomic selection using regularized linear regression models: ridge regression, lasso, elastic net and their extensions. BMC Proc. 6, S10. doi:10.1186/1753-6561-6-S2-S10.

Perez, P., and Campos, G. (2014). BGLR : A Statistical Package for Whole Genome Regression and Prediction. Genetics 198, 483–495.

Rothman, A. J., Levina, E., and Zhu, J. (2010). Sparse Multivariate Regression With Covariance Estimation. J. Comput. Graph. Stat. 19, 947. doi:10.1198/JCGS.2010.09188.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. Journal of Royal Statistical Society, 58: 267–288.

Usai, M. G., Goddard, M. E., and Hayes, B. J. (2009). LASSO with cross-validation for genomic selection. Genet. Res. (Camb). 91, 427–436. doi:10.1017/S0016672309990334.

Xu, S. (2007). An Empirical Bayes Method for Estimating Epistatic Effects of Quantitative Trait Loci. Biometrics 63, 513–521. doi:10.1111/J.1541-0420.2006.00711.X.

Zou, H., and Hastie, T. (2005). Regularization and variable selection via the elastic net. J. R. Stat. Soc. B 67, 301–320.

# Transcriptomic Data Analysis

**Mohammad Samir Farooqi and Sudhir Srivastava**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

The advent of Next-Generation Sequencing (NGS) technology has transformed genomic studies. One important application of NGS technology is the study of the *transcriptome*, which is defined as the complete collection of all the RNA molecules in a cell. Various types of RNA that have been classified so far are shown in **Fig. 1**. All of these molecules are called *transcripts* since they are produced by process of transcription.

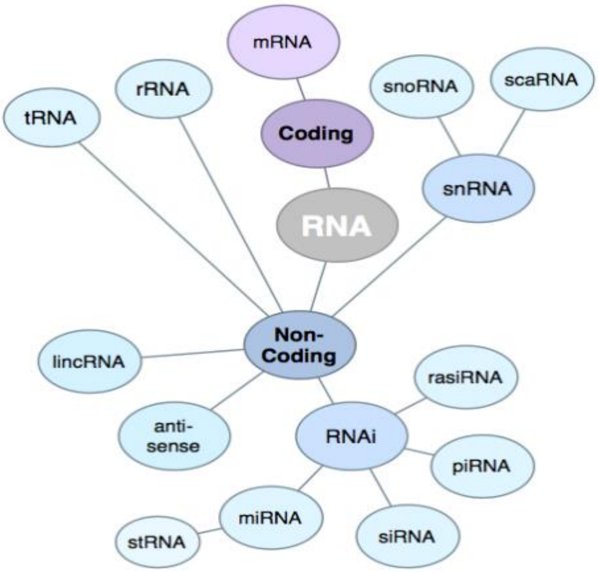


Fig. 1: Different types of RNA

(Image source: http://scienceblogs.com/digitalbio/2011/01/08/next-gene-sequencing)

Understanding the transcriptome is essential for interpreting the functional elements of the genome and revealing the molecular constituents of cells and tissues, and also for understanding development and disease [1]. The main purpose of transcriptomics are: to catalogue all species of transcript, including mRNAs, non-coding RNAs and small RNAs; to determine the transcriptional structure of genes, in terms of their start sites, 5′ and 3′ ends, splicing patterns and other post-transcriptional modifications; and to quantify the changing expression levels of each transcript during development and under different conditions.

The study of transcriptome is carried out through sequencing of RNAs. *RNA sequencing (RNA-Seq)* is a powerful method for discovering, profiling, and quantifying RNA transcripts [2]. RNA-Seq uses NGS datasets to obtain sequence reads from millions of individual RNAs. The RNA-Seq analysis is performed in several steps: First, all genes are extracted from the reference genome (using annotations of type *gene*). Other annotations on the gene sequences are preserved (e.g.CDS information about coding sequences etc). Next, all annotated transcripts (using annotations of type *mRNA*) are extracted [3]. If there are several annotated splice variants, they are all extracted. An example is shown in below **Fig. 2(a).**

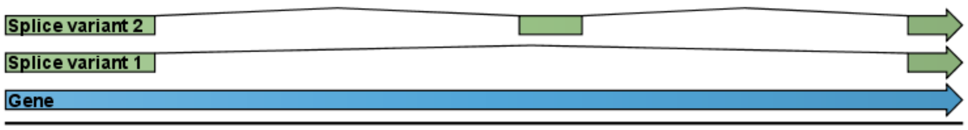


**Fig. 2(a): A simple gene with three exons and two splice variants.**

The given example is a simple gene with three exons and two splice variants. The transcripts are extracted as shown in **Fig. 2(b).**

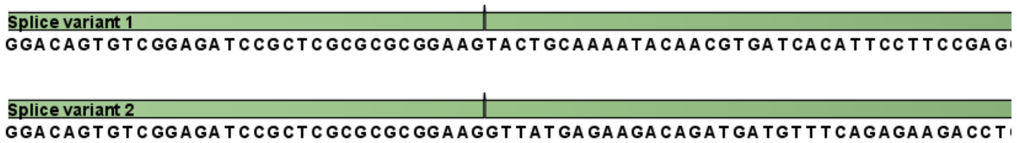

Splice variant 1
GGACAGTGTCGGAGATCCGCTCGCGCGCGGAAGTACTGCAAAATACAACGTGATCACATTCCTTCCGAG
Splice variant 2
GGACAGTGTCGGAGATCCGCTCGCGCGCGGAAGGTTATGAGAAGACAGATGATGTTTCAGAGAAGACCT

**Fig. 2(b): All the exon-exon junctions are joined in the extracted transcript.**

Next, the reads are mapped against all the transcripts plus the entire gene [see **Fig. 2(c)**].

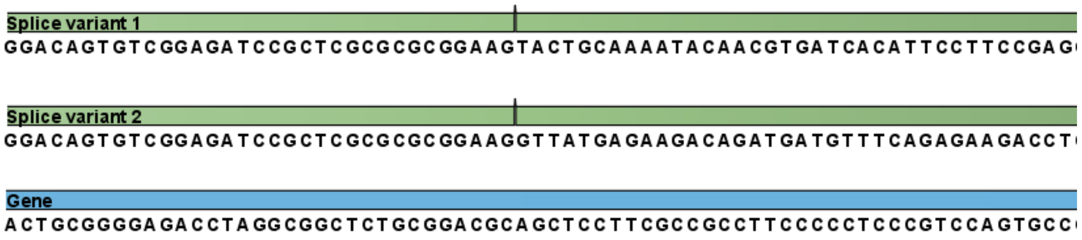

Splice variant 1
GGACAGTGTCGGAGATCCGCTCGCGCGCGGAAGTACTGCAAAATACAACGTGATCACATTCCTTCCGAG
Splice variant 2
GGACAGTGTCGGAGATCCGCTCGCGCGCGGAAGGTTATGAGAAGACAGATGATGTTTCAGAGAAGACCT
Gene
ACTGCGGGGAGACCTAGGCGGCTCTGCGGACGCAGCTCCTTCGCCGCCTTCCCCCTCCCGTCCAGTGCC

Fig. 2(c): The reference for mapping: all the exon-exon junctions and the gene

(Image source: CLC Genomic workbench tutorials)

From this mapping, the reads are categorized and assigned to the genes and expression values for each gene and each transcript are calculated and putative exons are then identified.

### RNA Sequencing Experiment

In a standard RNA-seq experiment, a sample of RNA is converted to a library of complementary DNA fragments and then sequenced on a high-throughput sequencing platform, such as Illumina's Genome Analyzer, SOLiD or Roche 454 [4]. Millions of short sequences, or reads, are obtained from this sequencing and then mapped to a reference genome (**Fig. 3**). The count of reads mapped to a given gene measures the expression level of this gene. The unmapped reads are usually discarded and mapped reads for each sample are assembled into gene-level, exon-level or transcript-level expression summaries, depending on the objectives of the experiment. The count of reads mapped to a given gene/exon/transcript measures the expression level for this region of the genome or transcriptome.

One of the primary goals for most RNA-seq experiments is to compare the gene expression levels across various treatments. A simple and common RNA-seq study involves two treatments in a randomized complete design, for example, treated versus untreated cells, two different tissues from an organism, plants, etc. In most of the studies, researchers are particularly interested in detecting gene with differential expressions (DE). A gene is declared differentially expressed if an observed difference or change in read counts between two experimental conditions is statistically significant, i.e. if the difference is greater than what would be expected just due to random variation [5]. Detecting DE genes can also be an important pre-step for subsequent studies, such as clustering gene expression profiles or testing gene set enrichments.

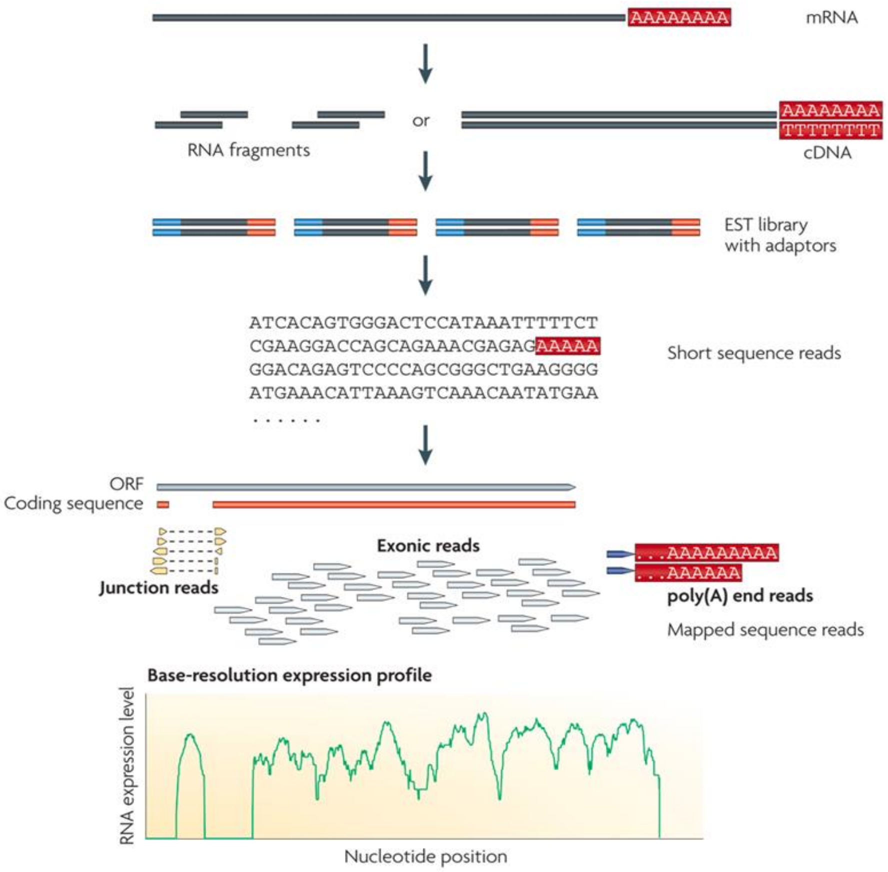


Fig. 3: General RNA-seq experiment. mRNA is converted to cDNA, and fragments from that library are used to generate short sequence reads. Those reads are assembled into contigs which may be mapped to reference sequences (Wang et al., 2009)

### Analysing RNA-Seq data

RNA-seq experiments must be analyzed with robust, efficient and statistically correct algorithms. Fortunately, the bioinformatics community has been striving hard at work for incorporating mathematics, statistics and computer science for RNA-seq and building these ideas into software tools. RNA-seq analysis tools generally fall into three categories: (i) those for read alignment; (ii) those for transcript assembly or genome annotation; and (iii) those for transcript and gene quantification. Some of the open source softwares available for RNA-seq analysis are as follows:

- **Data preprocessing**
    - Fastx toolkit
    - Samtools
- **Short reads aligners**

    - Bowtie, TOPHAT, Stampy, BWA, Novoalign, etc
- **Expression studies**
    - Cufflinks package
    - R packages (DESeq, edgeR, *more…*)
- **Visualisation**
    - CummeRbund, IGV, Bedtools, UCSC Genome Browser, etc.

Besides there are commercially data analysis pipelines like GenomeQuest, CLCBio etc available for researchers to use. The most commonly used pipeline is to identify protein coding genes by aligning RNA-Seq data to annotate data from sources like RefSeq. After generating the alignments, the number of aligning sequences is counted for each position. Since each alignment represents a transcript, the alignments allow to count the number of RNA molecules produced from every gene.

Using NGS technology, RNA-Seq enables to count the number of reads that align to one of thousands of different cDNAs, producing results similar to those of gene expression microarrays [6]. Sequences generated from an RNA-Seq experiment are usually mapped to libraries of known exons in known transcripts. RNA-Seq can be used for discovery applications such as identifying alternative splicing events, allele-specific expression, and rare and novel transcripts [7]. The sequencing output files (compressed FASTQ files) are the input for secondary analysis. Reads are aligned to an annotated reference genome, and those aligning to exons, genes and splice junctions are counted. The final steps are data visualisation and interpretation, consisting of calculating gene- and transcript-expression and reporting differential expression. A general Bioinformatics workflow to map transcripts from RNA-seq data is shown in **Fig. 4**.

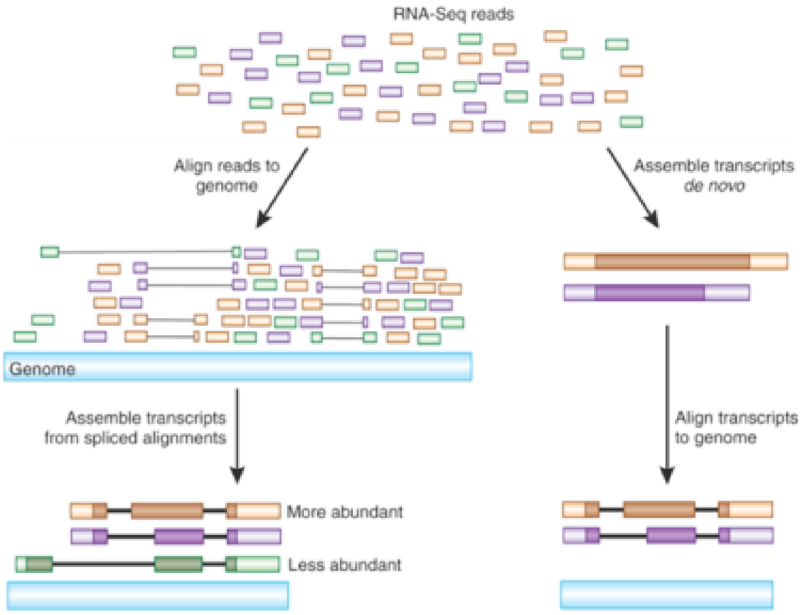

RNA-Seq reads
Align reads to genome
Assemble transcripts *de novo*
Genome
Assemble transcripts from spliced alignments
Align transcripts to genome
More abundant
Less abundant

Fig. 4: RNA-seq workflow (Adapted from Advancing RNA-Seq analysis Brian J. Haas and Michael C. Zody Nature Biotechnology 28, 421-423 (2010)

### RPKM (Reads per KB per million reads)

RNA-Seq provides quantitative approximations of the abundance of target transcripts in the form of counts. However, these counts must be normalized to remove technical biases inherent in the preparation steps for RNA-Seq, in particular the length of the RNA species and the sequencing depth of a sample. The most commonly used is RPKM (Reads Per Kilobase of exon model per Million mapped reads). The RPKM measure of read density reflects the molar concentration of a transcript in the starting sample by normalizing for RNA length and for the total read number in the measurement [8]. RPKM is mathematically represented as:

$$\text{RPKM} = \frac{total\ exon\ reads}{mapped\ reads\ (millions)\ \text{X}\ exon\ length\ (KB)}$$

### Total exon reads

This is the number of reads that have been mapped to a region in which an exon is annotated for the gene or across the boundaries of two exons or an intron and an exon for an annotated transcript of the gene. For eukaryotes, exons and their internal relationships are defined by annotations of type mRNA.

### Exon length

This is calculated as the sum of the lengths of all exons annotated for the gene. Each exon is included only once in this sum, even if it is present in more annotated transcripts for the gene. Partly overlapping exons will count with their full length, even though they share the same region.

### Mapped reads

The total gene reads for a gene is the total number of reads that after mapping have been mapped to the region of the gene. A gene's region is that comprised of the flanking regions, the exons, the introns and across exon-exon boundaries of all transcripts annotated for the gene. Thus, the sum of the total gene reads numbers is the number of mapped reads for the sample.

### Applications of RNA-seq

This technique can be used to:

- Measure gene expression
- Transcriptome assembly, gene discovery and annotation
- Detect differential transcript abundances between tissues, developmental stages, genetic backgrounds, and environmental conditions
- Characterize alternative splicing, alternative polyadenylation, and alternative transcription.

### Future Directions

Although RNA-Seq is still in the infancy stages of use, it has clear advantages over previously developed transcriptomic methods. Compared with microarray, which has been the dominant approach of studying gene expression in the last two decades, RNA-seq technology has a wider measurable range of expression levels, less noise, higher throughput, and more information to detect allele-specific expression, novel promoters, and isoforms [9]. For these reasons, RNA-seq is gradually replacing the array-based approach as the major platform in gene expression studies. The next big challenge for RNA-Seq is to target more complex transcriptomes to identify and track the expression changes of rare RNA isoforms from all genes. Technologies that will advance achievement of this goal are pair-end sequencing, strand-specific sequencing and the use of longer reads to increase coverage and depth. As the cost of sequencing continues to fall, RNA-Seq is expected to replace microarrays for many applications that involve determining the structure and dynamics of the transcriptome.

**References**

1. https://www.genome.gov/13014330
2. Wang Z., Gerstein M., Synder M. (2009). Rna-seq: a revolutionary tool for transciptomics, Nat Rev Genet 10(1): 57–63.
3. http://scienceblogs.com/digitalbio/2011/01/08/next-gene-sequencing-results-a/
4. Shendure J, Ji H (2008) Next-generation RNA sequencing. Nature Biotechnology 26: 2514-2521
5. Anders S, Huber W (2010). Differential expression analysis for sequence count data. Genome Biol. 11:R106.
   Illumina, Inc,. (2011). Getting started with RNA-Seq Data Analysis. Pub. No. 470-2011-003.
6. Illumina, Inc,. (2011). RNA-Seq Data Comparison with Gene Expression Microarrays. A cross-platform comparison of differential gene expression analysis. Pub. No. 470-2011-004
7. Yaqing Si (2012). Statistical analysis of RNA-seq data from next-generation sequencing technology. PhD thesis. Iowa State University, Ames, Iowa.
8. Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L., and Wold, B. (2008). Mapping and quantifying mammalian transcriptomes by rna-seq. Nat Methods, 5(7):621-628.
9. Wang L., Si Y., Dedow L.K., Shao Y., Liu P., Brutnell T.P. (2010). A low-cost library construction protocol and data analysis pipeline for Illumina-based strand-specific multiplex RNA-seq. PLoS One 6(10):e26426.
10. Brian J. H. and Michael C. Z. (2010). Advancing RNA-Seq analysis Nature Biotechnology 28, 421-423.

# Hands-on Session for Transcriptomic Data Analysis

**Soumya Sharma**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

Identification of differentially expressed genes from the RNA-Seq data is an important area of bioinformatics data analysis. There are several packages available in R to carry out the differential gene expression analysis, like **DESeq2 (**Love et al., 2014), **edgeR (**Robinson et al., 2010), **limma (**Smyth et al., 2005) *etc.* After preprocessing and quantification of reads in RNA-Seq data, we get a matrix of read counts of each gene in every sample. Then we can use the "**DESeq2**" package to identify differentially expressed genes. Here, we demonstrate the differential gene expression analysis with R using a sample dataset available in the R package **airway (**Himes et al., 2014) in following steps.

i) Download the sample dataset from the "**airway**" package. The package contains 2 data files. One file contains read counts of 64102 genes in 8 samples obtained from the RNA-Seq experiment on 4 primary human airway smooth muscle cell lines treated with 1 micromolar dexamethasone for 18 hours. Another file contains sample-wise metadata information, *viz.*, treated or untreated. Import the count matrix and metadata file into RStudio.

**R code to collect sample dataset from "airway" package:**

```
# installing Bioconductor packages
if (!requireNamespace("BiocManager", quietly=TRUE))
  install.packages("BiocManager")
   BiocManager::install("airway")
library(airway)
data(airway)
airway
sample_info <- as.data.frame(colData(airway))
sample_info <- sample_info[,c(2,3)]
sample_info$dex <- gsub('trt', 'treated', sample_info$dex)
sample_info$dex <- gsub('untrt', 'untreated', sample_info$dex)
```

```
names(sample_info) <- c('cellLine', 'dexamethasone')

# Get the samplewise metadata file

write.table(sample_info, file = "/sample_info.csv", sep = ',', col.names = T, row.names = T, quote = F)

# Get the matrix of read counts for each gene in every sample

countsData <- assay(airway)

write.table(countsData, file = "/counts_data.csv", sep = ',', col.names = T, row.names = T, quote = F)
```

ii) Then we have to load the package "**DESeq2**" to perform the subsequent differential gene expression analysis. We have to create a DESeqDataSet object and then run the 'DESeq()' function to perform the said analysis.

**Differential gene expression analysis using the "DESeq2" package in R**

```
BiocManager::install("DESeq2")

library(DESeq2)

# read in counts data

counts_data <- read.csv('/counts_data.csv')

# read in sample info

colData <- read.csv('/sample_info.csv')

# making sure the row names in colData matches to column names in counts_data

all(colnames(counts_data) %in% rownames(colData))

# are they in the same order?

all(colnames(counts_data) == rownames(colData))

dds <- DESeqDataSetFromMatrix(countData = counts_data, colData = colData, design = ~ dexamethasone)

dds

#pre-filtering: removing rows with low gene counts

# keeping rows that have at least 10 reads total

keep <- rowSums(counts(dds)) >= 10

dds <- dds[keep,]

# set the factor level
```

```
dds$dexamethasone <- relevel(dds$dexamethasone, ref = "untreated")
# --------Run DESeq ----------------------
dds <- DESeq(dds)
res <- results(dds)
res
summary(res)
res0.01 <- results(dds, alpha = 0.01) # When padj = 0.01
summary(res0.01)
```

Here, we are trying to find the genes which are differentially expressed in Dexamethasone treated conditions as compared to untreated conditions. Hence, the reference level is set as ‘untreated’. After the analysis, the result contains base means, log$_2$FoldChange values, p-values, adjusted p-values, *etc.* for each gene. If at 1% level, the adjusted p-value for a gene is found as > 0.01, it means the result has been obtained purely by chance, *i.e.*, a non-significant result. Otherwise, that gene is differentially expressed if the adjusted p-value is < 0.01. In the latter case, if the log2FoldChange value is > 0, the gene is upregulated and if it is < 0, then that gene is downregulated. Thus, we can find out differentially expressed genes using R.

iii) Visualization of differentially expressed genes in R. After identifying differentially expressed genes, we can visualize the result in terms of various plots such as MA plot, volcano plot, heatmap, *etc*. Several R packages are available to develop these plots. MA plot can be generated using the ‘plotMA()’ function. We can use the “**ggplot2**” package to develop volcano plot. Similarly, R package “**heatmap2”, “pheatmap”** *etc*. are useful to create heatmaps. MA plot (fig 1), volcano plot (fig 2) and heatmap (fig 3) created from the result of the previous analysis.

**R code to visualize the result of differential gene expression analysis**

```
 # MA plot
 plotMA(res)
 # Volcano plot
 library(ggplot2)
 library(tidyverse)
```

```
df<-as.data.frame(res)

df$diffexpressed <- "non-significant"

# if log2Foldchange > 0 and padj < 0.01, set as "UP"

df$diffexpressed[df$log2FoldChange > 0 & df$padj < 0.01] <- "UP"

# if log2Foldchange < 0 and padj < 0.01, set as "DOWN"

df$diffexpressed[df$log2FoldChange < 0 & df$padj < 0.01] <- "DOWN"

ggplot(df, aes(log2FoldChange, -log10(padj), col=
diffexpressed))+geom_point()+scale_color_manual(values = c("red", "black", "green"))


# Developing Heatmap of first 10 genes for better demonstration

library(pheatmap)

library(RColorBrewer)

breaksList = seq(-0.4, 0.5, by = 0.04)

rowLabel = row.names(counts_data[1:10,])

pheatmap(df$log2FoldChange[1:10], color = colorRampPalette(c("dark blue", "white",
"yellow"))(25), breaks = breaksList, border_color = "black", cellheight = 25, cellwidth = 25,
cluster_rows = F,cluster_cols = F, fontsize = 12, labels_row = rowLabel)
```

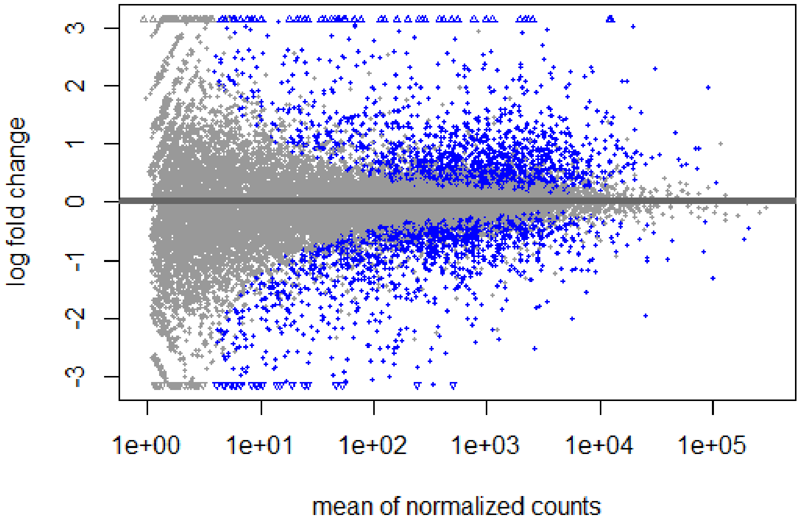


**Fig 1: MA plot showing significantly upregulated and downregulated genes as blue dots.**

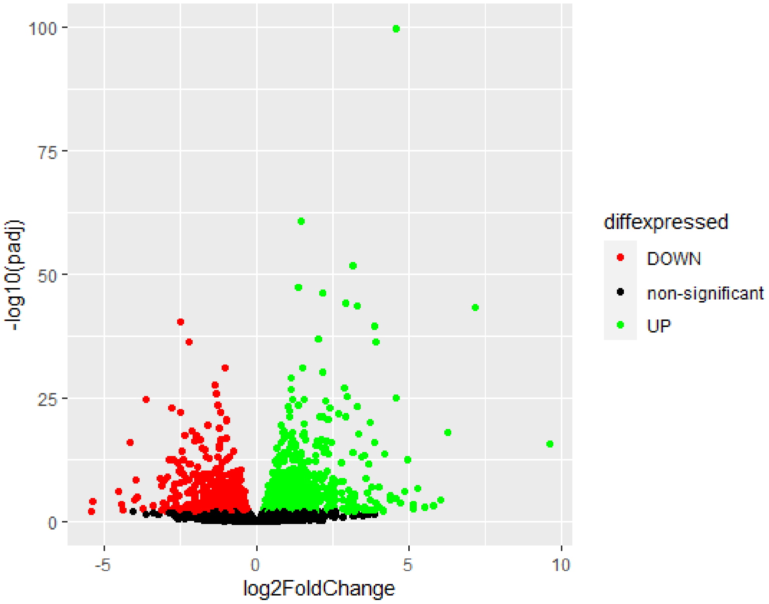


**Fig 2: Volcano plot representing upregulated genes as green, downregulated genes as red and non-significant genes as black dots.**

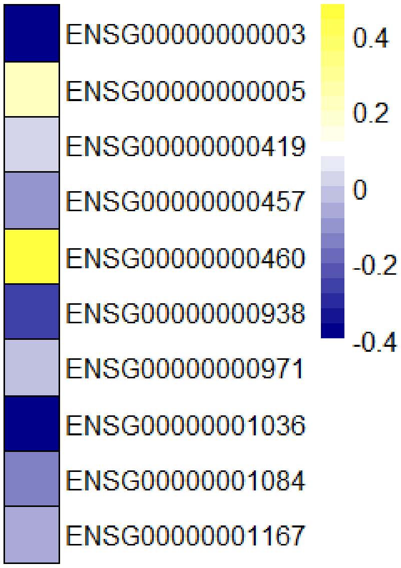


**Fig 3: Heatmap representing the expression levels of first 10 genes in terms of log2FoldChange values in a scale of -0.4 to 0.4 where, blue colour represents downregulated genes, yellow represents upregulated genes and expression levels of remaining genes are represented by gradation of colour between blue and yellow.**

**References:**

Himes, B. E., Jiang, X., Wagner, P., Hu, R., Wang, Q., Klanderman, B., & Lu, Q. (2014). RNA-Seq transcriptome profiling identifies CRISPLD2 as a glucocorticoid responsive gene that modulates cytokine function in airway smooth muscle cells. *PloS one*, *9*(6), e99625.

Love, M. I., Huber, W., & Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome biology*, *15*(12), 1-21.

Robinson, M. D., McCarthy, D. J., & Smyth, G. K. (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *bioinformatics*, *26*(1), 139-140.

Smyth, G. K. (2005). Limma: linear models for microarray data. *Bioinformatics and computational biology solutions using R and Bioconductor*, 397-420.

# Introduction to Python Programming

**U. B. Angadi and Sudhir Srivastava**

**ICAR- Indian Agricultural Statistics Research Institute, New Delhi**

Python is an easiest and simple open source powerful programming language. It has efficient high-level data structures with support of multiple programming paradigms, such as Procedural, Object Oriented and Functional paradigms.  it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form specifically ML, AI and Data science in Python web site, https://www.python.org/, and may be freely distributed.  The Python interpreter is easily extended with new functions and data types implemented in C or C++. This can be used as a scripting language or can be compiled to byte-code for building large application like Perl, R, LINUX shell script. Python has been developed under virtual machine concept and support.

## Installing Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org/ and also available in many source for a wide variety of platforms.

If the binary code for your platform is not available, you need a C compiler to compile the source code manually. Compiling the source code offers more flexibility in terms of choice of features that you require in your installation. Installation from source codes is better than binary.

## Linux Installation

- Open a Web browser and go to https://www.python.org/downloads/. Or use **wget** command with url of desire version of python.
- Follow the link to download zipped source code available for Unix/Linux.
- Download and extract files and change directory to python folder then run following commands
- **$  ./configure script**
- **$ make**
- **$ make install**

Or you can install though yum  i.e. **sudo yum install python3**

## Window installation

- Down load window version installation file
- Double click to installation file *python-XYZ.msi such as* python-3.10.2-amd64.exe

## Setting up PATH

- **Linux** − type export PATH="$PATH:/usr/local/bin/python" and press Enter. Or make entry the same entry in bashrc file
- **Window**- control panel→System Security→System→System Properties→ Environmental variables→path→ add path at last in existing path

### Running Python

**You can start Python from Unix, DOS, or any other system that provides you a command-line interpreter or shell window.**

Enter **python** the command line and press enter

Or

Stored program or packages with py file extension

Enter **python filename.py** and press enter i.e. $python script.py in linux

Or

Make python file into standard scripting language file by adding **#!/usr/bin/python** in top of the python code/scrip file

Add executable previlages to python file **$ chmod +x** pythonfile.py

Run python file by **$ ./pythonefile.py** (dot slash filename)

### GUI - Integrated Development Environment

You can run Python from a Graphical User Interface (GUI) environment as well, if you have a GUI application on your system that supports Python.

- **Unix** − IDLE is the very first Unix IDE for Python.
- **Windows –** PythonWin/pycharm/MSvisual studio are Windows interface for Python and is an IDE with a GUI.

For these IDE need to be set python interpreter

### Python Lines and Indentation

Python programming provides no braces to indicate blocks of code for class and function definitions or flow control. Blocks of code are denoted by **line indentation**, which is rigidly enforced.

The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount

### Comments- non executable statement

Python comments are non-executable and readable explanation or annotations for programmer. They are added with the purpose of making the source code easier for humans to understand and are ignored by Python interpreter.

### Single Line Comments

A hash sign (#) at beginning of a string. All characters after the # and up to the end of the physical line are part of the comment.

```
# This is a single line comment in python and below is print statement
```

```
print ("Hello, World! This print statement print constant and variable")
```

### Multi-Line Comments

Triple-quoted string can be used for multiline comments and it ignores by Python interpreter

```
"""
This is first in multi-lines
This is 2nd in multi-lines
This is 3rd in multi-lines
"""
```

### Docstring Comments

Python docstrings provide a convenient way to provide a help documentation with Python modules, functions, classes, and methods. The **docstring** is then made available via the __doc__ attribute.

```
def add(a, b):
   """Function to add the value of a and b"""
   return a+b
print(add.__doc__)
print(add.__doc__) # for help
print(add(10,20)) # for execution
```

### Variables

Python variables are name of memory location, in which values are stored. This means that when you create a variable you reserve some space in the memory to store values. Based on the data type of a variable, Python interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to Python variables, you can store integers, decimals or characters in these variables.

Python variables do not need explicit declaration  like other language to reserve memory space or to create a variable. A Python variable is created automatically when you assign a value to it. The equal sign (=) is used to assign values to variables.

The operand to the left of the = operator is the name of the variable and the operand to the right of the = operator is the value stored in the variable.

```
counter = 1000          # Creates an integer variable
miles   = 11234.567       # Creates a floating point variable
name    = "Arun Kumar"   # Creates a string variable
print (counter)
print (miles)
print (name)
```

### Delete a Variable

You can delete the reference to a number object by using the del statement.

```
del var1[,var2[,var3[....,varN]]]]
del var
del var_a, var_b
```

### Local Variable

Python Local Variables are defined inside a function. We can not access variable outside the function.

```
def sum(x,y):
  sum = x + y
  return sum
print(sum(5, 10))
```

### Global Variable

Any variable created outside a function can be accessed within any function and so they have global scope.

```
x = 5
y = 10
def sum():
  sum = x + y
  return sum
print(sum())
```

### Data Types

Python has various built-in data types which we will discuss with in this tutorial:

- **Numeric - int, float, complex**

```
# integer variable.
a=123
print("The type of variable having value", a, " is ", type(a))
# float variable.
b=2345.345
print("The type of variable having value", b, " is ", type(b))
# complex variable.
c=11+5j
print("The type of variable having value", c, " is ", type(c))
```

- **String – str**

```
str = 'Hello World!'
print (str)          # Prints complete string

print (str[0])       # Prints first character of the string

print (str[2:5])     # Prints characters starting from 3rd to 5th
print (str[2:])      # Prints string starting from 3rd character
print (str * 2)      # Prints string two times
print (str + "TEST") # Prints concatenated string
```

- **Sequence - list, tuple, range**

A Python list contains items separated by commas and enclosed within square brackets ([]). To some extent, Python lists are similar to arrays in C. One difference between them is that all the items belonging to a Python list can be of different data type

```
list = [ 'abcd', 786, 2.23, 'john', 70.2 ]
tinylist = [123, 'john']
print (list)           # Prints complete list
print (list[0])        # Prints first element of the list
print (list[1:3])      # Prints elements starting from 2nd till 3rd
print (list[2:])       # Prints elements starting from 3rd element
print (tinylist * 2)    # Prints list two times
print (list + tinylist) # Prints concatenated lists
```

Tuple is another sequence data type that is similar to a list. A Python tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

Lists are enclosed in brackets ( [ ] ) and their elements and size can be changed, while tuples are enclosed in parentheses ( ( ) ) and cannot be updated. Tuples can be thought of as **read-only** lists

```
tuple = ( 'abcd', 786 , 2.23, 'john', 70.2  )
tinytuple = (123, 'john')
print (tuple)              # Prints the complete tuple
print (tuple[0])           # Prints first element of the tuple
print (tuple[1:3])         # Prints elements of the tuple starting from 2nd till 3rd
print (tuple[2:])          # Prints elements of the tuple starting from 3rd element
print (tinytuple * 2)       # Prints the contents of the tuple twice
print (tuple + tinytuple)   # Prints concatenated tuples
```

Range - **range()** is an in-built function in Python which returns a sequence of numbers starting from 0 and increments to 1 until it reaches a specified number.

We use **range()** function with for and while loop to generate a sequence of numbers.

```
range(start, stop, step)
```

- **Mapping - dict**

Python dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object.

Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([])

```
dict = {}
dict['one'] = "This is one"
dict[2]     = "This is two"
tinydict = {'name': 'john','code':6734, 'dept': 'sales'}
print (dict['one'])      # Prints value for 'one' key
print (dict[2])          # Prints value for 2 key
print (tinydict)         # Prints complete dictionary
print (tinydict.keys())  # Prints all the keys
print (tinydict.values()) # Prints all the values
```

- **Binary - bytes, bytearray, memoryview**

```
hexStr = bytes.fromhex('A2f7 4509')
myByteArray = bytearray('String', 'UTF-8')
memView = memoryview(myByteArray)
```

- **Boolean – bool**

**Boolean** type is one of built-in data types which represents one of the two values either **True** or **False**. Python **bool()** function allows you to evaluate the value of any expression and returns either True or False based on the expression.

```
a = True
# display the value of a
print(a)
# display the data type of a
print(type(a))
```

- **Set - set, frozenset- immutable**

```
fruits = {"Apple", "Banana", "Cherry", "Apple", "Kiwi"}
fruits.add("Orange")
fruits.remove("Mango")
print('After removing element:', fruits)
l = ["Geeks", "for", "Geeks"]
fnum = frozenset(l)
```

### Data Type Conversion

Sometimes, you may need to perform conversions between the built-in data types. To convert data between different data types.

| Function & Description |
|---|
| **int(x [,base]) -**Converts x to an integer. base specifies the base if x is a string. |
| **long(x [,base] ) -**Converts x to a long integer. base specifies the base if x is a string. |

| |
|---|
| **float(x) -**Converts x to a floating-point number. |
| **complex(real [,imag]) -**Creates a complex number. |
| **str(x) -**Converts object x to a string representation. |
| **repr(x) -**Converts object x to an expression string. |
| **eval(str)-**Evaluates a string and returns an object. |
| **tuple(s)-**Converts s to a tuple. |
| **list(s)-**Converts s to a list. |
| **set(s)-**Converts s to a set. |
| **dict(d)-**Creates a dictionary. d must be a sequence of (key,value) tuples. |
| **frozenset(s)-**Converts s to a frozen set. |
| **chr(x)-**Converts an integer to a character. |
| **unichr(x)-**Converts an integer to a Unicode character. |
| **ord(x)-**Converts a single character to its integer value. |
| **hex(x)-**Converts an integer to a hexadecimal string. |
| **oct(x)-**Converts an integer to an octal string. |

### Arithmetic Operators

Arithmetic operators are used to perform mathematical operations on numerical values. List is given below table

| Operator | Name | Example |
|---|---|---|
| + | Addition | 10 + 20 = 30 |
| - | Subtraction | 20 – 10 = 10 |
| * | Multiplication | 10 * 20 = 200 |
| / | Division | 20 / 10 = 2 |
| % | Modulus | 22 % 10 = 2 |

|  |  |  |
|---|---|---|
| ** | Exponent | 4**2 = 16 |
| // | Floor Division | 9//2 = 4 |

**Comparison/relational Operators**

Python comparison operators compare the values on either sides of them and decide the relation among them.

| Operator | Name | Example |
|---|---|---|
| == | Equal | 4 == 5 is not true. |
| != | Not Equal | 4 != 5 is true. |
| > | Greater Than | 4 > 5 is not true. |
| < | Less Than | 4 < 5 is true. |
| >= | Greater than or Equal to | 4 >= 5 is not true. |
| <= | Less than or Equal to | 4 <= 5 is true. |

**Assignment Operators**

Python assignment operators are used to assign values to variables. These operators include simple and complex with arithmetic operator.

| Operator | Name | Example |
|---|---|---|
| = | Assignment Operator | a = 10 |
| += | Addition Assignment | a += 5 (Same as a = a + 5) |
| -= | Subtraction Assignment | a -= 5 (Same as a = a - 5) |
| *= | Multiplication Assignment | a *= 5 (Same as a = a * 5) |
| /= | Division Assignment | a /= 5 (Same as a = a / 5) |
| %= | Remainder Assignment | a %= 5 (Same as a = a % 5) |
| **= | Exponent Assignment | a **= 2 (Same as a = a ** 2) |
| //= | Floor Division Assignment | a //= 3 (Same as a = a // 3) |

**Bitwise Operators**

Bitwise operator works on bits and performs bit by bit operation. Assume if a = 60; and b = 13; Now in the binary format their values will be 0011 1100 and 0000 1101 respectively.

| Operator | Name | Example |
|---|---|---|
| & | Binary AND | Sets each bit to 1 if both bits are 1<br>a&b = 12 (0000 1100 |
| \| | Binary OR | Sets each bit to 1 if one of two bits is 1<br>a\|b = 61 (0011 1101) |
| ^ | Binary XOR | Sets each bit to 1 if only one of two bits is 1<br>a^b = 49 (0011 0001) |
| ~ | Binary Ones Complement | Inverts all the bits<br>~a  = -61 (1100 0011) |
| << | Binary Left Shift | Shift left by pushing zeros in from the right and let the leftmost bits fall off<br>a << 2 = 240 (1111 0000) |
| >> | Binary Right Shift | Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off<br>a>>2 = 15 (0000 1111) |

### Logical Operators

There are following logical operators supported by Python language. Assume variable a holds 10 and variable b holds 20 then

| Operator | Description | Example |
|---|---|---|
| and Logical AND | If both the operands are true then condition becomes true. | (a and b) is true. |
| or Logical OR | If any of the two operands are non-zero then condition becomes true. | (a or b) is true. |
| not Logical NOT | Used to reverse the logical state of its operand. | Not(a and b) is false. |

### Membership Operators

Membership operators test for membership in a sequence, such as strings, lists, or tuples. There are two membership operators as explained below −

| Operator | Description | Example |
|---|---|---|

| | | |
|---|---|---|
| in | Evaluates to true if it finds a variable in the specified sequence and false otherwise. | x in y, here in results in a 1 if x is a member of sequence y. |
| not in | Evaluates to true if it does not finds a variable in the specified sequence and false otherwise. | x not in y, here not in results in a 1 if x is not a member of sequence y. |

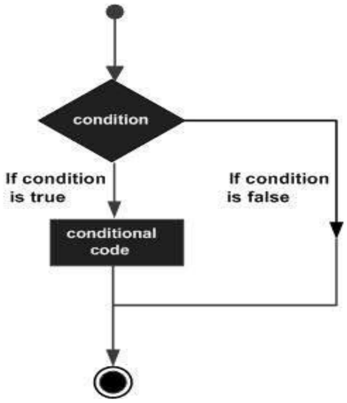
**Identity Operators**

Identity operators compare the memory locations of two objects.

| Operator | Description | Example |
|---|---|---|
| is | Evaluates to true if the variables on either side of the operator point to the same object and false otherwise. | x is y, here **is** results in 1 if id(x) equals id(y). |
| is not | Evaluates to false if the variables on either side of the operator point to the same object and true otherwise. | x is not y, here **is not** results in 1 if id(x) is not equal to id(y). |

**Decision making**

Usual codes are executed sequentially, the first statement in a function is executed first, followed by the second, and so on. Decision making is to change path on conditions while execution of the program and specifying action/path taken according to the conditions result(TRUE/FALSE).



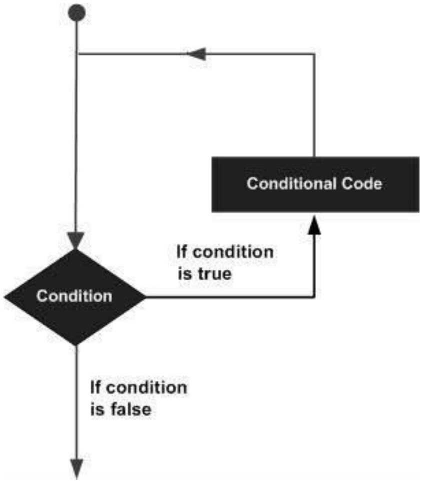
| Sr.No. | Statement & Description |
|---|---|
| 1 | if statements<br>An **if statement** consists of a boolean expression followed by one or more statements. |
| 2 | if...else statements |

| | |
|---|---|
| | An **if statement** can be followed by an optional **else statement**, which executes when the boolean expression is FALSE. |
| 3 | nested if statements<br>Again **if** or **else can use in if** statement inside another **if** or **else if** statement(s). |

```
var = 100
if ( var == 100 ) : print "Value of expression is 100"
print "Good bye!"
amount = 2000
if ( amount <10000 ) : print "Interest rate is 10%"
else:
print "Interest rate is 20 %"
```

### Loops

Generally statements are executed sequentially. There may be a situation when you need to execute a block of code several number of times or based termination condition. A loop statement allows us to execute a statement or group of statements multiple times.



Python programming language provides following types of loops to handle looping requirements.

| Sr.No. | Loop Type & Description |
|---|---|
| 1 | while loop<br>Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body. |
| 2 | for loop<br>Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable. |

| | |
|---|---|
| 3 | nested loops<br>You can use one or more loop inside any another while, for or do..while loop. |

### Loop Control Statements

Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed.

Python supports the following control statements. Click the following links to check their detail.

Let us go through the loop control statements briefly

| Sr.No. | Control Statement & Description |
|---|---|
| 1 | **break statement :**Terminates the loop statement and transfers execution to the statement immediately following the loop. |
| 2 | **continue statement :**Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating. |

```
i = 1
while i < 6:
  print(i)
  if i == 3:
    break
  i += 1
```

### Functions

A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity and a high degree of code reusing. You can define functions to provide the required functionality with following simple rules.

- Function blocks begin with the keyword **def** followed by the function name and parentheses ( ) and then a colon (:)
- Input parameters should be placed within these parentheses. parameters can be defined inside the parentheses.
- First statement of a function can be an optional statement - the documentation string of the function or *docstring*.
- The statement return [expression] exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as return None.

```
def printme( str ):
   "This prints a passed string into this function"
   print str
   return
```

### Calling a Function

Defining a function only gives it a name, specifies the parameters that are to be included in the function and structures the blocks of code.

```
printme("I'm first call to user defined function!")
printme("Again second call to the same function")
```

### Required arguments

Required arguments are the arguments passed to a function in correct positional order. Here, the number of arguments in the function call should match exactly with the function definition.

```
#!/usr/bin/python

# Function definition is here
def printme( str1, str2 ):
   "This prints a passed string into this function"
   print str1
   print str2
   return “Success”;
# Now you can call printme function
printme(“Hi”, “Good Morning”)
```

### Keyword arguments

Keyword arguments are related to the function calls. When you use keyword arguments in a function call, the caller identifies the arguments by the parameter name. This allows you to skip arguments (if default is assigned) or place them out of order because the Python interpreter is able to use the keywords provided to match the values with parameters

```
#!/usr/bin/python
# Function definition is here
def printme( str1, str2 ):
   "This prints a passed string into this function"
   print str
   return;
# Now you can call printme function
printme( str2 = "Good Morning", str1=”Hi!!”)
```

### Default arguments

A default argument is an argument that assumes a default value if a value is not provided in the function call for that argument. The following example gives an idea on default arguments, it prints default age if it is not passed −

```
#!/usr/bin/python
# Function definition is here
def printinfo( name, age = 35 ):
   "This prints a passed info into this function"
```

```
  print "Name: ", name
  print "Age ", age
  return;
# Now you can call printinfo function
printinfo( age=50, name="miki" )
printinfo( name="miki" )
```

### The *Anonymous* Functions

These functions are called anonymous because they are not declared in the standard manner by using the *def* keyword. You can use the *lambda* keyword to create small anonymous functions.

- Lambda form is one-line statement and can take any number of arguments but return just one.
- An anonymous function cannot be a direct call to print because lambda requires an expression
- Can be own local namespace and cannot access variables other than those in their parameter list.

**lambda [arg1 [,arg2,.....argn]]:expression**

### Modules

A module is a Python object with arbitrarily named attributes and logically organize python code/functions. Grouping related code into a module makes the code easier to understand and use. A module is a file consisting of Python code. A module can define functions, classes and variables.

The Python code for a module named *aname* normally resides in a file named *aname.py*. Here's an example of a simple module, support.py

```
def print_func( par ):
  print "Hello : ", par
  return
```

### The *import* Statement

You can use any Python source file as a module by executing an import statement in some other Python source file as below.

**import module1[, module2[,... moduleN]**

It imports the module if the module is present in the search path. A search path is a list of directories that the interpreter searches before importing a module. Example, to import the module support.py, need to put the following command at top of the script

```
#!/usr/bin/python
# Import module support
import support
# Now you can call defined function that module as follows
support.print_func("Zara")
```

A module is loaded only once, regardless of the number of times it is imported.

### The *from...import* Statement

Python's *from* statement lets you import specific attributes from a module into the current namespace. The *from...import* has the following syntax −

**from modname import name1[, name2[, ... nameN]]**

Import the function fibonacci from the module fib, use the following statement

**from fib import fibonacci**

This statement does not import the entire module fib into the current namespace; it just introduces the item fibonacci from the module fib into the global symbol table of the importing module.

### The from...import * Statement

It is also possible to import all names from a module into the current namespace.

from modname import *

### Locating Modules

When you import a module, the Python interpreter searches for the module in the following sequences.

- The current directory.
- If the module isn't found, Python then searches each directory in the shell variable PYTHONPATH.
- If all else fails, Python checks the default path. On UNIX, this default path is normally /usr/local/lib/python/.

The module search path is stored in the system module sys as the **sys.path** variable. The sys.path variable contains the current directory, PYTHONPATH, and the installation-dependent default.

### The *PYTHONPATH* Variable

The PYTHONPATH is an environment variable, consisting of a list of directories. The syntax of PYTHONPATH is the same as that of the shell variable PATH.

Here is a typical PYTHONPATH from a Windows system −

set PYTHONPATH = c:\python20\lib;

And here is a typical PYTHONPATH from a UNIX system −

set PYTHONPATH = /usr/local/lib/python

```
#!/usr/bin/python
Money = 2000
def AddMoney():
  # Uncomment the following line to fix the code:
  # global Money
  Money = Money + 1
print Money
AddMoney()
print Money
```

### The *reload()* Function

When the module is imported into a script, the code in the top-level portion of a module is executed only once. if you want to reexecute the top-level code in a module while module development stage or modified, you can use the *reload()* function. The reload() function imports a previously imported module again.

**reload(module_name)**

## Files I/O

### Printing to the Screen

The simplest way to produce output is using the *print* statement where you can pass zero or more expressions separated by commas. This function converts the expressions you pass into a string and writes the result to standard output (screen)

```
#!/usr/bin/python
print "Python is really a great language,", "isn't it?"
```

### Reading Keyboard Input

Python provides two built-in functions to read a line of text from standard input, which by default comes from the keyboard.

- raw_input
- input

### The *raw_input* Function

The *raw_input([prompt])* function reads one line from standard input and returns it as a string (removing the trailing newline).

```
#!/usr/bin/python
str = raw_input("Enter your input: ")
print "Received input is : ", str
```

```
Enter your input: Hello Python
Received input is :  Hello Python
```

### The *input* Function

The *input([prompt])* function is equivalent to raw_input, except that it assumes the input is a valid Python expression and returns the evaluated result.

```
#!/usr/bin/python
str = input("Enter your input: ")
print "Received input is : ", str
```

This would produce the following result against the entered input −

```
Enter your input: [x*5 for x in range(2,10,2)]

Recieved input is :  [10, 20, 30, 40]
```

### Opening and Closing Files

Until now, you have been reading and writing to the standard input and output. Now, we will see how to use actual data files.

### The *open* Function

Before you can read or write a file, you have to open it using Python's built-in *open()* function. This function creates a **file** object, which would be utilized to call other support methods associated with it.

**file object = open(file_name [, access_mode][, buffering])**

Here are parameter details −

- **file_name** − The file_name argument is a string that contains the name of the file that you want to access.
- **access_mode** − The access_mode determines the mode in which the file has to be opened, i.e., read, write, append, etc and details as below.
- **buffering** − If the buffering value is set to 0, no buffering takes place. If the buffering value is 1, line buffering is performed while accessing a file. If you specify the buffering value as an integer greater than 1, then buffering action is performed with the indicated buffer size. If negative, the buffer size is the system default(default behavior).

Here is a list of the different modes of opening a file

| Sr.No. | Modes & Description |
|---|---|
| 1 | **r**<br>Opens a file for reading only. The file pointer is placed at the beginning of the file. This is the default mode. |
| 2 | **rb**<br>Opens a file for reading only in binary format. The file pointer is placed at the beginning of the file. This is the default mode. |
| 3 | **r+**<br>Opens a file for both reading and writing. The file pointer placed at the beginning of the file. |
| 4 | **rb+**<br>Opens a file for both reading and writing in binary format. The file pointer placed at the beginning of the file. |
| 5 | **w**<br>Opens a file for writing only. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing. |
| 6 | **wb**<br>Opens a file for writing only in binary format. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing. |

| | |
|---|---|
| 7 | **w+**<br>Opens a file for both writing and reading. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing. |
| 8 | **wb+**<br>Opens a file for both writing and reading in binary format. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing. |
| 9 | **a**<br>Opens a file for appending. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing. |
| 10 | **ab**<br>Opens a file for appending in binary format. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing. |
| 11 | **a+**<br>Opens a file for both appending and reading. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing. |
| 12 | **ab+**<br>Opens a file for both appending and reading in binary format. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing. |

### The *file* Object Attributes

Once a file is opened and you have one *file* object, you can get various information related to that file.

Here is a list of all attributes related to file object −

| Sr.No. | Attribute & Description |
|---|---|
| 1 | **file.closed**<br>Returns true if file is closed, false otherwise. |
| 2 | **file.mode**<br>Returns access mode with which file was opened. |
| 3 | **file.name**<br>Returns name of the file. |

| | |
|---|---|
| 4 | **file.softspace**<br>Returns false if space explicitly required with print, true otherwise. |

```
#!/usr/bin/python
# Open a file
fo = open("foo.txt", "wb")
print "Name of the file: ", fo.name
print "Closed or not : ", fo.closed
print "Opening mode : ", fo.mode
print "Softspace flag : ", fo.softspace
```

This produces the following result −

```
Name of the file:  foo.txt
Closed or not :  False
Opening mode :  wb
Softspace flag :  0
```

### The *close()* Method

The close() method of a *file* object closes the file object, after which no more access for read or write. Python automatically closes a file when the reference object of a file is reassigned to another file. It is a good practice to use the close() method to close a file.

**fileObject.close()**

```
#!/usr/bin/python
# Open a file
fo = open("foo.txt", "wb")
print "Name of the file: ", fo.name
# Close opend file
fo.close()
```

```
Name of the file:  foo.txt
```

### The *write()* Method

The *write()* method writes any string to the opened file. The write() method does not add a newline character ('\n') to the end of the string

**fileObject.write(string)**

Here, passed parameter is the content to be written into the opened file.

```
#!/usr/bin/python
# Open a file
fo = open("foo.txt", "wb")
fo.write( "Python is a great language.\nYeah its great!!\n")
# Close opend file
fo.close()
```

### The *read()* Method

The *read()* method reads a string from an open file. It is important to note that Python strings can have binary data. apart from text data.

**fileObject.read([count])**

Here, passed parameter is the number of bytes to be read from the opened file. This method starts reading from the beginning of the file and if *count* is missing, then it tries to read as much as possible, maybe until the end of file.

```
#!/usr/bin/python
# Open a file
fo = open("foo.txt", "r+")
str = fo.read(10);
print "Read String is : ", str
# Close opend file
fo.close()
```

### File Positions

The *tell()* method tells you the current position within the file; in other words, the next read or write will occur at that many bytes from the beginning of the file.

The *seek(offset[, from])* method changes the current file position. The *offset* argument indicates the number of bytes to be moved. The *from* argument specifies the reference position from where the bytes are to be moved.

If *from* is set to 0, it means use the beginning of the file as the reference position and 1 means use the current position as the reference position and if it is set to 2 then the end of the file would be taken as the reference position.

```
#!/usr/bin/python
# Open a file
fo = open("foo.txt", "r+")
str = fo.read(10)
print "Read String is : ", str
# Check current position
position = fo.tell()
print "Current file position : ", position
# Reposition pointer at the beginning once again
position = fo.seek(0, 0);
str = fo.read(10)
print "Again read String is : ", str
# Close opend file
fo.close()
```

```
Read String is :  Python is

Current file position :  10

Again read String is :  Python is
```

## Renaming and Deleting Files

Python **os** module provides methods that help you perform file-processing operations, such as renaming and deleting files.

To use this module you need to import **os** module first and then you can call any related functions.

### The rename() Method

The *rename()* method takes two arguments, the current filename and the new filename.

**os.rename(current_file_name, new_file_name)**

```
#!/usr/bin/python
import os
# Rename a file from test1.txt to test2.txt
os.rename( "test1.txt", "test2.txt" )
```

You can use the *remove()* method to delete files by supplying the name of the file to be deleted as the argument.

**os.remove(file_name)**

```
#!/usr/bin/python
import os
# Delete file test2.txt
os.remove("text2.txt")
```

### Directories in Python

All files are contained within various directories, and Python has handling these too. The **os** module has several methods that help you create, remove, and change directories.

### The *mkdir()* Method

You can use the *mkdir()* method of the **os** module to create directories in the current directory. You need to supply an argument to this method which contains the name of the directory to be created.

**os.mkdir("newdir")**

```
#!/usr/bin/python
import os
# Create a directory "test"
os.mkdir("test")
```

### The *chdir()* Method

You can use the *chdir()* method to change the current directory. The chdir() method takes an argument, which is the name of the directory that you want to make the current directory.

**os.chdir("newdir")**

```
#!/usr/bin/python
import os
# Changing a directory to "/home/newdir"
```

```
os.chdir("/home/newdir")
```

### The *getcwd()* Method

The *getcwd()* method displays the current working directory.

os.getcwd()

```
#!/usr/bin/python
import os
# This would give location of the current directory
os.getcwd()
```

### The *rmdir()* Method

The *rmdir()* method deletes the directory, which is passed as an argument in the method.

**os.rmdir('dirname')**

```
#!/usr/bin/python
import os
# This would  remove "/tmp/test"  directory.
os.rmdir( "/tmp/test"  )
```

### Virtual environments

A virtual environment is a provision to keep dependencies required by different projects. For a scenario, working on two python projects one of them uses Tensorflow 4.0 and another uses Tensorflow 4.1. In this scenario tow environment may be created. When used from within a virtual environment, common installation tools such as pip will install Python packages into a virtual environment

Creating virtual environments

```
python3 -m venv /path/to/new/virtual/environment
usage: venv [-h] [--system-site-packages] [--symlinks | --copies] [--clear]
            [--upgrade] [--without-pip] [--prompt PROMPT] [--upgrade-deps]
            ENV_DIR [ENV_DIR ...]
Creates virtual Python environments in one or more target directories.
positional arguments:
  ENV_DIR               A directory to create the environment in.
optional arguments:
  -h, --help            show this help message and exit
  --system-site-packages
                        Give the virtual environment access to the system
                        site-packages dir.
  --symlinks            Try to use symlinks rather than copies, when symlinks
                        are not the default for the platform.
  --copies              Try to use copies rather than symlinks, even when
                        symlinks are the default for the platform.
  --clear               Delete the contents of the environment directory if it
```

```
                        already exists, before environment creation.
  --upgrade             Upgrade the environment directory to use this version
                        of Python, assuming Python has been upgraded in-place.
  --without-pip         Skips installing or upgrading pip in the virtual
                        environment (pip is bootstrapped by default)
  --prompt PROMPT       Provides an alternative prompt prefix for this
                        environment.
  --upgrade-deps        Upgrade core dependencies: pip setuptools to the
                        latest version in PyPI
Once an environment has been created, you may wish to activate it, e.g. by
sourcing an activate script in its bin directory.
source env/bin/activate
python3 -m pip install requests
deactivate
```

**Using requirements files**

Instead of installing packages individually, pip allows you to declare all dependencies in a Requirements File. Example you could create a plain text file "requirements.txt" with following

```
requests==2.18.4
google-auth==1.1.0
python3 -m pip install -r requirements.txt
```

**Some of python based Bioinformatics tools are given below:**

| **Tool** | **Description** |
|---|---|
| vcfR | Variant call format (VCF) files document the genetic variation observed after DNA sequencing, alignment and variant calling of a sample cohort. Given the complexity of the VCF format as well as the diverse variant annotations and genotype metadata, there is a need for fast, flexible methods enabling intuitive analysis of the variant data within VCF and BCF files. |
| circexplorer2 | it is a comprehensive and integrative circular RNA analysis toolset. |
| VCF-KIt | VCF-kit is a command-line based collection of utilities for performing analysis on Variant Call Format (VCF) files. |
| DMRfinder | it is written in Python and R, DMRfinder efficiently identifies genomic regions with differentially methylated CpG sites from high-throughput MethylC-seq datasets |
| Trim Galore | is a wrapper script to automate quality and adapter trimming as well as quality control, with some added functionality to remove biased methylation positions for RRBS sequence files |

| | |
|---|---|
| mltest | A fast, robust and easy-to-use calculation of multiclass classification evaluation metrics based on confusion matrix. |
| SqueezeMeta | SqueezeMeta is a full automatic pipeline for metagenomics/metatranscriptomics, covering all steps of the analysis. |
| checkM | CheckM provides a set of tools for assessing the quality of genomes recovered from isolates, single cells, or metagenomes. |
| Primer3 | Primer3-py is a Python-abstracted API for the popular Primer3 library. The intention is to provide a simple and reliable interface for automated oligo analysis and design. |
| VCF-kit | VCF-kit is a command-line based collection of utilities for performing analysis on Variant Call Format (VCF) files. |
| gmx-MMPBSA | gmx_MMPBSA is a new tool based on AMBER's MMPBSA.py aiming to perform end-state free energy calculations with GROMACS files |
| MODELLER | MODELLER is used for homology or comparative modeling of protein three-dimensional structures |

## References

- Peter J. A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, Michiel J. L. de Hoon. Biopython: freely available Python tools for computational molecular biology and bioinformatics, *Bioinformatics*, Volume 25, Issue 11, June 2009, Pages 1422–1423, https://doi.org/10.1093/bioinformatics/btp163
- Brad Chapman and Jeff Chang. Biopython: Python tools for computational biology. *ACM SIGBIO Newsletter* 20 (2): 15–19 (August 2000).
- https://docs.python.org/3/tutorial/
- https://en.wikipedia.org/wiki/Python_(programming_language)
- http://biopython.org/DIST/docs/tutorial/Tutorial.html

# Role of Machine Learning Techniques in Bioinformatics

**Sanjeev Kumar**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

The main goal of learning theory is to provide a framework for studying the problem of inference that is of gaining knowledge, making predictions, making decisions or constructing models from a set of data. A theory of inference gives a formal definition of words like learning, generalization, over fitting, and also to characterize the performance of learning algorithms so that, ultimately, it may help design better learning algorithms. There are thus two goals: make things more precise and derive new or improved algorithms.

### *Learning*

What is under study here is the process of inductive inference which can roughly be summarized as the following steps:

- Observe a phenomenon
- Construct a model of that phenomenon
- Make predictions using this model

Though, this definition is very general and could be taken more or less as the goal of Natural Sciences. The goal of Machine Learning is to actually automate this process and the goal of Learning Theory is to formalize it. Given some training data, it is always possible to build a function that fits exactly the data. But, in the presence of noise, this may not be the best thing to do as it would lead to a poor performance on unseen instances; this is usually referred to as over fitting. The general idea behind the design of learning algorithms is thus to look for regularities in the observed phenomenon i.e. training data. These can then be generalized from the observed past to the future. Typically, one would look, in a collection of possible models, for one which fits well the data. This immediately raises the question of how to measure and quantify simplicity of a model.

It turns out that there are many ways to do so, but no best one. In classical statistics, the number of free parameters of a model is usually a measure of its complexity. Surprisingly as it may seem, there is no universal way of measuring simplicity or its counterpart complexity and the choice of a specific measure inherently depends on the problem at hand. It is actually in this choice that the designer of the learning algorithm introduces knowledge about the specific phenomenon under study.

This lack of universally best choice can actually be formalized in what is called the No Free Lunch theorem, which in essence says that, if there is no assumption on how the past i.e. training data is related to the future i.e. test data, prediction is impossible. Even more, if there is no a priori restriction on the possible phenomena that are expected, it is impossible to generalize and there is thus no better algorithm. Hence there is a need to make assumptions.

### *Assumptions*

At the core of the theory is a probabilistic model of the phenomenon or data generation process. Within this model, the relationship between past and future observations is that they both are sampled independently from the identical distribution (i.i.d.). The independence assumption means that each new observation yields maximum information. The identical distribution means that the observations give information about the underlying phenomenon i.e. a probability distribution. An immediate consequence of this very general setting is that one can construct algorithms that are consistent, which means that, as one gets more and more data, the predictions of the algorithm are closer and closer to the optimal ones. So this seems to indicate that we can have some sort of universal algorithm. Unfortunately, any (consistent) algorithm can have an arbitrarily bad behavior when given a finite training set. Again, these assumptions indicate that generalization can only come when one adds specific knowledge to the data. Each learning algorithm encodes specific, and works best when this assumption is satisfied by the problem to which it is applied.

## Formulation of the Learning Problem

Let us consider a model of the learning and analysis of this model can be conducted in the general statistical framework of minimizing expected loss using observed data. The practical problems such as pattern recognition, regression estimation, and density estimation are particular case of this general model.

### *Function Estimation Model*

The model of learning from examples can be described using three components:

- A generator of random vectors *x*, drawn independently from a fixed but unknown distribution *P(x)* ;
- A supervisor that returns an output vector *y* for every input vector *x*, according to a conditional distribution function *P(y/x)* , also fixed but unknown;
- A learning machine capable of implementing a set of functions $f(x,\alpha),\alpha \in \Lambda$.

The problem of learning is that of choosing from the given set of functions $f(x,\alpha),\alpha \in \Lambda$, the one which predicts the supervisor's response in the best possible way. The selection is based on a training set of *l* random independent identically distributed (i.i.d.) observations drawn according to *P(x,y)=P(x)P(y/x).*

$$(x_1,y_1,\ \dots,\ x_l,y_l\ ) \tag{1}$$

### *Problem of Risk Minimization*

In order to choose the best available approximation to the supervisor's response, one measures the *loss* or discrepancy $L(y, f(x,\alpha))$ between the response *y* of the supervisor to a given input *x* and the response $f(x,\alpha)$ provided by the learning machine. Consider the expected value of the loss, given by the *risk functional*

$$R(\alpha)=\int l(y, f(x,\alpha))dP(x,y) \tag{2}$$

The goal is to find the function $f(x,\alpha_0)$ which minimizes the risk functional $R(\alpha)$ (over the class of functions $f(x,\alpha),\alpha \in \Lambda$ in the situation where the joint probability distribution *P(x,y)* is unknown and the only available information is contained in the training set (1).

### *Three Main Learning Problems*

This formulation of the learning problem is rather general. It encompasses many specific problems; the important ones are the problems of pattern recognition, regression estimation, and density estimation.

#### *a) The Problem of Pattern Recognition:*

Let the supervisor's output *y* take on only two values *y={0,1}* and let $f(x,\alpha),\alpha \in \Lambda$, be a set of *indicator* functions (functions which take on only two values zero and one). Consider the following loss-function:

$$L(y, f(x,\alpha)) = \begin{cases} 0 \;\; if \;\; y = f(x,\alpha) \\ 1 \;\; if \;\; y \neq f(x,\alpha) \end{cases} \tag{3}$$

For this loss function, the functional (2) provides the probability of classification error. The problem, therefore, is to find the function which minimizes the probability of classification errors when probability measure *P(x,y)* is unknown, but the data (1) are given.

#### *b) The Problem of Regression Estimation:*

Let the supervisor's answer *y* be a real value, and let $f(x,\alpha),\alpha \in \Lambda$, be a set of real functions which contains the *regression function*

$$f(x,\alpha)=\int ydP(y/x)$$

It is known that if $f(x,\alpha) \in L_2$ then the regression function is the one which minimizes the functional (2) with the the following loss-function:

$$L(y, f(x,\alpha))=(y-f(x,a))^2 \tag{4}$$

Thus the problem of regression estimation is the problem of minimizing the risk functional (2) with the loss function (4) in the situation where the probability measure *P(x,y)* is unknown but the data (1) are given.

#### *c) The Problem of Density Estimation*

Finally, the problem of density estimation from the set of densities $p(x,a), a \in \Lambda$. For this problem we consider the following loss-function:

$$L(p(x,a))=-\log p(x,a) \tag{5}$$

It is known that desired density minimizes the risk functional (2) with the loss-function (5). Thus, again, to estimate the density from the data one has to minimize the risk-functional under the condition where the corresponding probability measure *P(x)* is unknown but i.i.d. data $x_1, \dots, x_n$ are given.

### *The General Setting of the Learning Problem*

The general setting of the learning problem can be described as follows. Let the probability measure *P(z)* be defined on the space *Z*. Consider the set of functions $Q(z,a), a \in \Lambda$ . The goal is to minimize the risk functional

$$R(a) = \int Q(z,a)dP(z), \qquad a \in \Lambda \tag{6}$$

if probability measure *P(z)* is unknown but an i.i.d. sample

$$z_1, \dots , z_l \tag{7}$$

is given. The learning problems considered above are particular cases of this general problem of *minimizing the risk functional (6) on the basis of empirical data* (7), where *z* describes a pair *(x,y)* and *Q(z,a)* is the specific loss function. [for example, one of (3), (4), or (5)].

### *Empirical Risk Minimization Induction Principle*

In order to minimize the risk functional (6), for an unknown probability *P(z)* measure the following induction principle is usually used. The expected risk functional *R(a)* is replaced by the *empirical risk* functional

$$R_{emp}(a) = \frac{1}{l}\sum_{i=1}^{l} Q(z,a) \tag{8}$$

constructed on the basis of the training set (7). The principle is to approximate the function $Q(z,a_0)$ which minimizes risk (6) by the function $Q(z,a_l)$ which minimizes empirical risk (8). This principle is called the empirical risk minimization induction principle (ERM principle).

### *Empirical Risk Minimization Principle and the Classical Methods*

The ERM principle is quite general. The classical methods for solving a specific learning problem, such as the least squares method in the problem of regression estimation or the maximum likelihood method in the problem of density estimation are realizations of the ERM principle for the specific loss functions considered above. In order to specify the regression problem one introduces an n+1 dimensional variable $z = (x,y) = (x^1, \dots, x^n, y)$and uses loss function (4). Using this loss function in the functional (8) yields the functional

$$R_{emp}(a) = \frac{1}{l}\sum_{i=1}^{l} (y_i - f(z,a))^2 \tag{9}$$

which one needs to minimize in order to find the regression estimate (i.e., the least square method). In order to estimate a density function from a given set of functions *p(x,a)* one uses the loss function (5). Putting this loss function into (8) one obtains the maximum likelihood method: the functional $R_{emp}(a) = \frac{1}{l}\sum_{i=1}^{l} \ln p(x_i, a)$ which one needs to minimize in order to find the approximation to the density. Since the ERM principle is a general formulation of these classical estimation problems, any theory concerning the ERM principle applies to the classical methods as well.

### *Structural Risk Minimization Induction Principle*

The ERM principle is intended for dealing with a large sample size. Indeed, the ERM principle can be justified by considering the inequalities.

*Theorem:* With probability at least$1-\eta$, the inequality

$$R(a) \leq R_{emp}(a) + \frac{B_{\varepsilon}}{2}\left(1 + \sqrt{1 + \frac{4R_{emp}(a)}{B_{\varepsilon}}}\right) \qquad (10)$$

holds true simultaneously for all functions of the set $0 \leq Q(z,a) \leq B, \quad a \in \Lambda$,

When *l/h* is large, the second summand on the right hand side of inequality (10) becomes small. The actual risk is then close to the value of the empirical risk. In this case, a small value of the empirical risk provides a small value of (expected) risk. However, if is small, then even a small $R_{emp}(a_l)$ does not guarantee a small value of risk. In this case the minimization for *R(a)* requires a new principle, based on the simultaneous minimization of two terms in (10) one of which depends on the value of the empirical risk while the second depends on the VC-dimension of the set of functions. To minimize risk in this case it is necessary to find a method which, along with minimizing the value of empirical risk, controls the VC-dimension of the learning machine. The following principle, which is called the principle of structural risk minimization (SRM), is intended to minimize the risk functional with respect to both empirical risk and VC-dimension of the set of functions.

**Machine Learning**

Learning denotes changes in a system that enable a system to do the same task more efficiently the next time or Learning is constructing or modifying representations of what is being experienced. Machine learning is a scientific discipline concerned with the design and development of algorithms that allow computers to evolve behaviors based on empirical data, such as from sensor data or databases. A learner can take advantage of examples (data) to capture characteristics of interest of their unknown underlying probability distribution. Data can be seen as examples that illustrate relations between observed variables. Discover new things or structure that is unknown to humans eg. data mining. A major focus of machine learning research is to automatically learn to recognize complex patterns and make intelligent decisions based on data.

***Types of Machine Learning***

Broadly, machine learning is classified into two categories i.e. supervised and unsupervised learning. Supervised learning generates a function that maps inputs to desired outputs based on labelled training data, where the desired output for each object is known. Approaches of supervised learning are classification and prediction. The prevalent techniques of supervised learning are Naïve Bayes classifier, Logistic Regression, Linear Discriminant Analysis, K-Nearest-Neighbour classifiers, Artificial Neural Networks, Support vector machine etc.

Unsupervised learning discovers underlying patterns in the data based on unlabelled training data. In other words if data has to be processed by machine learning methods, where the desired output is not known, then the learning task is called unsupervised. Approaches to unsupervised learning include clustering (e.g., k-means, hierarchical clustering)

***Selection of learning algorithms***

Major issues which needs special consideration in section of supervised learning algorithms

***a) Tradeoff between bias and variance:*** The prediction error is sum of bias and variance of the learning algorithms. Generally it is desirable that a learning algorithm with low bias should be flexible such that it can fit the data set but it should not be that flexible that it fit differently to each training data set due to its high variance. Therefore, it is necessary to adjust this tradeoff between bias and variance.

***b) Availability of dataset and complexity of function:*** In case, simple true function, learning algorithm with high bias and low variance will results reliable inferred function with the help of small amount of dataset. But in case of highly complex true function resulting from interactions within different components needs large amount of training dataset to build learning algorithm with low bias and high variance. Therefore, it is desirable for good learning algorithms to automatically adjust the bias/variance tradeoff based on the amount of data available and the apparent complexity of the function to be learned.

***c) Dimensions of input dataset:*** Large dimension of the dataset may create confusion and it may become difficult learning problem even if the true function depends on only small number of features. This will results in large variance. Hence, high input dimensionality typically requires tuning the classifier to have low variance and high bias. It is always desirable to apply feature selection procedures or dimensionality reduction techniques to get desirable output.

***d) Noisy output values:*** In case output values are incorrect beyond a limit due to response errors then the learning algorithm is expected to lead to undesirable inferred function. This is case where it is usually best to employ a high bias, low variance classifier.

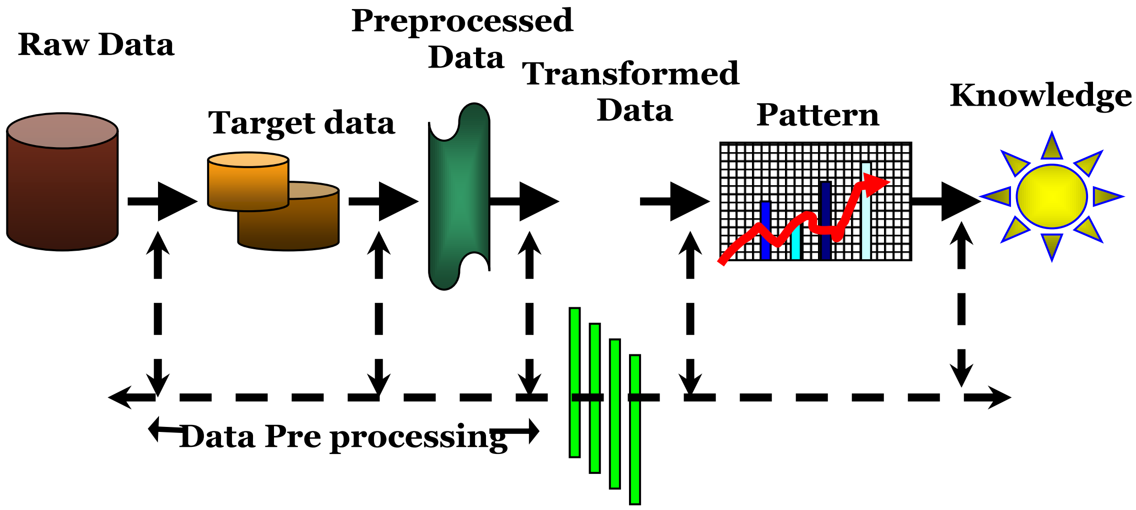
The selection of learning algorithms also depends on number of other factors such as (i) heterogeneity of the data, (ii) redundancy of data and (iii) linear and non-linear relationships among the factors etc.

## Data Mining

The field of data mining and knowledge discovery is emerging as a new, fundamental research area with important applications to science, engineering, medicine, business, and education. Data mining attempts to formulate analyze and implement basic induction processes that facilitate the extraction of meaningful information and knowledge from unstructured data. Data mining extracts patterns, changes, associations and anomalies from large data sets. Work in data mining ranges from theoretical work on the principles of learning and mathematical representations of data to building advanced engineering systems that perform information filtering on the web, find genes in DNA sequences, help understand trends and anomalies in economics and education, and detect network intrusion. Data mining is also a promising computational paradigm that enhances traditional approaches to discovery and increases the opportunities for breakthroughs in the understanding of complex physical and biological systems. Researchers from many intellectual communities have much to contribute to this field. These include the communities of machine learning, statistics, databases, visualization and graphics, optimization, computational mathematics, and the theory of algorithms.

### *The Process of Data Mining*

The data mining process is often characterized as a multi-stage iterative process involving data selection, data cleaning, and application of data mining algorithms, evaluation, and so forth. Here it is taken as process-oriented and break down into different steps:



***a) Exploring and Preprocessing:*** the initial steps of exploring, visualizing, and querying the data, to gain insight into the data in an interactive manner. Preprocessing steps such as variable selection, data focusing, and data validation can also be included in these initial steps.

***b) Modeling:*** the steps involved in (a) selecting the model representations that we seek to fit to the data (e.g., a tree, a linear function, a probability density model, etc.), (b) selecting the score functions that score different models with respect to the data, and (c) specifying the computational methods and algorithms to optimize the score function (e.g., greedy local search). These \components" combined together specify the data mining algorithm to be used. The components may be \precompiled" into a specific algorithm (e.g., CART or C4.5 decision tree implementations) or may be integrated in a \customized" manner for a specific application (much more common in the sciences).

***c) Mining:*** the step (often repeated) of actually running a particular data mining algorithm on a particular data set.

***d) Evaluating:*** the step (often ignored) of critically evaluating the quality of the output of the data mining algorithm from step 3, both the predictions of the model and the interpretation of the fitted model itself.

***e) Deploying:*** the step (rarely achieved) of putting a model from a data mining algorithm into routine predictive use, e.g., using the model continuously in real-time for scoring customers visiting an ecommerce Web site. A challenging (and under-appreciated) technical issue in this context is how and when models should be updated for such continuous data stream" applications.

***Recent Research Achievements***

The opportunities today in data mining rest solidly on a variety of research achievements, which were interdisciplinary in nature, resting on discoveries made by researchers from different disciplines working together collaboratively.

Neural Networks: Neural networks are systems inspired by the human brain. A basic example is provided by a back propagation network which consists of input nodes, output nodes, and intermediate nodes called hidden nodes. Initially, the nodes are connected with random weights. During the training, a gradient descent algorithm is used to adjust the weights so that the output nodes correctly classify data presented to the input nodes. The algorithm was invented independently by several groups of researchers.

Tree-based Classifiers: A tree is a convenient way to break large data sets into smaller ones. By presenting a learning set to the root and asking questions at each interior node, the data at the leaves can often be analyzed very simply. For example, a classifier to predict the likelihood that a credit card transaction is fraudulent may use an interior node to divide a training data set into two sets, depending upon whether or not five or fewer transactions were processed during the previous hour. After a series of such questions, each leaf can be labeled fraud/no-fraud by using a simple majority vote. Tree based classifiers were independently invented in information theory, statistics, pattern recognition and machine learning.

Graphical Models and Hierarchical Probabilistic Representations: A directed graph is a good means of organizing information about qualitative knowledge about conditional independence and causality gleamed from domain experts. Graphical models generalize Markov models and hidden Markov models, which have proved themselves to be a powerful modeling tool. Graphical models were independently invented by computational probabilists and artificial intelligence researchers studying uncertainty.

Ensemble Learning: Rather than use data mining to build a single predictive model, it is often better to build a collection or ensemble of models and to combine them, say with a simple, efficient voting strategy. This simple idea has now been applied in a wide variety of contexts and applications. In some circumstances, this technique is known to reduce variance of the predictions and therefore to decrease the overall error of the model.

Linear Algebra: Scaling data mining algorithms often depends critically upon scaling underlying computations in linear algebra. Recent work in parallel algorithms for solving linear system and algorithms for solving sparse linear systems in high dimensions are important for a variety of data mining applications, ranging from text mining to detecting network intrusions.

Large Scale Optimization: Some data mining algorithms can be expressed as large-scale, often non-convex, optimization problems. Recent work has provided parallel and distributed methods for large-scale continuous and discrete optimization problems, including heuristic search methods for problems too large to be solved exactly.

High Performance Computing and Communication: Data mining requires statistically intensive operations on large data sets. These types of computations would not be practical without the emergence of powerful SMP workstations and high performance clusters of workstations supporting protocols for high performance computing such as MPI and MPIO. Distributed data mining can require moving large amounts of data between geographically separated sites, something which is now possible with the emergence of wide area high performance networks.

Databases, Data Warehouses, and Digital Libraries: The most time consuming part of the data mining process is preparing data for data mining. This step can be stream-lined in part if the data is already in a database, data warehouse, or digital library, although mining data across different databases, for example, is still a challenge. Some algorithms, such as association algorithms, are closely connected to databases, while some of the primitive operations being built into tomorrow's data warehouses should prove useful for some data mining applications.

Visualization of Massive Data Sets: Massive data sets, often generated by complex simulation programs, required graphical visualization methods for best comprehension.

Recent advances in multi-scale visualization allow the rendering to be done far more quickly and in parallel, making these visualization tasks practical.

### *Research Challenges*

The amount of digital data has been exploding during past decade, while the number of scientists, engineers, and analysts available to analyze the data has been static. To bridge this gap requires the solution of fundamentally new research problems, which can be grouped into the following broad areas

- Developing a unifying theory of data mining
- Scaling up for high dimensional data and high speed data streams
- Mining sequence data and time series data
- Mining complex knowledge from complex data
- Data mining in a network setting
- Distributed data mining and mining multi-agent data
- Data mining for biological and environmental problems
- Data Mining process-related problems
- Security, privacy and data integrity
- Dealing with non-static, unbalanced and cost-sensitive data

## Machine Learning and data miming in Bioinformatics

Machine learning techniques are widely accepted as tool to perform tasks of molecular biology. Many machine learning approaches which have been used to solve important biological problems are briefly described below.

### *Gene prediction*

The problem of gene prediction is to first determine which regions in DNA are gene regions, and then to determine which parts of the gene regions are introns and exons. The predicted gene region is sensitive to the type of the algorithm. This is the typical problem of classifying DNA bases according to how they participate during transcription. Machine Learning techniques based on SVM have been successfully used in classifying DNA bases according to their role in transcription in nematode genome. A highly accurate gene-prediction system for eukaryotic genomes, called mGene which combines in an unprecedented manner the flexibility of generalized hidden Markov models (gHMMs) with the predictive power of modern machine learning methods, such as SVMs.

### *Splice site prediction*

Splice sites are locations in DNA which separate protein-coding regions (exons) from noncoding regions (introns). Accurate splice site detectors thus form important components of computational gene predictors. Splice site prediction can be considered as a classification problem with the classifier learnt from a labeled data set consisting of only local information around the potential splice site. Classification algorithms such as ANN, SVM have been used extensively.

### *Single nucleotide polymorphism (SNP)*

SNP is nothing but DNA sequence variation occurring in a single nucleotide in inter or intra genomic sequences. SNPs are important in crop and livestock breeding programs because a single or multiple SNPs may cause simple or complex diseases respectively. Recent discovery of SNP in genome-wide association (GWA) studies to revolutionize not only the process of genetic variation and disease detection but also the convention of preventative and curative medicine for future prospects. Genes are classified for a particular disease condition based on SNPs data. Various machine learning based classifiers such as logistic regression, naïve bayes classifier, SVM etc. have been used for this purpose.

### *Protein secondary structure prediction*

Protein structure prediction is of great interest to biologists because proteins are able to perform their functions based on their specific three-dimensional structures. Protein structure prediction is a difficult task because the number of possible protein structures is extremely large, and the physical basis of protein structural stability is not fully understood till now. Therefore, computational approaches have been developed to reveal the protein structure from the protein sequence information. Machine learning approaches such as neural networks and support vector machines have been used in protein secondary structure prediction with remarkable success.

### *Systems biology and modelling*

Systems biology approach allows researchers to move beyond a reductionist approach. This integrates and comprehends the interactions of multiple components interacting within the system. Understanding of the specific roles of various metabolites will give rise to strategy for the metabolic engineering to improve productivity. Large numbers of approaches have been proposed to model the behaviour of gene regulatory networks. These approaches are based on various machine learning methods along with other methods, such as graph theory, neural network, fuzzy logic, hidden markov model, bayesian belief network, boolean network and nonlinear ordinary differential equations.

# Analysis of Non-Coding Sequencing Data

**Sarika Sahu**

**ICAR- Indian Agricultural Statistics Research Institute, New Delhi**

## Abstract

Non-coding RNA (ncRNA) has emerged as a pivotal player in the intricate regulatory networks governing gene expression in agriculturally important crops. The diverse roles and regulatory mechanisms of ncRNAs in crop plants, shedding light on their impact on key biological processes. From microRNAs (miRNAs) modulating post-transcriptional gene silencing to long non-coding RNAs (lncRNAs) orchestrating chromatin remodelling and endogenous target mimicking (eTMs), these molecular entities act as fine-tuners of gene expression, influencing plant growth, development, and stress responses. Understanding the regulatory roles of ncRNAs presents a promising avenue for enhancing crop yield, quality, and resilience in the face of changing environmental conditions. The potential applications of ncRNAs in crop improvement strategies, including the development of RNA-based tools for targeted gene regulation. As researchers uncover the intricate web of non-coding RNA interactions from the transcriptome data, future directions in agricultural research are poised to harness this knowledge for the sustainable advancement of crop productivity, addressing global food security challenges in the 21st century.

Keywords: ncRNAs, miRNAs, lncRNAs, eTMs

## Introduction

Non-coding RNAs (ncRNAs) are RNA molecules that do not code for proteins. They are transcribed from DNA and can be categorized into two main types: long non-coding RNAs (lncRNAs) and small non-coding RNAs (sncRNAs). While sncRNAs are shorter than 200 nucleotides, lncRNAs are usually longer than 200 nucleotides. Non-coding RNAs have been found to play important roles in a variety of cellular processes, including gene expression, cell differentiation, and development.

One of the well-studied classes of sncRNAs are microRNAs (miRNAs). miRNAs are single-stranded RNA molecules that are about 21-25 nucleotides long. They play important roles in post-transcriptional regulation of gene expression by targeting mRNAs for degradation or translational repression. This means that miRNAs can control the amount of protein that is produced from a particular gene. miRNAs have been implicated in a variety of biological processes, including cell proliferation, differentiation, and apoptosis. Dysregulation of miRNA expression has been linked to various diseases, such as cancer, neurological disorders, and cardiovascular disease. Another type of sncRNA is the small interfering RNA (siRNA). Like miRNAs, siRNAs are about 21-25 nucleotides long and are involved in gene regulation by inducing degradation of specific mRNAs. However, siRNAs are usually exogenously introduced into cells for therapeutic purposes or for use in research. They can be used to specifically target and silence disease-causing genes or to study gene function in experimental systems. Piwi-interacting RNAs (piRNAs) are a class of sncRNAs that interact with a family of proteins known as Piwi proteins. piRNAs are typically longer than miRNAs or siRNAs and are expressed primarily in the germ cells of animals. They play important roles in protecting the genome from transposable elements (mobile genetic elements that can cause mutations) by inducing their silencing or degradation. piRNAs have also been implicated in other processes such as epigenetic regulation and germ cell development.

In addition to sncRNAs, lncRNAs have also been found to play important roles in various biological processes. They are involved in gene regulation at multiple levels, including transcription, splicing, and chromatin remodelling. lncRNAs can interact with DNA, RNA, and proteins to modulate gene expression. Dysregulation of lncRNA expression has been implicated in a variety of diseases, such as cancer, cardiovascular disease, and neurological disorders.

One example of a lncRNA is Xist, which is involved in X chromosome inactivation in female mammals. Xist is expressed from one of the two X chromosomes in female cells and coats the same chromosome it is transcribed from, leading to silencing of most genes on that chromosome. Another example is HOTAIR, which is involved in regulating gene expression during development and has been found to be dysregulated in various types of cancer.

In conclusion, non-coding RNAs are a diverse group of RNA molecules that play important roles in a variety of cellular processes. While sncRNAs like miRNAs and siRNAs are involved in post-transcriptional regulation of gene expression, piRNAs are involved in transposon silencing in germ cells. lncRNAs, on the other hand, are involved in gene regulation at multiple levels and have been implicated in various diseases. With the continued development of new technologies for studying RNA, we can expect to uncover many more functions and roles for these fascinating molecules in the future.

Long non-coding RNAs (lncRNAs) are a diverse class of RNA molecules that have been found to play important roles in gene regulation and other biological processes in many different organisms, including plants. In this discussion, we will explore the current understanding of lncRNAs in plants, their functions, and their potential applications in agriculture.

Plant lncRNAs are typically longer than 200 nucleotides and are transcribed from intergenic regions, introns, and other non-coding regions of the genome. They can be classified into several different categories based on their genomic origin and structure, including natural antisense transcripts (NATs) and long intergenic non-coding RNAs (lincRNAs). NATs are RNA molecules that are complementary to other RNA transcripts and transcribed from the opposite DNA strand. They may also overlapping with the sequence of protein-coding genes. These antisense transcripts can be transcribed in the opposite direction to the sense (coding) strand of the DNA, forming RNA-RNA duplexes with their complementary sense transcripts. One of the most well-studied plant lncRNAs involved in growth and development is COOLAIR, a NAT of the FLOWERING LOCUS C (FLC) gene in Arabidopsis thaliana. FLC is a key regulator of flowering time, and the expression of COOLAIR promotes FLC mRNA decay, leading to earlier flowering. COOLAIR is also involved in regulating the expression of other genes related to plant development, such as genes involved in the biosynthesis of gibberellins, a class of plant hormones that promote stem elongation and other growth processes.

Moreover, LincRNAs are transcribed from intergenic regions of the genome and can interact with DNA, RNA, and proteins to modulate gene expression. They can act as scaffolds for the assembly of regulatory complexes, as well as serve as guides for chromatin-modifying enzymes. In rice, a lincRNA called NERICA1 is involved in promoting nodulation in response to symbiotic bacteria by interacting with chromatin-modifying enzymes to regulate gene expression. In addition to their roles in plant growth and development, lncRNAs have also been implicated in stress responses. For example, a lincRNA called COLDAIR in Arabidopsis is involved in the regulation of the COLD-REGULATED (COR) genes in response to cold stress. COLDAIR interacts with a transcription factor called CBF1 to promote the expression of COR genes, which are involved in protecting plants from freezing damage. Another lncRNA involved in the regulation of flowering time is IPS1 (Induced by Phosphate Starvation 1) in Arabidopsis. IPS1 is a lincRNA that is induced by phosphate starvation and negatively regulates the expression of miR399, a microRNA that targets a gene involved in phosphate homeostasis. The downregulation of miR399 by IPS1 promotes the expression of genes involved in phosphate uptake and transport, leading to earlier flowering.

LINC5 is another lincRNA involved in the regulation of flowering time in Arabidopsis. LINC5 is specifically expressed in the shoot apical meristem, where it interacts with the transcription factor WUSCHEL (WUS) to promote its expression. WUS is a key regulator of stem cell maintenance and differentiation in the shoot apical meristem, and the expression of LINC5 is required for normal shoot development. Similarly, in rice, a lincRNA called LDMAR is involved in the regulation of lateral root development. LDMAR is specifically expressed in lateral root primordia and promotes the expression of genes involved in lateral root development. Knockdown of LDMAR leads to a reduction in the number of lateral roots, indicating its importance in this process.

The roles of plant lncRNAs in development have also been extensively studied. In maize, a lincRNA called Zm401 is involved in regulating the expression of key genes during the transition from vegetative growth to reproductive development. Zm401 interacts with a chromatin-modifying complex to regulate the expression of genes involved in flowering and other developmental processes.

One study identified 285 lncRNAs in potato leaves and tubers and analysed their expression patterns during potato development. The researchers found that many lncRNAs were differentially expressed in different tissues and developmental stages, indicating their potential roles in regulating potato growth and development.

Another study investigated the role of a potato lncRNA called lncRNA1604 in response to potato virus Y (PVY) infection. The researchers found that lncRNA1604 was induced in response to PVY infection and was involved in regulating the expression of genes involved in defence responses. Knockdown of lncRNA1604 resulted in increased susceptibility to PVY infection, indicating its role in potato resistance to viral infections.

In addition to their roles in development and stress responses, lncRNAs in potato have also been implicated in other biological processes. For example, a recent study identified a potato lncRNA called StTILLING1 that was involved in regulating the production of starch in potato tubers. Knockdown of StTILLING1 resulted in reduced starch content and altered starch granule morphology, indicating its role in starch synthesis.

Overall, the study of lncRNAs in plants is still in its early stages, and much remains to be learned about their functions and mechanisms of action. However, the identification of lncRNAs involved in growth and development processes in plants provides new insights into the regulatory networks underlying these processes and offers new targets for crop improvement and genetic engineering.

Circular RNAs (circRNAs) are a relatively new class of ncRNAs that are formed by back-splicing events, in which a downstream splice acceptor is joined to an upstream splice donor. circRNAs can act as sponges for microRNAs (miRNAs) and other RNA-binding proteins, thereby regulating gene expression. In tomato, a circRNA called ciRs-7 is involved in regulating fruit ripening by sequestering miR-7, which targets several genes involved in fruit ripening. Some of the known functions of circRNAs in plants include regulating gene expression at both the transcriptional and post-transcriptional levels, modulating alternative splicing, and participating in stress responses. For example, a circRNA called circRNA_022653 has been shown to regulate the expression of the transcription factor WRKY40 in response to salt stress in Arabidopsis thaliana. In addition, circRNAs have been implicated in plant development, particularly in the regulation of flowering time. A circRNA called circFTO has been found to play a role in the photoperiodic flowering pathway in Arabidopsis, by regulating the expression of a key flowering-time regulator called CONSTANS.

## Conclusion

The intricate regulatory roles of ncRNAs (lncRNA, miRNAs, circRNAs) in agriculturally important crops underscore their significance in shaping plant development, stress responses, and overall productivity. As we unveil the complex interplay of these molecular entities, the potential for harnessing ncRNAs as tools for crop improvement becomes increasingly evident. Future research directions should focus on elucidating specific ncRNA functions and developing innovative strategies to leverage their regulatory prowess for sustainable agriculture, ultimately contributing to global food security in the face of environmental challenges. The evolving landscape of ncRNA research holds promise for unlocking novel avenues in crop science, paving the way for precision agriculture and resilient crop varieties.

## References

1. Bader GD, Hogue CW. (2003). An automated method for finding molecular complexes in large protein interaction networks. *BMC bioinformatics*. 4(1):1-27.
2. Baulcombe, D. (2004). RNA silencing in plants. *Nature* 431, 356–363.

3. Denman, R. B. (1993). Using RNAFOLD to predict the activity of small catalytic RNAs. *BioTechniques* 15, 1090–5.
4. Dong, P., Wang, H., Fang, T., Wang, Y., and Ye, Q. (2019). Assessment of extracellular antibiotic resistance genes (eARGs) in typical environmental samples and the transforming ability of eARG. *Environment International* 125, 90–96.
5. Fujita, Y., Fujita, M., Satoh, R., Maruyama, K., Parvez, M. M., Seki, M., *et al*. (2005). AREB1 Is a Transcription Activator of Novel ABRE-Dependent ABA Signaling That Enhances Drought Stress Tolerance in *Arabidopsis*. *The Plant Cell* 17, 3470–3488.
6. Gao Y, Wang J, Zheng Y, Zhang J, Chen S, Zhao F. (2016). Comprehensive identification of internal structure and alternative splicing events in circular RNAs. *Nature Communications* 7:12060
7. Gkirtzou K, Tsamardinos I, Tsakalides P, Poirazi P. (2010). MatureBayes: a probabilistic algorithm for identifying the mature miRNA within novel precursors. *PloS one*. 5(8):e11843.
8. Jain P., Sharma V., Dubey H., Singh P.K., Kapoor R., Kumari M., Singh J., Pawar D., Bisht D., Solanke A.U., Mondal T.K., Sharma T.R. (2017) Identification of long non-coding RNA in rice lines resistant to Rice blast pathogen *Magnaporthe oryzae.* Bioinformation. 13:249-55.
9. Jeyaraj, A., Liu, S., Zhang, X., Zhang, R., Shangguan, M., and Wei, C. (2017). Genome-wide identification of microRNAs responsive to Ectropis oblique feeding in tea plant (Camellia sinensis L.). *Scientific Reports* 7, 13634.
10. Liu, J., Jung, C., Xu, J., Wang, H., Deng, S., Bernad, L., Arenas-Huertero, C., and Chua, N.H. (2012). Genome-wide analysis uncovers regulation of long intergenic noncoding RNAs in Arabidopsis. Plant Cell 24, 4333–4345
11. Meng X, Li X, Zhang P, Wang J, Zhou Y, Chen M. (2017). Circular RNA: an emerging key player in RNA world. *Briefings in bioinformatics*. 18(4):547-57.
12. Ramírez Gonzales L, Shi L, Bergonzi SB, Oortwijn M, Franco-Zorrilla JM, Solano-Tavira R, Visser RG, Abelenda JA, Bachem CW. (2021). Potato CYCLING DOF FACTOR 1 and its lncRNA counterpart StFLORE link tuber development and drought response. The Plant Journal. 105(4):855-69.
13. Sahu, S, Rao, A R, Pandey, J, Gaikwad, K, Ghoshal, S, and Mohapatra, T (2018). Genome-wide identification and characterization of lncRNAs and miRNAs in cluster bean (Cyamopsis tetragonoloba). *Gene* 667, 112–121.
14. Tian R, Sun X, Liu C, Chu J, Zhao M, Zhang WH. A Medicago truncatula lncRNA MtCIR1 negatively regulates response to salt stress. Planta. 2023, 257(2):32.
15. Zhang G, Diao S, Zhang T, Chen D, He C, Zhang J. (2019). Identification and characterization of circular RNAs during the sea buckthorn fruit development. RNA biology. 16(3):354-61.

# PERL Programming for Bioinformatics

**K. K. Chaturvedi**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

### What is Perl?

Perl stands for "Practical Extraction and Report Language" Perl is the natural outgrowth of a project started by Larry Wall in 1986. Originally intended as a configuration and control system for six VAXes and six SUNs located on opposite ends of the country, it grew into a more general tool for system administration on many platforms. Since its unveiling to programmers at large, it has become the work of a large body of developers. Larry Wall, however, remains its principle architect. Although the first platform Perl inhabited was UNIX, it has since been ported to over 70 different operating systems including, but not limited to, Windows 9x/NT/2000, MacOS, VMS, Linux, UNIX (many variants), BeOS, LynxOS, and QNX.

### Uses of Perl

1. Tool for general system administration

2. Processing textual or numerical data

3. Database interconnectivity

4. Common Gateway Interface (CGI/Web) programming

5. Driving other programs! (FTP, Mail, WWW, OLE)

## Philosophy & Idioms

### The Virtues of a Programmer

Perl is a language designed to cater to the three chief virtues of a programmer.

- Laziness - develop reusable and general solutions to problems
- Impatience - develop programs that anticipate your needs and solve problems for you.
- Hubris - write programs that you want other people to see (and be able to maintain)

### There are many means to the same end

Perl provides you with more than enough rope to hang yourself. Depending on the problem, there may be several "official" solutions. Generally those that are approached using "Perl idioms" will be more efficient.

### Resources

· The Perl Institute (http://www.perl.org)

· The Comprehensive Perl Archive Network (http://www.cpan.org)

· The Win32 port of Perl (http://www.activestate.com/ActivePerl/)

## Perl Basics

### Script names

While generally speaking you can name your script/program anything you want, there are a number of conventional extensions applied to portions of the Perl bestiary:

**.pm** - Perl modules

**.pl** - Perl libraries (and scripts on UNIX)

**.plx** - Perl scripts

### Language properties

- Perl is an interpreted language – program code is interpreted at run time. Perl is unique among interpreted languages, though. Code is compiled by the interpreter before it is actually executed.
- Many Perl idioms read like English
- Free format language – whitespace between tokens is optional
- Comments are single-line, beginning with **#**
- Statements end with a semicolon (**;**)
- Only subroutines and functions need to be explicitly declared
- Blocks of statements are enclosed in curly braces **{}**
- A script has no "**main()**"

## Data Types & Variables

### Basic Types

The basic data types known to Perl are scalars, lists, and hashes. Scalar **$foo** Simple variables that can be a number, a string, or a reference. A scalar is a "thingy." List **@foo** An ordered array of scalars accessed using a numeric subscript. **$foo[0]** Hash **%foo** An unordered set of key/value pairs accessed using the keys as subscripts. **$foo{key}** Perl uses an internal type called a typeglob to hold an entire symbol table entry. The effect is that scalars, lists, hashes, and filehandles occupy separate namespaces (i.e., **$foo[0]** is not part of **$foo** or of **%foo**). The prefix of a typeglob is *****, to indicate "all types." Literals are symbols that give an actual value, rather than represent possible values, as do variables. For example in **$foo = 1**, **$foo** is a scalar variable and **1** is an integer literal. Variables have a value of undef before they are defined (assigned). The upshot is that accessing values of a previously undefined variable will not (necessarily) raise an exception.

### Variable Contexts

Perl data types can be treated in different ways depending on the context in which they are accessed. Scalar Accessing data items as scalar values. In the case of lists and hashes, $foo[0] and $foo{key}, respectively. Scalars also have numeric, string, and don't-care contexts to cover situations in which conversions need to be done. List Treating lists and hashes as atomic objects

Boolean Used in situations where an expression is evaluated as true or false. (Numeric: 0=false; String: null=false, Other: undef=false) Void Does not care (or want to care) about return value Interpolative Takes place inside quotes or things that act like quotes

**Special Variables (defaults)**

Some variables have a predefined and special meaning to Perl. A few of the most commonly used ones are listed below:

**$_** The default input and pattern-searching space

**$0** Program name

**$$** Current process ID

**$!** Current value of errno

**@ARGV** Array containing command-line arguments for the script

**@INC** The array containing the list of places to look for Perl scripts to

be evaluated by the do, require, or use constructs

**%ENV** The hash containing the current environment

**%SIG** The hash used to set signal handlers for various signals

**Scalars**

Scalars are simple variables that are either numbers or strings of characters. Scalar variable names begin with a dollar sign followed by a letter, then possibly more letters, digits, or underscores. Variable names are case-sensitive.

**Numbers**

Numbers are represented internally as either signed integers or double precision floating point numbers. Floating point literals are the same used in C. Integer literals include decimal (255), octal (0377), and hexadecimal (0xff) values.

**Strings**

Strings are simply sequences of characters. String literals are delimited by quotes: Single quote **'string'** Enclose a sequence of characters Double quote **"string"** Subject to backslash and variable interpolation Back quote **`command`** Evaluates to the output of the enclosed command The backslash escapes are the same as those used in C:

**\n** Newline **\e** Escape

**\r** Carriage return \\ Backslash

**\t** Tab **\"** Double quote

**\b** Backspace \' Single quote

In Windows, to represent a path, use either "**c:\\temp**" (an escaped backslash) or

"**c:/temp**" (UNIX-style forward slash). Strings can be concatenated using the "**.**" operator: **$foo = "hello" . "world";**

### Basic I/O

The easiest means to get operator input to your program is using the “diamond” operator:

**$input = <>;**The input from the diamond operator includes a newline (\n). To get rid of this peskycharacter, use either **chop()** or **chomp(). chop()** removes the last character of thestring, while **chomp()** removes any line-ending characters (defined in the specialvariable **$/**). If no argument is given, these functions operate on the **$_** variable.To do the converse, simply use Perl’s print function:

**print $output.”\n”;**

### Basic Operators

### Arithmetic

Example Name Result

**$a + $b** Addition Sum of **$a** and **$b**

**$a * $b** Multiplication Product of **$a** and **$b**

**$a % $b** Modulus Remainder of **$a** divided by **$b**

**$a ** $b** Exponentiation **$a** to the power of **$b**

### String

Example Name Result

**$a . “string”** Concatenation String built from pieces

**“$a string”** Interpolation String incorporating the value of **$a**

**$a x $b** Repeat String in which **$a** is repeated **$b** times

### Assignment

The basic assignment operator is “**=**”: **$a = $b**. Perl conforms to the C idiom that lvalue operator= expression is evaluated as: lvalue = lvalue operator expression So that **$a *= $b** is equivalent to **$a = $a * $b $a += $b $a = $a + $b** This also works for the string concatenation operator: **$a .= “\n”**

### Autoincrement and Autodecrement

The autoincrement and autodecrement operators are special cases of the assignment operators, which add or subtract 1 from the value of a variable:

**++$a, $a++** Autoincrement Add 1 to $a

**--$a, $a--** Autodecrement Subtract 1 from $a

### Logical

Conditions for truth:Any string is true except for “” and “0”Any number is true except for 0 Any reference is trueAny undefined value is false Example Name Result **$a && $b** And True if both

**$a** and **$b** are true **$a || $b** Or **$a** if **$a** is true; **$b** otherwise **!$a** Not True if **$a** is not true **$a and $b** And True if both **$a** and **$b** are true **$a or $b** Or **$a** if **$a** is true; **$b** otherwise **not $a** Not True if **$a** is not true Logical operators are often used to “short circuit” expressions, as in: **open(FILE,”< input.dat”) or die “Can’t open file”;**

### Comparison

Comparison Numeric String Result Equal **== eq** True if $a equal to $b Not equal **!= ne** True if $a not equal to $b Less than **< lt** True if $a less than $bGreater than **> gt** True if $a greater than $b Less than or equal **<= le** True if $a not greater than $b Comparison **<=> cmp** 0 if $a and $b equal1 if $a greater -1 if $b greater

### Operator Precedence

Perl operators have the following precedence, listed from the highest to the lowest, where operators at the same precedence level resolve according to associativity:

Associativity Operators Description

Left Terms and

list operators

Left **->** Infix dereference operator

**++**

**--**

Auto-increment

Auto-decrement

Right

Right

Right

\

**! ~**

**+ -**

Reference to an object (unary)

Unary negation, bitwise complement

Unary plus, minus

Left

Left

**=~**

**!~**

Binds scalar to a match pattern

Same, but negates the result

Left *** / % x** Multiplication, Division, Modulo, Repeat

Left **+ - .** Addition, Subtraction, Concatenation

Left >> << Bitwise shift right, left

< > <= >=

**lt gt le ge**

Numerical relational operators

String relational operators

== != <=>

**eq ne cmp**

Numerical comparison operators

String comparison operators

Left **&** Bitwise AND

Left **|** ^ Bitwise OR, Exclusive OR

Left **&&** Logical AND

Left **||** Logical OR

In scalar context, range operator

In array context, enumeration

Right **?:** Conditional (if ? then : else) operator

Right **= += -= etc** Assignment operators

Left **,**

=>

Comma operator, also list element separator

Same, enforces left operand to be string

Right **not** Low precedence logical NOT

Right **and** Low precedence logical AND

Right **or xor** Low precedence logical OR

Parentheses can be used to group an expression into a term.

A list consists of expressions, variables, or lists, separated by commas. An array variable or an array slice many always be used instead of a list.

## Control Structures

### Statement Blocks

A statement block is simply a sequence of statements enclose in curly braces:

```
{
first_statement;
second_statement;
last_statement
}
```

### Conditional Structures (If/elsif/else)

The basic construction to execute blocks of statements is the **if** statement. The **if** statement permits execution of the associated statement block if the test expression evaluates as true. It is important to note that unlike many compiled languages, it is necessary to enclose the statement block in curly braces, even if only one statement is to be executed.The general form of an if/then/else type of control statement is as follows:

```
if (expression_one) {
true_one_statement;
} elsif (expression_two) {
true_two_statement;
} else {
all_false_statement;
}
```

### Loops

Perl provides several different means of repetitively executing blocks of statements.

### While

The basic while loop tests an expression before executing a statement block

```
while (expression) {
statements;
}
```

### Until

The until loop tests an expression at the end of a statement block; statements will be executed until the expression evaluates as true.

```
until (expression) {
statements;
}
```

**Do while**

A statement block is executed at least once, and then repeatedly until the test expression is false.

```
do {
statements;
} while (expression);
```

**Do until**

A statement block is executed at least once, and then repeatedly until the test expression is true.

```
do {
statements;
} until (expression);
```

**For**

The for loop has three semicolon-separated expressions within its parentheses. These expressions function respectively for the initialization, the condition, and re-initialization expressions of the loop. The for loop

```
for (initial_exp; test_exp; reinit_exp) {
statements;
}
```

This structure is typically used to iterate over a range of values. The loop runs until the **test_exp** is false.

```
for ($i; $i<10;$i++) {
print $i;
}
```

### Foreach

The foreach statement is much like the for statement except it loops over the elements of a list:

```
foreach $i (@some_list) {
statements;
}
```

### Indexed Arrays (Lists)

A list is an ordered set of scalar data. List names follow the same basic rules as for scalars. A reference to a list has the form **@foo**.

### List literals

List literals consist of comma-separated values enclosed in parentheses:

**(1,2,3)**

**(“foo”,4.5)**

A range can be represented using a list constructor function (such as “**..**”):

**(1..9) = (1,2,3,4,5,6,7,8,9)**

**($a..$b) = ($a, $a+1, … , $b-1,$b)**

In the case of string values, it can be convenient to use the “quote-word” syntax

**@a = (“fred”,”barney”,”betty”,”wilma”);**

**@a = qw( fred barney betty wilma );**

### Accessing List Elements

List elements are subscripted by sequential integers, beginning with 0

**$foo[5]** is the sixth element of **@foo**

The special variable **$#foo** provides the index value of the last element of **@foo**.

A subset of elements from a list is called a slice.

**@foo[0,1]** is the same as **($foo[0],$foo[1])**

You can also access slices of list literals:

**@foo = (qw( fred barney betty wilma ))[2,3]**

**List operators and functions**

Many list-processing functions operate on the paradigm in which the list is a stack. The highest subscript end of the list is the "top," and the lowest is the bottom.

**push** Appends a value to the end of the list

**push(@mylist,$newvalue)**

**pop** Removes the last element from the list (and returns it)

**pop(@mylist)**

**shift** Removes the first element from the list (and returns it)

**shift(@mylist)**

**unshift** Prepends a value to the beginning of the list

**unshift(@mylist,$newvalue)**

**splice** Inserts elements into a list at an arbitrary position

**splice(@mylist,$offset,$replace,@newlist)**

The **reverse** function reverses the order of the elements of a list

**@b = reverse(@a);**

The **sort** function sorts the elements of its argument as strings in ASCII order. You can also customize the sorting algorithm if you want to do something special.

**@x = sort(@y);**

The **chomp** function works on lists as well as scalars. When invoked on a list, it removes newlines (record separators) from each element of its argument.

**Associative Arrays (Hashes)**

A hash (or associative array) is an unordered set of key/value pairs whose elements are indexed by their keys. Hash variable names have the form **%foo**.

**Hash Variables and Literals**

A literal representation of a hash is a list with an even number of elements (key/value pairs, remember?).

**%foo = qw( fred wilma barney betty );**

**%foo = @foolist;**

To add individual elements to a hash, all you have to do is set them individually:

```
$foo{fred} = “wilma”;
$foo{barney} = “betty”;
```

You can also access slices of hashes in a manner similar to the list case:

```
@foo{“fred”,”barney”} = qw( wilma betty );
```

**Hash Functions**

The **keys** function returns a list of all the current keys for the hash in question.

```
@hashkeys = keys(%hash);
```

As with all other built-in functions, the parentheses are optional:

```
@hashkeys = keys %hash;
```

This is often used to iterate over all elements of a hash:

```
foreach $key (keys %hash) {
print $hash{$key}.”\n”;
}
```

In a scalar context, the **keys** function gives the number of elements in the hash. Conversely, the **values** function returns a list of all current values of the argument hash:

```
@hashvals = values(%hash);
```

The **each** function provides another means of iterating over the elements in a hash:

```
while (($key, $value) = each (%hash)) {
statements;
}
```

You can remove elements from a hash using the **delete** function:

```
delete $hash{‘key’};
```

# Overview of Metagenomics Data Analysis

**Mohammad Samir Farooqi and Sudhir Srivastava**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

Metagenomics is the study of overall genomes present in any environment without the need for prior individual identification or amplification. It encompasses microbial communities sampled directly from their natural environment, without prior culturing. Community genomics, environmental genomics, and population genomics are synonyms for the same approach. Metagenomics term was first used by Jo Handelsman *et al.* and first appeared in publication in 1998. The field initially started with the cloning of environmental DNA, followed by functional expression screening and was then quickly complemented by direct random shotgun sequencing of environmental DNA. The idea of cloning DNA directly from environmental samples was first proposed by Pace in 1991.There has been remarkable progress in this field of research due to recent advances in Next Generation Sequencing (NGS) technologies. Since over 99.8% of microbes in some environments are still far from culturing in the media, metagenomics offers a path to the study of microbial community structure, phylogenetic composition, species diversity and abundance, metabolic capacity and functional diversity.

Metagenomics helps in knowing about the functional gene composition of the microbial communities and thus gives more information about the phylogenetic surveys, which are more often based on the diversity of one gene like 16s rRNA gene. It gives genetic information on potentially novel biocatalysts or enzymes, genomic linkages between function and phylogeny for uncultured organisms, and evolutionary profiles of community function and structure. So it acts as novel tool for generating novel hypothesis of microbial function.

Majority of microorganisms have not been cultivated in the laboratory, and almost all of our knowledge of microbial life is based on organisms raised in pure culture. Metagenomics provides an additional set of tools to study uncultured species. Metagenomics entails extraction of DNA from a community so that all of the genomes of organisms in the community are pooled. These genomes are usually fragmented and cloned into an organism that can be cultured to create 'metagenomic libraries', and these libraries are then subjected to analysis based on DNA sequence or on functions conferred on the surrogate host by the metagenomic DNA.

For a typical sequence-based metagenome project one need to go through sampling and processing, sequencing technology, assembly, binning, annotation, experimental design, statistical analysis, and data storage and sharing.

These steps are described as follow:

## Sampling and Processing

DNA extracted should represent all cell present in the sample and sufficient amount of high-quality nucleic acids must be obtained for subsequent library production and sequencing. Also processing requires specific protocols for each sample type. The physical and chemical structure of each microbial community affects the quality, size, and amount of microbial DNA that can be extracted.

## Sequencing Technology

High-throughput sequencing technologies has improved the capabilities of metagenomic studies to a greater strength but at the same time, it has led to generation of huge and big data sets that largely require high end algorithms and computational tools for data analysis and storage. Metagenome sequencing, also called shotgun sequencing, refers to sequencing DNA fragments extracted from microbial populations. Over the past few years metagenomic shotgun sequencing has gradually shifted from classical Sanger sequencing technology to next-generation sequencing (NGS). However, Sanger sequencing is still best because of its low error rate, long read length (> 700 bp) and large insert sizes (e.g. >30 Kb for fosmids or bacterial artificial chromosomes (BACs)). The only drawback associated is the labor intensive cloning process.

## Bioinformatics Approach

Metagenomic projects running worldwide pose several levels of challenges with respect to the processing, analyzing and storing huge data being accumulated. Some of the major computational challenges include the assembly of the whole data, phylogenetic surveys, gene finding and comparative metagenomic analysis for the metabolic pathways.

The data generated by metagenomics experiments are both enormous and inherently noisy. Collecting, curating, and extracting useful biological information from datasets as well as pre-filtering steps in which low-quality sequences and sequences of probable eukaryotic origin (especially in metagenomes of human origin) are removed.

### *Assembly*

DNA sequence data from genomic and metagenomic projects are essentially the same, but genomic sequence data offers higher coverage while metagenomic data is usually highly non redundant. Furthermore, the increased use of second-generation sequencing technologies with short read lengths means that much of future metagenomic data will be error-prone. Taken in combination, these factors make the assembly of metagenomic sequence reads into genomes difficult and unreliable. Mis-assemblies are caused by the presence of repetitive DNA sequences that make assembly especially difficult because of the difference in the relative abundance of species present in the sample. Mis-assemblies can also involve the combination of sequences from more than one species into chimeric contigs.

Two strategies can be employed for metagenomics samples:

i) Reference-based assembly (co-assembly)

ii) De novo assembly

Reference-based assembly can be done with software packages such as Newbler (Roche), AMOS (http://sourceforge.net/projects/amos/ ), or MIRA. It works well, if the metagenomic dataset contains sequences where closely related reference genomes are available. De novo assembly typically requires larger computational resources. Tools based on the de Bruijn graphs was specifically created to handle very large amounts of data. Machine requirements for the de Bruijn assemblers Velvet or SOAP are still significantly higher than for reference-based assembly (co-assembly), often requiring hundreds of gigabytes of memory in a single machine and run times frequently being days.

In metagenomics single reads have generally lower quality and hence lower confidence in accuracy than do multiple reads that cover the same segment of genetic information. Therefore, merging reads increases the quality of information. So in a complex community with low sequencing depth or coverage, it is unlikely to actually get many reads that cover the same fragment of DNA. Hence assembly may be of limited value for metagenomics. Hence there is a need for metagenomic assembly to obtain high-confidence contigs that enable the study of, e.g., major repeat classes.

### *Binning*

Taxonomic binning is another problem in metagenomics analysis. Sequence binning refers to the separation of sequences into taxon specific groups. A binning step may be part of the assembly process of metagenomic data or may be used for separating the genomes of a few members in order to study the biological processes carried by each one of them. Various algorithms have been developed, which employ two types of information contained within a given DNA sequence.

i) First compositional binning makes use of the fact that genomes have conserved nucleotide composition (e.g. a certain GC or the particular abundance distribution of k-mers).

ii) Secondly, the unknown DNA fragment might encode for a gene and the similarity of this gene with known genes in a reference database can be used to classify and hence bin the sequence.

Important considerations for using any binning algorithm are the type of input data available and the existence of a suitable training datasets or reference genomes. In general, composition-based binning is not reliable for short reads, as they do not contain enough information. It can however be improved, if training datasets (e.g. a long DNA fragment of known origin) exist and that is used to define a compositional classifier. These “training” fragments can either be derived from assembled data or from sequenced fosmids and should ideally contain a phylogenetic marker (such as rRNA gene) that can be used for high-resolution, taxonomic assignment of the binned fragment.

### *Annotation*

For annotation of metagenomics two approaches are used for annotation of coding regions in the assembled contigs. First, if assembly has produced large contigs and reconstructed genomes are the objective of the study then it is preferable to use existing pipelines for genome annotation, such as RAST or IMG. For this, minimal contigs length of 30,000 bp or longer are required. Second, annotation can be performed on the entire community and relies on unassembled reads or short contigs. Here the tools for genome annotation are significantly less useful than those specifically developed for metagenomic analyses.

## Experimental Design and Statistical Analysis

For the reduction of sequencing cost and a much wider appreciation of the utility of metagenomics to address fundamental questions in microbial ecology require proper experimental designs with appropriate replication and statistical analysis. The data from multiple metagenomic shotgun-sequencing projects can be reduced to tables, where the columns represent samples and the rows indicate either a taxonomic group or a gene function (or groups thereof) and the fields containing abundance or presence/absence data. As metagenomic data often contain many more species or gene functions then the number of samples taken, so appropriate corrections for multiple hypothesis testing have to be implemented (e.g. Bonferroni correction for t-test based analyses).

Sometimes variation between sample types can be due to true biological variation and technical variation and this should be carefully considered when planning the experiment. One should kept in mind that many microbial systems are highly dynamic, so temporal aspects of sampling can have a substantial impact on data analysis and interpretation. Taking multiple samples and then pooling them will lose all information on variability and hence will be of little use for statistical purposes. Ultimately, good experimental design of metagenomic projects will facilitate integration of datasets into new or existing ecological theories. One of the ultimate aims of metagenomics is to link functional and phylogenetic information to the chemical, physical, and other biological parameters that characterize an environment.

### Sharing and Storage of Data

Data sharing is important for the genomic research, there is a requirement for whole new level of organization and collaboration to provide metadata and centralized services (e.g., IMG/M, CAMERA and MG-RAST) as well as sharing of both data and computational results. Once this has been achieved, researchers will be able to download intermediate processed results from any one of the major repositories for local analysis or comparison. A suite of standard languages for metadata is currently provided by the Minimum Information about any (x) Sequence checklists (MIxS). MIxS is an umbrella term to describe MIGS (the Minimum Information about a Genome Sequence), MIMS (the Minimum Information about a Metagenome Sequence) and MIMARKS (Minimum Information about a MARKer Sequence) and contains standard formats for recording environmental and experimental data. The latest of these checklists, MIMARKS builds on the foundation of the MIGS and MIMS checklists, by including an expansion of the rich contextual information about each environmental sample.

The US National Center for Biotechnology Information (NCBI) is mandated to store all metagenomic data, however, the sheer volume of data being generated means there is an urgent need for appropriate ways of storing vast amounts of sequences. As the cost of sequencing continues to drop while the cost for analysis and storing remains more or less constant, selection of data storage in either biological (i.e. the sample that was sequenced) or digital form in (de-) centralized archives might be required. Ongoing work and successes in compression of (meta-) genomic data, help in the storage of digital information cost-efficiently.

### Applications of Metagenomics

Among the enormous applications of metagenomics the most important ones include environmental studies, human health, identification of novel microbes, genes, pathways and mechanisms of their survival, biodegradation of sewage, ocean pollutants, plastics, garbage, energy generation and bio-fuels and biotechnological and industrial implications of the huge meta-sequence data coming out from the unseen microbial communities.

#### *Community Metabolism*

In many bacterial communities, natural or engineered (such as bioreactors), there is significant division of labor in metabolism (Syntrophy), during which the waste products of some organisms are metabolites for others. Eg. in methanogenic bioreactor.

#### *Metatranscriptomics*

Metagenomics allows researchers to access the functional and metabolic diversity of microbial communities, but it cannot show which of these processes are active. The extraction and analysis of metagenomic mRNA (the metatranscriptome) provides information on the regulation and expression profiles of complex communities apart from its technical difficulties (e g. the short half-life of mRNA).

### *Viruses*

Metagenomic sequencing is particularly useful in the study of viral communities. As viruses lack a shared universal phylogenetic marker (as 16S RNA for bacteria and *archaea*, and 18S RNA for *eukarya*), the only way to access the genetic diversity of the viral community from an environmental sample is through metagenomics. Viral metagenomes (also called viromes) should thus provide more and more information about viral diversity and evolution.

## Advantages of Metagenomics in Different Areas

Metagenomics has the potential to advance knowledge in a wide variety of fields. It can also be applied to solve practical challenges in medicine, engineering, agriculture, sustainability and ecology.

### *Agriculture*

As one gram of soil contains around $10^9$-$10^{10}$ microbial cells which comprise about one gigabase of sequence information. They perform a wide variety of ecosystem services necessary for plant growth, including fixing atmospheric nitrogen, nutrient cycling, disease suppression, and sequester iron and other metals. Metagenomic approaches can contribute to improved disease detection in crops and livestock and the adaptation of enhanced farming practices which improve crop health by harnessing the relationship between microbes and plants.

### *Biotechnology*

Recent progress in mining the rich genetic resource of non-culturable microbes has led to the discovery of new genes, enzymes, and natural products. The application of metagenomics has allowed the development of fine chemicals, agrochemicals and pharmaceuticals *etc.*

### *Ecology*

Metagenomics can provide valuable insights into the functional ecology of environmental communities. *eg*. Breaking down of defecations helps to release the nutrients in the faeces into a bioavailable form that can be taken up into the food chain.

### *Environmental remediation*

Metagenomics can improve strategies for monitoring the impact of pollutants on ecosystems and for cleaning up contaminated environments. Increased understanding of how microbial communities cope with pollutants improves assessments of the potential of contaminated sites to recover from pollution and increases the chances of bioaugmentation or biostimulation trials to succeed.

### *Medicine*

Metagenomic sequencing of human microbiome helps to determine the core human microbiome. It also helps to understand the changes in the human microbiome that can be correlated with human health, and to develop new technological and bioinformatics tools to support these goals.

### *Biofuels*

Biofuels are fuels derived from biomass conversion, as in the conversion of cellulose contained in corn stalks, switchgrass, and other biomass into cellulosic ethanol. Metagenomic approaches helps in the analysis of complex microbial communities thus allowing the targeted screening of enzymes with industrial applications in biofuel production, such as glycoside hydrolases.

## Conclusion

Metagenomics has changed the way microbiologists approach many problems, redefined the concept of a genome, and accelerated the rate of gene discovery. The potential for application of metagenomics to human benifit seems endless. Metagenomics gives genetic information on potentially novel biocatalysts or enzymes, genomic linkages between function and phylogeny for uncultured organisms and evolutionary profile of community function and structure. It can also be complemented with metatranscriptomic or metaproteomic approaches to describe expressed activities. Metagenomics is also a powerful tool for generating novel hypotheses of microbial functions, remarkable discoveries of proteorhodopsin-based photoheterotrophy or ammonia-oxidizing Archaea. One of the primary goals of metagenomics projects is to perform a comparative analysis of microbial communities residing in diverse ecological niches. Assessing such differences can not only yield valuable insights into the inherent structure of these microbial communities, but can also identify genes/proteins/organisms that may confer specific functional characteristics to a given environment. Insights gained from such comparative studies are expected to have immense potential in several important areas of biological research, ranging from healthcare (e.g., disease diagnostics, detection of pathogenic contamination and characterization of novel pathogens), industrial biotechnology (bio-prospecting) and bio-remediation studies.

## References

1. Chen, K.; Pachter, L. (2005). "Bioinformatics for Whole-Genome Shotgun Sequencing of Microbial Communities". *PLoS Computational Biology*, 1 (2): e24, doi:10.1371/journal.pcbi.0010024

2. Field D, Amaral-Zettler L, Cochrane G, et al., (2011). The Genomic Standards Consortium: Minimum information about a marker gene sequence (MIMARKS) and minimum information about any (x) sequence (MIxS) specifications. *PLoS Biol*, 9(6):e1001088.

3. Gilbert J.A., Field D., Huang Y., Edwards R., Li W., Glina P. and Joint I. (2008). Detection of large numbers of novel sequences in the metatranscriptomes of complex marine microbial communities. *PLoS ONE,* 3: e3042.

4. Glass EM, Wilkening J, Wilke A, Antonopoulos D, Meyer F, 2010(1). Using the metagenomics RAST server (MG-RAST) for analyzing shotgun metagenomes. *Cold Spring Harb Protocol,* pdb prot5368.

5. Huson DH, Auch AF, Qi J, Schuster SC, (2007). MEGAN analysis of metagenomic data. *Genome Research*, 17(3):377-386.

6. Kristiansson E, Hugenholtz P, Dalevi D, (2009). ShotgunFunctionalizeR, An Rpackage for functional comparison of metagenomes. *Bioinformatics*, 25(20):2737-2738.

7. Markowitz VM, Ivanova NN, *et al.* (2008). IMG/M: a data management and analysis system for metagenomes. *Nucleic Acids Res*, 36 Database: D534-538.

8. Morris R. M., Nunn B. L., Frazar C., Goodlett D. R., Ting Y. S., Rocap G. (2010). Comparative metaproteomics reveals ocean-scale shifts in microbial nutrient utilization and energy transduction. *ISME Journal,* 4: 673–685.

9. Rho M, Tang H, Ye Y, (2010). FragGeneScan: predicting genes in short and error prone reads. *Nucleic Acids Research*, 38(20):e191.

10. Thomas *et al*., (2012). Metagenomics - a guide from sampling to data analysis. *Microbial Informatics and Experimentation* 2:3.

11. Quast, C., Pruesse, E., Yilmaz, P., Gerken, J., Schweer, T., Yarza, P., & Glöckner, F. O. (2012). The SILVA ribosomal RNA gene database project: improved data processing and web-based tools. Nucleic acids research, 41(D1): D590-D596.

12. Z L Sabree, M R Rondon, and J Handelsman, University of Wisconsin-Madison, Madison, WI, USA (2009). Metagenomics. *Elsevier Inc.*

# Statistical Aspects on Analysis of Metagenomics Data

**Sudhir Srivastava and Deepa Bhatt**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

The term "microbiome" is used to describe the community of microorganisms (such as fungi, bacteria and viruses) that exists in a particular environment. The microbiome has been defined as a characteristic microbial community occupying a reasonable well-defined habitat which has distinct physio-chemical properties. The microbiota consists of all living members forming the microbiome. The microbiome encompasses the microorganisms involved as well as their theatre of activity, which results in the formation of specific ecological niches. Plants live in association with diverse microbial consortia. In plants, the microbes live both inside (the endosphere) and outside (the episphere) of plant tissues. The plant microbiome plays roles in plant health and productivity and has received significant attention in recent years.

With the introduction of high-throughput DNA sequencing technologies, there is advancement in microbiome research which enables the study of the genomes of all microbes of a given environment and a precise quantification of microbiome abundances and function. The basic steps of a microbiome study are as follows:

1. Extraction of microbial DNA followed by sequencing: There are two main types of sequencing:
   (i) Amplicon sequencing (reads belong to a fixed gene of each species, most commonly 16S rRNA)
   (ii) Shotgun sequencing (random sequences for the totality of the genetic material are obtained)
2. Sequence processing by using bioinformatics tools
3. Statistical analysis

Amplicon sequencing relies on sequencing a phylogenetic marker gene (e.g. 16S, 18S, ITS). For bacteria and archaea, the marker gene is the 16S ribosomal RNA. There are various bioinformatic pipelines available for processing microbiome 16S sequence data such as mothur, QIIME (Quantitative Insights into Microbial Ecology), BioMaS, etc. Main steps involved in most of the bioinformatics pipelines are given below:

### 1. Preprocessing and quality control

The sequences are assigned to the samples (Demultiplexing). Quality control is performed to remove too short sequences, ambiguous base pairs and chimeras.

**2. Operational taxonomic unit (OTU) binning**

Binning refers to the process of clustering similar DNA sequences into OTUs. Usually, group of DNA sequences should have at least 97% similarity.

**3. Taxonomy assignment**

Taxanomy assisgnment is obtained by comparing OTU consensus sequences to microbial 16S rRNA reference databases such as GreenGenes (http://greengenes.second.genome.com), SILVA (http://www.arbsilva.de), RDP (http://rdp.cme.msu.edu), etc. It provides the available annotation of each OTU to the different taxonomy levels (domain, kingdom, phylum, class, order, family, genus, and species).

**4. Construction of the abundance table**

An OTU abundance table is constructed where each entry in the table corresponds to the number of sequences (reads) observed for each sample corresponding to each OTU. Many OTUs are observed in a few samples. In this situation, it is better to agglomerate OTUs at broader taxonomic groups or taxa.

**5. Phylogenetic analysis**

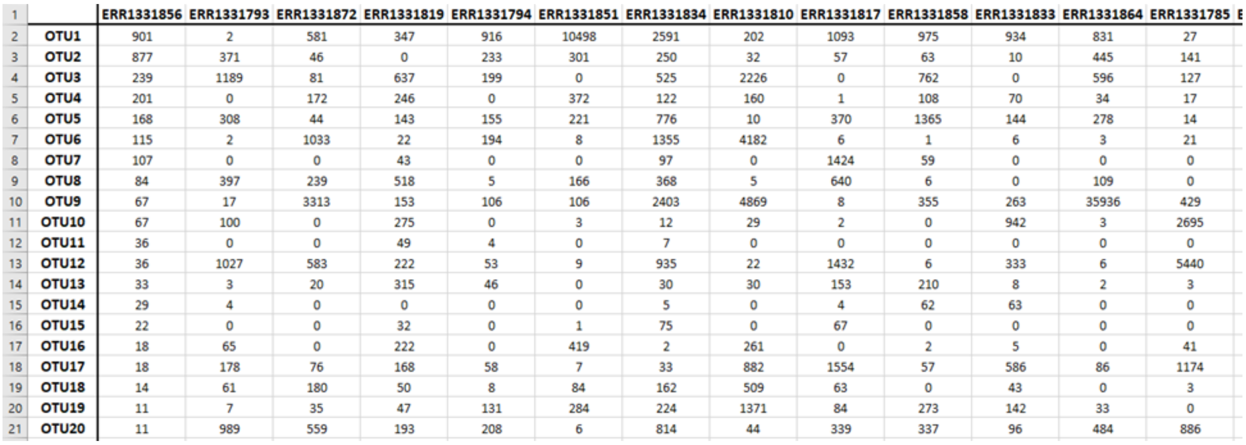
It is the study of evolutionary relatedness among biological groups. Phylogenetic trees are used to obtain phylogenetic distances between samples.

Shotgun metagenomics sequencing involves sequencing the total microbial DNA of a sample. By using this technique, one can

- Infer the relative abundance of each microbial gene.
- Quantify specific metabolic pathways to predict the potential functionality of the entire community – by mapping the obtained sequences against a database [e.g., Kyoto Encyclopedia of Genes and Genomes (KEGG); http://www.genome.jp/kegg/pathway.html]

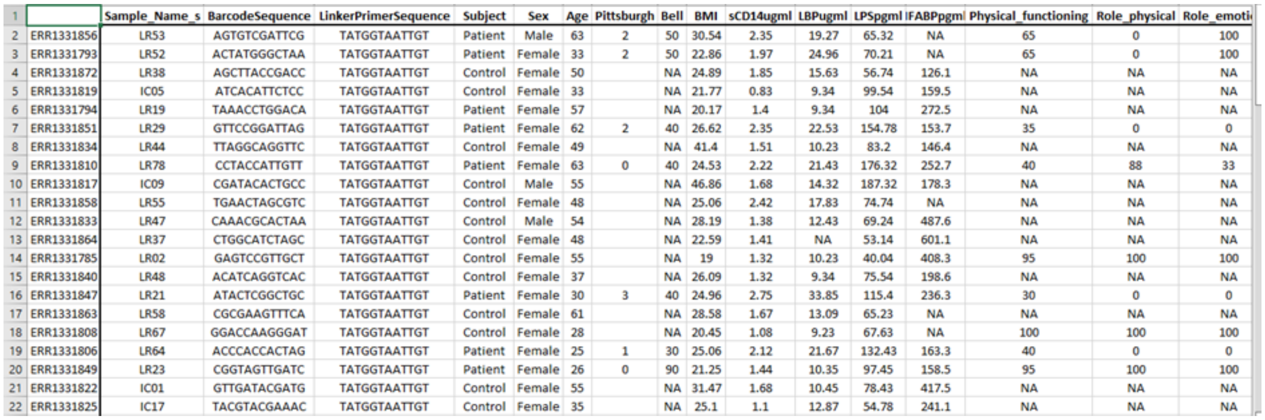
Examples of bioinformatics pipelines for metagenomics analysis: HumanN2, MetaPhlAn 2, SqueezeMeta, etc.

The output (abundance table of counts) of both the approaches (amplicon and shotgun sequencing) is similar. The main element of a microbiome study is the abundance table of counts which represents the number of sequences per sample for a specific taxon. A microbiome abundance table is a matrix of counts, X, with *n* rows (samples) and *k* columns (taxa) where each entry $x_{ij}$ provides the number of sequences (reads) corresponding to taxon *j* in sample *i*. Sometimes, abundance tables are transposed where rows are taxa and columns are samples. In R and Bioconductor packages such as phyloseq, besides abundance table, other elements are also available such as sample data, taxonomy table, phylogenetic tree and DNA String Set (reference sequences).



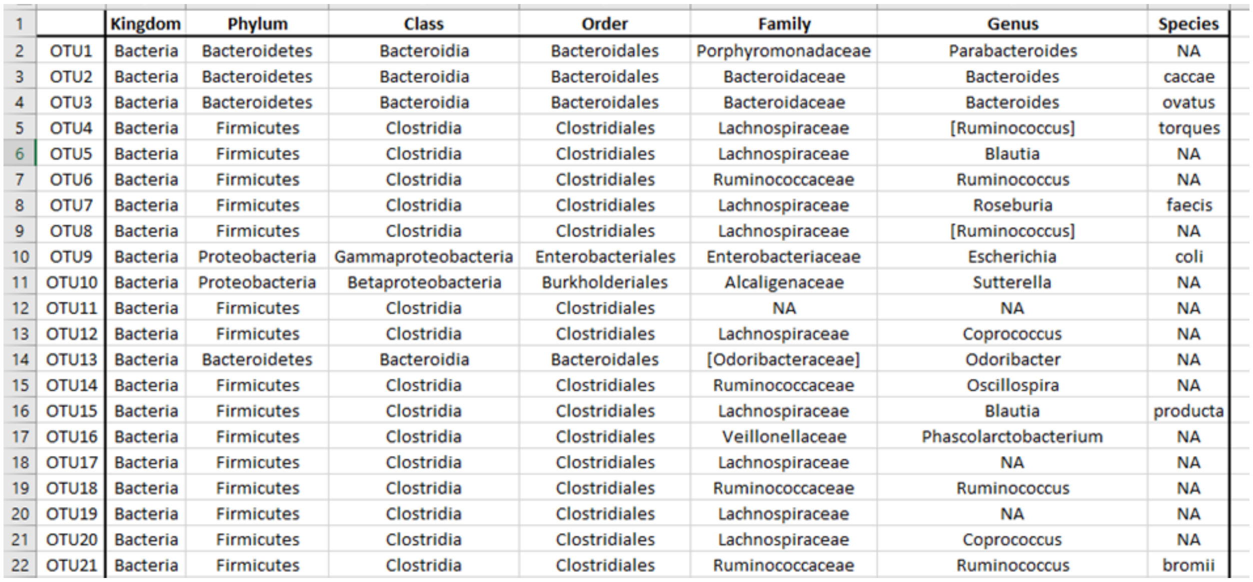
| 1 |  | ERR1331856 | ERR1331793 | ERR1331872 | ERR1331819 | ERR1331794 | ERR1331851 | ERR1331834 | ERR1331810 | ERR1331817 | ERR1331858 | ERR1331833 | ERR1331864 | ERR1331785 | E |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | OTU1 | 901 | 2 | 581 | 347 | 916 | 10498 | 2591 | 202 | 1093 | 975 | 934 | 831 | 27 |  |
| 3 | OTU2 | 877 | 371 | 46 | 0 | 233 | 301 | 250 | 32 | 57 | 63 | 10 | 445 | 141 |  |
| 4 | OTU3 | 239 | 1189 | 81 | 637 | 199 | 0 | 525 | 2226 | 0 | 762 | 0 | 596 | 127 |  |
| 5 | OTU4 | 201 | 0 | 172 | 246 | 0 | 372 | 122 | 160 | 1 | 108 | 70 | 34 | 17 |  |
| 6 | OTU5 | 168 | 308 | 44 | 143 | 155 | 221 | 776 | 10 | 370 | 1365 | 144 | 278 | 14 |  |
| 7 | OTU6 | 115 | 2 | 1033 | 22 | 194 | 8 | 1355 | 4182 | 6 | 1 | 6 | 3 | 21 |  |
| 8 | OTU7 | 107 | 0 | 0 | 43 | 0 | 0 | 97 | 0 | 1424 | 59 | 0 | 0 | 0 |  |
| 9 | OTU8 | 84 | 397 | 239 | 518 | 5 | 166 | 368 | 5 | 640 | 6 | 0 | 109 | 0 |  |
| 10 | OTU9 | 67 | 17 | 3313 | 153 | 106 | 106 | 2403 | 4869 | 8 | 355 | 263 | 35936 | 429 |  |
| 11 | OTU10 | 67 | 100 | 0 | 275 | 0 | 3 | 12 | 29 | 2 | 0 | 942 | 3 | 2695 |  |
| 12 | OTU11 | 36 | 0 | 0 | 49 | 4 | 0 | 7 | 0 | 0 | 0 | 0 | 0 | 0 |  |
| 13 | OTU12 | 36 | 1027 | 583 | 222 | 53 | 9 | 935 | 22 | 1432 | 6 | 333 | 6 | 5440 |  |
| 14 | OTU13 | 33 | 3 | 20 | 315 | 46 | 0 | 30 | 30 | 153 | 210 | 8 | 2 | 3 |  |
| 15 | OTU14 | 29 | 4 | 0 | 0 | 0 | 0 | 5 | 0 | 4 | 62 | 63 | 0 | 0 |  |
| 16 | OTU15 | 22 | 0 | 0 | 32 | 0 | 1 | 75 | 0 | 67 | 0 | 0 | 0 | 0 |  |
| 17 | OTU16 | 18 | 65 | 0 | 222 | 0 | 419 | 2 | 261 | 0 | 2 | 5 | 0 | 41 |  |
| 18 | OTU17 | 18 | 178 | 76 | 168 | 58 | 7 | 33 | 882 | 1554 | 57 | 586 | 86 | 1174 |  |
| 19 | OTU18 | 14 | 61 | 180 | 50 | 8 | 84 | 162 | 509 | 63 | 0 | 43 | 0 | 3 |  |
| 20 | OTU19 | 11 | 7 | 35 | 47 | 131 | 284 | 224 | 1371 | 84 | 273 | 142 | 33 | 0 |  |
| 21 | OTU20 | 11 | 989 | 559 | 193 | 208 | 6 | 814 | 44 | 339 | 337 | 96 | 484 | 886 |  |

Figure 1. Abundance table of counts



| 1 |  | Sample_Name_s | BarcodeSequence | LinkerPrimerSequence | Subject | Sex | Age | Pittsburgh | Bell | BMI | sCD14ugml | LBPugml | LPSpgml | IFABPpgml | Physical_functioning | Role_physical | Role_emoti |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | ERR1331856 | LR53 | AGTGTCGATTCG | TATGGTAATTGT | Patient | Male | 63 | 2 | 50 | 30.54 | 2.35 | 19.27 | 65.32 | NA | 65 | 0 | 100 |
| 3 | ERR1331793 | LR52 | ACTATGGGCTAA | TATGGTAATTGT | Patient | Female | 33 | 2 | 50 | 22.86 | 1.97 | 24.96 | 70.21 | NA | 65 | 0 | 100 |
| 4 | ERR1331872 | LR38 | AGCTTACCGACC | TATGGTAATTGT | Control | Female | 50 |  | NA | 24.89 | 1.85 | 15.63 | 56.74 | 126.1 | NA | NA | NA |
| 5 | ERR1331819 | IC05 | ATCACATTCTCC | TATGGTAATTGT | Control | Female | 33 |  | NA | 21.77 | 0.83 | 9.34 | 99.54 | 159.5 | NA | NA | NA |
| 6 | ERR1331794 | LR19 | TAAACCTGGACA | TATGGTAATTGT | Patient | Female | 57 |  | NA | 20.17 | 1.4 | 9.34 | 104 | 272.5 | NA | NA | NA |
| 7 | ERR1331851 | LR29 | GTTCCGGATTAG | TATGGTAATTGT | Patient | Female | 62 | 2 | 40 | 26.62 | 2.35 | 22.53 | 154.78 | 153.7 | 35 | 0 | 0 |
| 8 | ERR1331834 | LR44 | TTAGGCAGGTTC | TATGGTAATTGT | Control | Female | 49 |  | NA | 41.4 | 1.51 | 10.23 | 83.2 | 146.4 | NA | NA | NA |
| 9 | ERR1331810 | LR78 | CCTACCATTGTT | TATGGTAATTGT | Patient | Female | 63 | 0 | 40 | 24.53 | 2.22 | 21.43 | 176.32 | 252.7 | 40 | 88 | 33 |
| 10 | ERR1331817 | IC09 | CGATACACTGCC | TATGGTAATTGT | Control | Male | 55 |  | NA | 46.86 | 1.68 | 14.32 | 187.32 | 178.3 | NA | NA | NA |
| 11 | ERR1331858 | LR55 | TGAACTAGCGTC | TATGGTAATTGT | Control | Female | 48 |  | NA | 25.06 | 2.42 | 17.83 | 74.74 | NA | NA | NA | NA |
| 12 | ERR1331833 | LR47 | CAAACGCACTAA | TATGGTAATTGT | Control | Male | 54 |  | NA | 28.19 | 1.38 | 12.43 | 69.24 | 487.6 | NA | NA | NA |
| 13 | ERR1331864 | LR37 | CTGGCATCTAGC | TATGGTAATTGT | Control | Female | 48 |  | NA | 22.59 | 1.41 | NA | 53.14 | 601.1 | NA | NA | NA |
| 14 | ERR1331785 | LR02 | GAGTCCGTTGCT | TATGGTAATTGT | Control | Female | 55 |  | NA | 19 | 1.32 | 10.23 | 40.04 | 408.3 | 95 | 100 | 100 |
| 15 | ERR1331840 | LR48 | ACATCAGGTCAC | TATGGTAATTGT | Control | Female | 37 |  | NA | 26.09 | 1.32 | 9.34 | 75.54 | 198.6 | NA | NA | NA |
| 16 | ERR1331847 | LR21 | ATACTCGGCTGC | TATGGTAATTGT | Patient | Female | 30 | 3 | 40 | 24.96 | 2.75 | 33.85 | 115.4 | 236.3 | 30 | 0 | 0 |
| 17 | ERR1331863 | LR58 | CGCGAAGTTTCA | TATGGTAATTGT | Control | Female | 61 |  | NA | 28.58 | 1.67 | 13.09 | 65.23 | NA | NA | NA | NA |
| 18 | ERR1331808 | LR67 | GGACCAAGGGAT | TATGGTAATTGT | Control | Female | 28 |  | NA | 20.45 | 1.08 | 9.23 | 67.63 | NA | 100 | 100 | 100 |
| 19 | ERR1331806 | LR64 | ACCCACCACTAG | TATGGTAATTGT | Patient | Female | 25 | 1 | 30 | 25.06 | 2.12 | 21.67 | 132.43 | 163.3 | 40 | 0 | 0 |
| 20 | ERR1331849 | LR23 | CGGTAGTTGATC | TATGGTAATTGT | Patient | Female | 26 | 0 | 90 | 21.25 | 1.44 | 10.35 | 97.45 | 158.5 | 95 | 100 | 100 |
| 21 | ERR1331822 | IC01 | GTTGATACGATG | TATGGTAATTGT | Control | Female | 55 |  | NA | 31.47 | 1.68 | 10.45 | 78.43 | 417.5 | NA | NA | NA |
| 22 | ERR1331825 | IC17 | TACGTACGAAAC | TATGGTAATTGT | Control | Female | 35 |  | NA | 25.1 | 1.1 | 12.87 | 54.78 | 241.1 | NA | NA | NA |

Figure 2. Sample data



| 1 |  | Kingdom | Phylum | Class | Order | Family | Genus | Species |
|---|---|---|---|---|---|---|---|---|
| 2 | OTU1 | Bacteria | Bacteroidetes | Bacteroidia | Bacteroidales | Porphyromonadaceae | Parabacteroides | NA |
| 3 | OTU2 | Bacteria | Bacteroidetes | Bacteroidia | Bacteroidales | Bacteroidaceae | Bacteroides | caccae |
| 4 | OTU3 | Bacteria | Bacteroidetes | Bacteroidia | Bacteroidales | Bacteroidaceae | Bacteroides | ovatus |
| 5 | OTU4 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | [Ruminococcus] | torques |
| 6 | OTU5 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | Blautia | NA |
| 7 | OTU6 | Bacteria | Firmicutes | Clostridia | Clostridiales | Ruminococcaceae | Ruminococcus | NA |
| 8 | OTU7 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | Roseburia | faecis |
| 9 | OTU8 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | [Ruminococcus] | NA |
| 10 | OTU9 | Bacteria | Proteobacteria | Gammaproteobacteria | Enterobacteriales | Enterobacteriaceae | Escherichia | coli |
| 11 | OTU10 | Bacteria | Proteobacteria | Betaproteobacteria | Burkholderiales | Alcaligenaceae | Sutterella | NA |
| 12 | OTU11 | Bacteria | Firmicutes | Clostridia | Clostridiales | NA | NA | NA |
| 13 | OTU12 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | Coprococcus | NA |
| 14 | OTU13 | Bacteria | Bacteroidetes | Bacteroidia | Bacteroidales | [Odoribacteraceae] | Odoribacter | NA |
| 15 | OTU14 | Bacteria | Firmicutes | Clostridia | Clostridiales | Ruminococcaceae | Oscillospira | NA |
| 16 | OTU15 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | Blautia | producta |
| 17 | OTU16 | Bacteria | Firmicutes | Clostridia | Clostridiales | Veillonellaceae | Phascolarctobacterium | NA |
| 18 | OTU17 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | NA | NA |
| 19 | OTU18 | Bacteria | Firmicutes | Clostridia | Clostridiales | Ruminococcaceae | Ruminococcus | NA |
| 20 | OTU19 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | NA | NA |
| 21 | OTU20 | Bacteria | Firmicutes | Clostridia | Clostridiales | Lachnospiraceae | Coprococcus | NA |
| 22 | OTU21 | Bacteria | Firmicutes | Clostridia | Clostridiales | Ruminococcaceae | Ruminococcus | bromii |

Figure 3. Taxonomy table

## Statistical Analysis of Microbiome Data

A microbiome statistical analysis consists of the following major steps:

- Normalization
- Diversity analysis
- Ordination
- Differential abundance testing

The statistical analysis of microbiome abundance data starts with the normalization of the data followed by an exploratory study of the microbiome composition for the identification of possible data structures. The exploratory part consists of the analysis of diversity measures and their visualization through ordination plots. There are many challenges involved in the analysis of microbiome count data. One of the challenges is related to count data analysis which involves skewed distribution, zero inflation and over-dispersion.

## Normalization

The microbiome data is very noisy due variations caused during the execution of experiment and preprocessing steps such as quality control filtering. The total number of counts per sample is highly variable which may arise due to biological and technical issues. Therefore, some normalization is required prior to the analysis so that the microbiome abundances among the different samples are comparable. Abundance tables are usually sparse since many species are infrequent. Further, there is much redundant information because of co-abundance of many species. Various approaches of normalization are as follows:

- Computation of relative abundances: The simplest way is the computation of relative abundances by dividing the raw abundances by the total number of counts per sample.
- Rarefaction: It consists of subsampling the same number of reads for each sample so that all samples have the same number of total counts. However, this method is not recommended as it entails loss of important information and precision of measurement is decreased. Further, the random choice of reads decreases repeatability of experiment and adds bias.
- Sophisticated techniques implemented in some R packages for RNA-seq data analysis such as DESeq2 and edgeR:
  - TMM (Trimmed Mean of M-values)
  - TMMwsp (TMM with singleton pairing)
  - RLE (relative log expression)

Compositional Data Analysis (CoDA) techniques such as log-ratio approach can be used as an alternative because these do not require the normalization step.

Examples:

- Additive log-ratio transformation (alr)
- Centered log-ratio transformation (clr)
- Isometric log-ratio transformation (ilr)

Microbiome abundance tables are sparse and contain many zeros. This should be properly addressed before CoDA methods can be applied. One of the simplest approaches is to replace zeros by a small pseudo-count or to add a small constant (e.g. 1) to all the elements of the abundance matrix.

### Diversity Analysis

Microbiome diversity can be measured through multiple ecological indices. There are basically two kind of measures:

- Alpha diversity (within sample variability)
- Beta diversity (between samples variability)

### Alpha diversity (within sample variability)

The simplest measure of alpha diversity is richness. Richness is estimated by the observed richness, $R_{obs}$, the number of different species observed in the sample. The observed richness tends to underestimate the real richness in the environment, where the less frequent species are likely to be undetected. There are different indices that adjust for less frequent or undetected species.

$$\text{Chao1 index, } R_{Chao1} = R_{obs} + \frac{f_1(f_1-1)}{2(f_2+1)}$$

where $f_1$ is the number of species observed only once and $f_2$ is the number of species observed twice.

Another important measure of alpha diversity is evenness which measures the homogeneity in abundance of different species in a sample. Most commonly used measure of evenness is the Shannon index:

$$R_{Shannon} = -\sum_{i=1}^{k} p_i \log(p_i)$$

where $p_i$ represents the relative abundances of the $i^{\text{th}}$ taxon.

### Beta diversity (between samples variability)

It measures the differences in microbiome composition between samples. It provides a measure of similarity, or dissimilarity, of one microbial composition to another. There is a wide range of ecological distances or dissimilarities for measuring beta diversity such as Bray-Curtis, UniFrac, weighted UniFrac distances, Aitchison distance, etc.

The R package “vegan” provides a large set of diversity measures.

Let $p_1 = (p_{11},\ p_{12}, \dots, p_{1i})$ an $p_2 = (p_{21},\ p_{22}, \dots, p_{2i})$ denote the microbiome relative abundance of two different samples. Bray-Curtis is defined as

$$d_{BC}(p_1, p_2) = \frac{\sum_{i=1}^{k} |p_{1i} - p_{2i}|}{\sum_{i=1}^{k} (p_{1i} + p_{2i})}$$

Consider a phylogenetic tree with *r* branches. Let $b = (b_1,\ b_2, \dots, b_r)$ denotes the length of the different branches in the phylogenetic tree. Let $q_1 = (q_{11},\ q_{12}, \dots, q_{1r})$ and $q_2 = (q_{21},\ q_{22}, \dots, q_{2r})$ denote the relative abundances associated to each branch for the first and the second sample, respectively. The unweighted UniFrac distance is defined as

$$d_v(b, q_1, q_2) = \frac{\sum_{i=1}^{r} b_i |I(q_{1i} > 0) - I(q_{2i} > 0)|}{\sum_{i=1}^{r} b_i I(q_{1i} + q_{2i} > 0)}$$

The weighted UniFrac distance is defined as

$$d_W(b, q_1, q_2) = \frac{\sum_{i=1}^{r} b_i |q_{1i} - q_{2i}|}{\sum_{i=1}^{r} (q_{1i} + q_{2i}) I(q_{1i} + q_{2i} > 0)}$$

Given two compositions $x_1$ and $x_2$, the Aitchison distance is defined as

$$d_A(x_1, x_2) = d_E\big(clr(x_1), clr(x_2)\big)$$

where $d_E$ denotes Euclidean distance.

## Ordination

The purpose of ordination plots is to visualize beta diversity for identification of possible data structures. The multidimensional data is represented into a reduced number of orthogonal axes while keeping the main trends of the data and preserving the distances among samples as much as possible. Two most commonly used ordination methods for microbiome data are

- Principal coordinates analysis (PCoA) or multidimensional scaling (MDS)
- Non-metric multidimensional scaling (NMDS)

PCoA is an extension of Principal Components Analysis (PCA). PCoA results exactly the same as PCA. In PCoA, some eigenvalues may be negative and the graphical representation will not perform properly. Therefore, in such case, NMDS is more commonly used. It maximizes the rank-based correlation between the original distances and the distances between samples in the new reduced ordination space. Ordination plots can be obtained using R and Bioconductor packages such as vegan, phyloseq, etc.

**Differential abundance testing**

An inference analysis is performed where microbiome composition is tested for association with a variable of interest. Differential abundance testing is usually done when the outcome of interest is dichotomous (e.g., healthy and diseased). These association tests can be:

1. Univariate - aim is to identify which taxa are differentially abundant between sample groups

2. Multivariate - assess for global differences in microbial composition between sample groups

**1. Univariate differential abundance testing:** Every taxa is separately tested for association with the response variable. Various methods for univariate abundance testing are given below.

**(i) Nonparametric tests**, e.g., Wilcoxon rank-sum test or Kruskal-Wallis test

**(ii) Parametric approaches**

- Available in the Bioconductor packages such as edgeR and DESeq2, initially proposed for RNA-Seq data analysis can be used.
- Both fit a generalized linear model and assume that read counts follow a Negative Binomial distribution.

CoDA methods such as ANCOM and ALDEx2 can be applied.

- ANCOM - the log-ratio of all pairs of variables is tested for differences in means.
- ALDEx2 algorithm
  - ✓ It uses a Dirichlet-multinomial model to infer the multivariate abundance distribution from counts.
  - ✓ After clr transformation, it performs the Wilcoxon rank test (two groups) or Kruskal-Wallis tests (more than two groups).

**2. Multivariate differential abundance testing**

It refers to a global test of differences in microbial composition between two or more groups of samples. Some of the methods for multivariate differential abundance testing are given below:

(i) Permutational Multivariate Analysis of Variance Using Distance Matrices (PERMANOVA)

- The null hypothesis of no differences in composition among groups is formulated by the condition that the different groups of samples have the same center of masses.
- Implemented in function “adonis” of R package “vegan”.
- Consists of a multivariate ANOVA based on dissimilarities.
- Significance is evaluated through permutations to generate a distribution of pseudo F statistic under the null hypothesis.

(ii) A popular distance-based approach is the analysis of similarities implemented in the function “anosim” of R package “vegan”.

(iii) Kernel machine regression (KMR)

- A model-based approach for multivariate microbiome analysis that extends PERMANOVA to a regression framework.
- A semi-parametric regression model that includes a nonparametric component.

(iv) Model-based methods for hypothesis testing, power and sample size calculations based on Dirichlet-Multinomial distribution:

- Proposed by La Rosa et al.
- The methods are implemented in the R package "HMP".

(v) Multivariate statistical framework mixMC

- Proposed by Le Cao et al. where sparse partial least squares discriminant analysis (sPLS-DA) is performed.
- The proposed method has been implemented in the R package "mixOmics".

## References

1. Berg, G., Rybakova, D., Fischer, D., Cernava, T., Vergès, M. C., Charles, T., Chen, X., Cocolin, L., Eversole, K., Corral, G. H., Kazou, M., Kinkel, L., Lange, L., Lima, N., Loy, A., Macklin, J. A., Maguin, E., Mauchline, T., McClure, R., Mitter, B., … Schloter, M. (2020). Microbiome definition re-visited: old concepts and new challenges. Microbiome, 8(1), 103. https://doi.org/10.1186/s40168-020-00875-0
2. Calle M. L. (2019). Statistical Analysis of Metagenomics Data. Genomics & informatics, 17(1), e6. https://doi.org/10.5808/GI.2019.17.1.e6
3. Caporaso, J. G., Kuczynski, J., Stombaugh, J., Bittinger, K., Bushman, F. D., Costello, E. K., Fierer, N., Peña, A. G., Goodrich, J. K., Gordon, J. I., Huttley, G. A., Kelley, S. T., Knights, D., Koenig, J. E., Ley, R. E., Lozupone, C. A., McDonald, D., Muegge, B. D., Pirrung, M., Reeder, J., … Knight, R. (2010). QIIME allows analysis of high-throughput community sequencing data. Nature methods, 7(5), 335–336. https://doi.org/10.1038/nmeth.f.303
4. Khondoker M.G. Dastogeer, Farzana Haque Tumpa, Afruja Sultana, Mst Arjina Akter, Anindita Chakraborty (2020). Plant microbiome–an account of the factors that shape community composition and diversity, Current Plant Biology, 23, 100161, ISSN 2214-6628. https://doi.org/10.1016/j.cpb.2020.100161.
5. La Rosa, P. S., Brooks, J. P., Deych, E., Boone, E. L., Edwards, D. J., Wang, Q., Sodergren, E., Weinstock, G., & Shannon, W. D. (2012). Hypothesis testing and power calculations for taxonomic-based human microbiome data. PloS one, 7(12), e52078. https://doi.org/10.1371/journal.pone.0052078

6. Lê Cao, K. A., Boitard, S., & Besse, P. (2011). Sparse PLS discriminant analysis: biologically relevant feature selection and graphical displays for multiclass problems. BMC bioinformatics, 12, 253. https://doi.org/10.1186/1471-2105-12- 253
7. McMurdie, P. J., & Holmes, S. (2013). phyloseq: an R package for reproducible interactive analysis and graphics of microbiome census data. PloS one, 8(4), e61217. https://doi.org/10.1371/journal.pone.0061217
8. Odintsova, V., Tyakht, A., & Alexeev, D. (2017). Guidelines to Statistical Analysis of Microbial Composition Data Inferred from Metagenomic Sequencing. Current issues in molecular biology, 24, 17–36. https://doi.org/10.21775/cimb.024.017
9. Schloss, P. D., Westcott, S. L., Ryabin, T., Hall, J. R., Hartmann, M., Hollister, E. B., Lesniewski, R. A., Oakley, B. B., Parks, D. H., Robinson, C. J., Sahl, J. W., Stres, B., Thallinger, G. G., Van Horn, D. J., & Weber, C. F. (2009). Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. Applied and environmental microbiology, 75(23), 7537–7541. https://doi.org/10.1128/AEM.01541-09

# Metagenomics Data Analysis using QIIME 2

**Anu Sharma**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## 1. Introduction

QIIME 2 is a completely reengineered microbiome bioinformatics platform based on the popular QIIME platform, which it has replaced. QIIME 2 facilitates comprehensive and fully reproducible microbiome data science, improving accessibility to diverse users by adding multiple user interfaces.

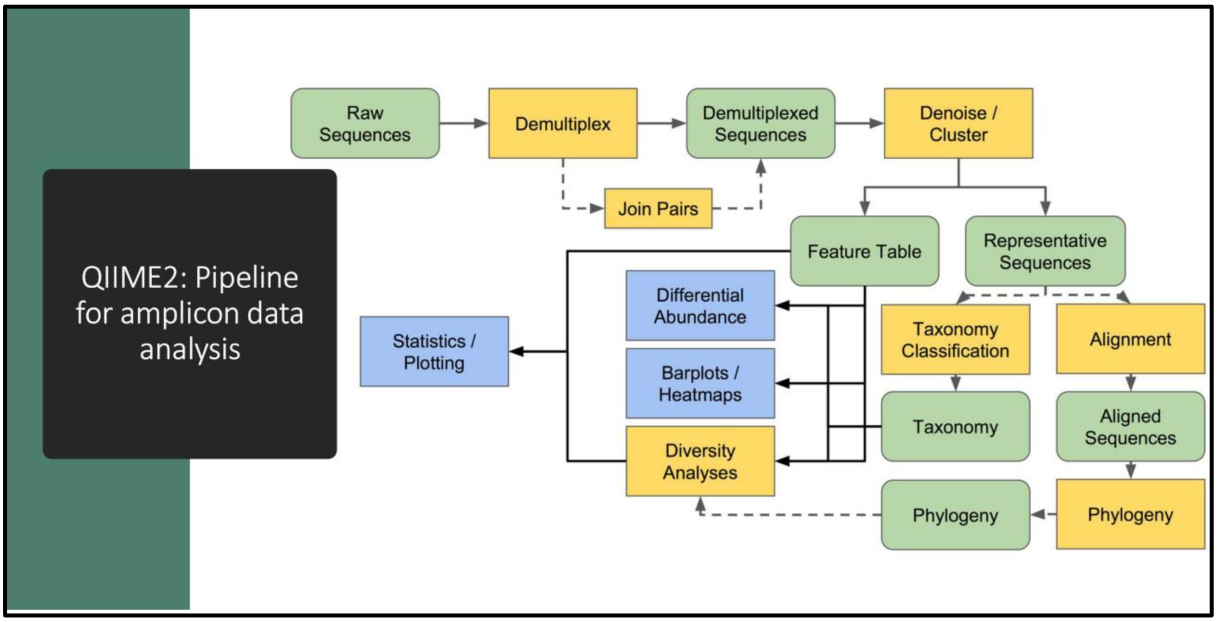


Fig. 1: Pipeline for amplicon data analysis

**Key features:**

- Integrated and automatic tracking of data provenance
- Semantic type system
- Plugin system for extending microbiome analysis functionality
- Support for multiple types of user interfaces (e.g. API, command line, graphical)

## 2. Data files: QIIME 2 artifacts

Data produced by QIIME 2 exist as QIIME 2 artifacts. A QIIME 2 artifact contains data and metadata. The metadata describes things about the data, such as its type, format, and how it was generated (provenance). A QIIME 2 artifact typically has the .qza file extension when stored in a file.

Since QIIME 2 works with artifacts instead of data files (e.g. FASTA files), data can be imported at any step in an analysis, though typically it start by importing raw sequence data. QIIME 2 also has tools to export data from an artifact. By using QIIME 2 artifacts instead of simple data files, QIIME 2 can automatically track the type, format, and provenance of data for researchers. Using artifacts instead of data files enables researchers to focus on the analyses they want to perform, instead of the particular format the data needs to be in for an analysis.

### 2.1 Data files: visualizations

Visualizations are another type of data generated by QIIME 2. When written to disk, visualization files typically have the .qzv file extension. Visualizations contain similar types of metadata as QIIME 2 artifacts, including provenance information. Similar to QIIME 2 artifacts, visualizations are standalone information that can be archived or shared with collaborators.

In contrast to QIIME 2 artifacts, visualizations are terminal outputs of an analysis, and can represent, for example, a statistical results table, an interactive visualization, static images, or really any combination of visual data representations. Since visualizations are terminal outputs, they cannot be used as input to other analyses in QIIME 2.

### 2.2 Semantic types

Every artifact generated by QIIME 2 has a semantic type associated with it. Semantic types enable QIIME 2 to identify artifacts that are suitable inputs to an analysis. For example, if an analysis expects a distance matrix as input, QIIME 2 can determine which artifacts have a distance matrix semantic type and prevent incompatible artifacts from being used in the analysis (e.g. an artifact representing a phylogenetic tree). Semantic types also help users avoid semantically incorrect analyses. For example, a feature table could contain presence/absence data (i.e., a 1 to indicate that an OTU was observed at least one time in a given sample, and a 0 to indicate than an OTU was not observed at least one time in a given sample). However, if that feature table were provided to an analysis computing a quantitative diversity metric where OTU abundances are included in the calculation (e.g., weighted UniFrac), the analysis would complete successfully, but the result would not be meaningful.

This guide assumes that QIIME 2 have been installed using one of the procedures in the install documents at https://docs.qiime2.org/2022.8/install/.

## 3. Obtaining and importing data

```
wget \
  -O 'emp-single-end-sequences.zip' \
  'https://docs.qiime2.org/2021.11/data/tutorials/moving-pictures-usage/emp
-single-end-sequences.zip'

unzip -d emp-single-end-sequences emp-single-end-sequences.zip
```

```
qiime tools import \
  --type 'EMPSingleEndSequences' \
  --input-path emp-single-end-sequences \
  --output-path emp-single-end-sequences.qza
```

## 4. Demultiplexing sequences

To demultiplex sequences we need to know which barcode sequence is associated with each sample. This information is contained in the sample metadata file. You can run the following commands to demultiplex the sequences (the demux emp-single command refers to the fact that these sequences are barcoded according to the Earth Microbiome Project protocol, and are single-end reads). The demux.qza QIIME 2 artifact will contain the demultiplexed sequences.

```
qiime demux emp-single \
  --i-seqs emp-single-end-sequences.qza \
  --m-barcodes-file sample-metadata.tsv \
  --m-barcodes-column barcode-sequence \
  --o-per-sample-sequences demux.qza \
  --o-error-correction-details demux-details.qza
```

After demultiplexing, it's useful to generate a summary of the demultiplexing results. This allows you to determine how many sequences were obtained per sample, and also to get a summary of the distribution of sequence qualities at each position in your sequence data.

```
qiime demux summarize \
  --i-data demux.qza \
  --o-visualization demux.qzv
```

### 5. Sequence quality control and feature table construction

QIIME 2 plugins are available for several quality control methods, including DADA2, Deblur, and basic quality-score-based filtering. In this tutorial we present this step using DADA2. These steps are interchangeable, so you can use whichever of these you prefer. The result of both of these methods will be a FeatureTable[Frequency] QIIME 2 artifact, which contains counts (frequencies) of each unique sequence in each sample in the dataset, and a FeatureData[Sequence] QIIME 2 artifact, which maps feature identifiers in the FeatureTable to the sequences they represent.

```
qiime dada2 denoise-single \
  --i-demultiplexed-seqs demux.qza \
  --p-trim-left 0 \
  --p-trunc-len 120 \
  --o-representative-sequences rep-seqs.qza \
  --o-table table.qza \
  --o-denoising-stats stats.qza

qiime metadata tabulate \
  --m-input-file stats.qza \
  --o-visualization stats.qzv
```

### 6. FeatureTable and FeatureData summaries

```
qiime feature-table summarize \
  --i-table table.qza \
  --m-sample-metadata-file sample-metadata.tsv \
  --o-visualization table.qzv
qiime feature-table tabulate-seqs \
  --i-data rep-seqs.qza \
  --o-visualization rep-seqs.qzv
```

### 7. Generate a tree for phylogenetic diversity analyses

```
qiime phylogeny align-to-tree-mafft-fasttree \
  --i-sequences rep-seqs.qza \
  --output-dir phylogeny-align-to-tree-mafft-fasttree
```

### 8. Alpha and beta diversity analysis

```
qiime diversity core-metrics-phylogenetic \
  --i-phylogeny phylogeny-align-to-tree-mafft-fasttree/rooted_tree.qza
\
  --i-table table.qza \
  --p-sampling-depth 1103 \
  --m-metadata-file sample-metadata.tsv \
  --output-dir diversity-core-metrics-phylogenetic
```

### 9. Taxonomic analysis

```
wget \
  -O 'gg-13-8-99-515-806-nb-classifier.qza' \
  'https://docs.qiime2.org/2021.11/data/tutorials/moving-pictures-usage
/gg-13-8-99-515-806-nb-classifier.qza'

qiime feature-classifier classify-sklearn \
  --i-classifier gg-13-8-99-515-806-nb-classifier.qza \
  --i-reads rep-seqs.qza \
  --o-classification taxonomy.qza

qiime metadata tabulate \
  --m-input-file taxonomy.qza \
  --o-visualization taxonomy.qzv

qiime taxa barplot \
  --i-table table.qza \
  --i-taxonomy taxonomy.qza \
  --m-metadata-file sample-metadata.tsv \
  --o-visualization taxa-bar-plots.qzv
```

**References:**

1. https://docs.qiime2.org/2022.8/tutorials/moving-pictures-usage/
2. https://docs.qiime2.org/2022.8/concepts/#data-files-qiime-2-artifacts
3. Mehrbod Estaki,Lingjing Jiang,Nicholas A. Bokulich,Daniel McDonald,Antonio González,Tomasz Kosciolek,Cameron Martino,Qiyun Zhu,Amanda Birmingham,Yoshiki Vázquez-Baeza,Matthew R. Dillon,Evan Bolyen,J. Gregory Caporaso,Rob Knight (2020). QIIME 2 Enables Comprehensive End-to-End Analysis of Diverse Microbiome Data and Comparative Studies with Publicly Available Data. Current Protocols in Bioinformatics. *Current Protocols in Bioinformaticse100, Volume 70,* Published in Wiley Online Library (wileyonlinelibrary.com).doi: 10.1002/cpbi.100

# Statistical Analysis of Metagenomics Data

**Ritwika Das**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## ❑ Statistical Analysis of Metagenomic Profiles

Taxonomic and functional differences between metagenomic samples can highlight the influence of ecological factors on patterns of microbial life in a wide range of habitats. Statistical hypothesis tests help to distinguish ecological influences from sampling artifacts, but knowledge of only the p-value is insufficient to make inferences about biological relevance. Biological relevance of a feature requires consideration of effect sizes and their associated confidence intervals. Interpretation of statistical results can also benefit from transforming raw p-values to superior interpretations and by allowing interactive filtering that permits focusing on features with specific statistical properties.

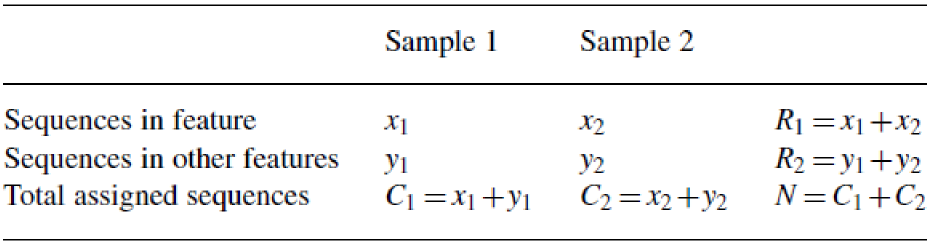
p-value indicates the probability of an observed difference occurring simply by chance. Features in a profile with p-values below 0.05 are termed as statistically significant and can reasonably be assumed to be enriched in one of the metagenomes due to ecological or taxonomic differences as opposed to being the result of a sampling artifact. Fisher's exact test uses hypergeometric distribution to efficiently calculate the exact p-value without the requirement of all possible permutation of sequences in a pair of metagenomic samples. The chi-square test and G-test are well-known large sample approximations to Fisher's exact test. Barnard's test is computationally prohibitive for the majority of features in a typical metagenomic profile. So, we need to decide between an approximation to Barnard's exact test (*e.g.*, bootstrapping) and Fisher's exact test.

A typical metagenomic profile consists of several hundred features. When performing multiple hypothesis tests, it is useful to modify the p-values so that they reflect a particular interpretation. If we wish to examine a list of features where the probability of observing one or more false positive is less than a specified probability, we can use a correction method. Commonly applied correction methods include Bonferroni, Holm-Bonferroni and Šidák (Abdi, 2007). Alternatively, during exploratory analysis, we may be willing to accept a specific percentage of false positives. This can be achieved using the Benjamini–Hochberg false discovery rate (FDR) procedure (Benjamini and Hochberg, 1995) or the Storey FDR approach (Storey and Tibshirani, 2003). These approaches complement each other while performing an exploratory analysis. The list of significant features obtained without any multiple test correction method gives us an initial global look at those features which may be differentially abundant between our samples. An FDR approach can be used to refine this initial list and to make the number of expected false positives explicit. Finally, a correction technique can be applied to focus our attention to only those features where the observed enrichment or depletion is highly unlikely to be a sampling artifact.

## ❑ Effect Size and Confidence Intervals

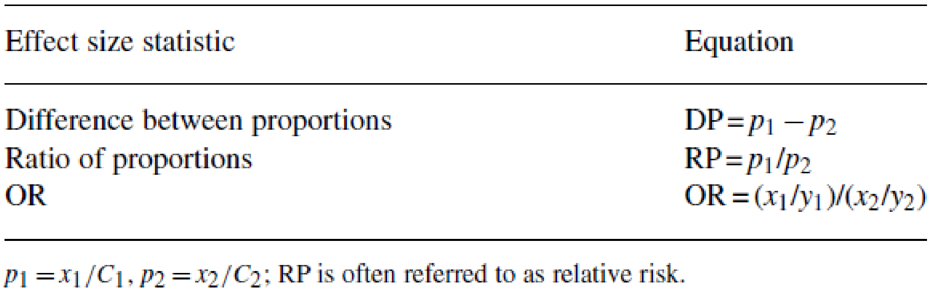
To assess if a feature is of biological relevance, we should consider the magnitude of the observed difference (*i.e.*, an effect size statistic). An arbitrarily small effect can be statistically significant if the sample sizes are sufficiently large. So, biological significance of a feature must be supported by effect size statistics as well as p-values.

**Table 1: Contingency table summarising data for a feature of interest**



| | Sample 1 | Sample 2 | |
|---|---|---|---|
| Sequences in feature | $x_1$ | $x_2$ | $R_1 = x_1 + x_2$ |
| Sequences in other features | $y_1$ | $y_2$ | $R_2 = y_1 + y_2$ |
| Total assigned sequences | $C_1 = x_1 + y_1$ | $C_2 = x_2 + y_2$ | $N = C_1 + C_2$ |

**Table 2: Effect size statistics of a feature of interest**



| Effect size statistic | Equation |
|---|---|
| Difference between proportions | $\text{DP} = p_1 - p_2$ |
| Ratio of proportions | $\text{RP} = p_1 / p_2$ |
| OR | $\text{OR} = (x_1/y_1)/(x_2/y_2)$ |

$p_1 = x_1/C_1$, $p_2 = x_2/C_2$; RP is often referred to as relative risk.

The most intuitive effect size statistic is the difference between proportions (DP) of sequences assigned to a given feature in the two samples. Ratio of proportions (RP) is also a measure that provides complementary information to the DP. Consideration of multiple effect size statistics is often essential while assessing biological relevance as features can have a small (or, large) DP, but a large (or, small) RP. The odds ratio (OR) has many desirable mathematical properties. However, RP is preferred over OR due to the difficulty in interpretation of the latter.

Confidence interval (CI) indicates the range of effect size values that have a specified probability of being compatible with the observed data. A 95% CI gives a lower and upper bound in which the true effect size will be contained 19 times out of 20. There is a close relationship between p-values and CI. CI that encompasses the identity effect size (*e.g.*, DP = 0 or RP = OR = 1) will have a p-value > (1 – the coverage of the CI) (*i.e.*, a p-value ≥ 0.05 for a 95% CI). If the identity effect size is outside the CI, the p-value will be ≤ 0.05 for a 95% CI. Critically, CI provides a mean to infer the biological relevance of a feature even when it is marginally statistically significant.

❑ **Software: STAMP (Parks *et al.*, 2010)**

❑ **Concept of STAMP**

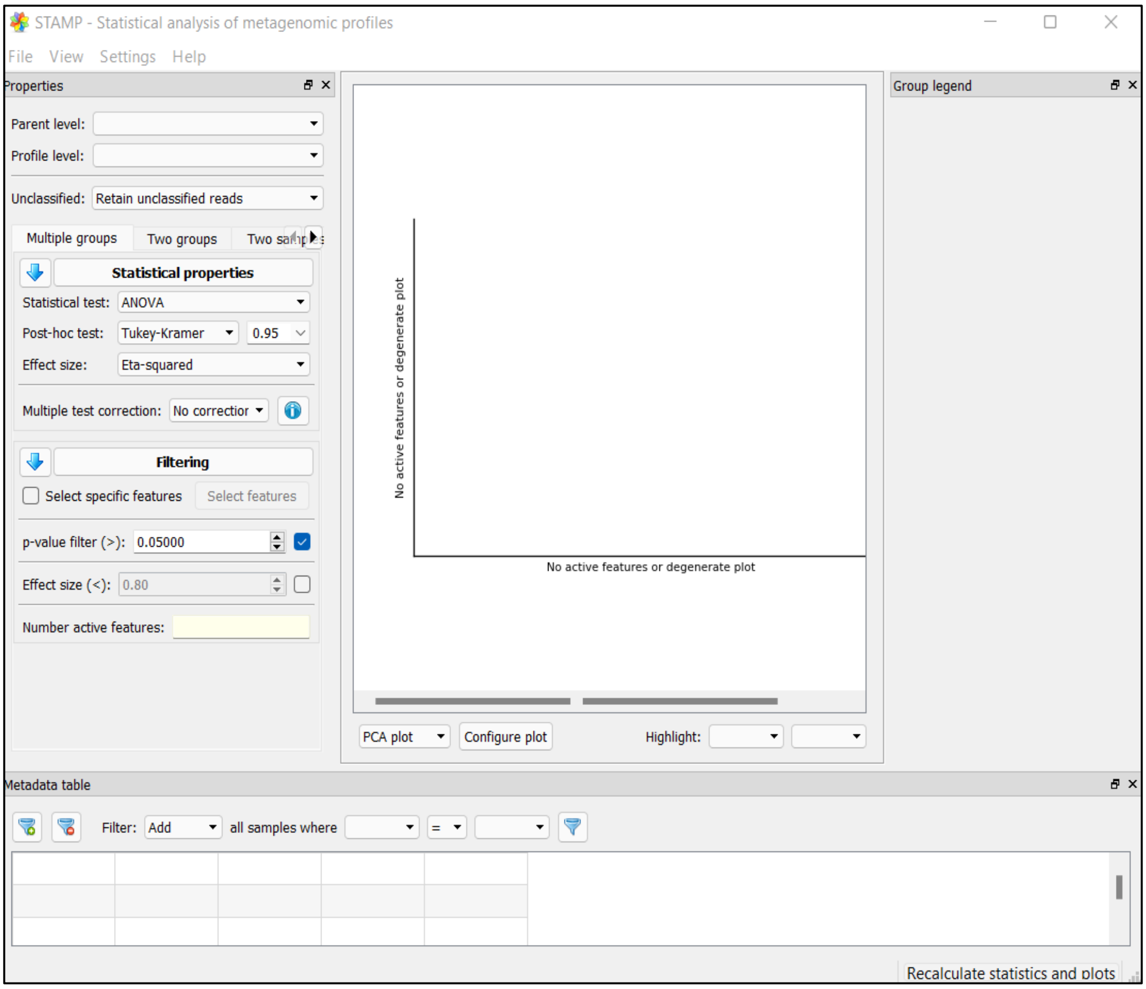
STAMP is a open source software package for analyzing various metagenomic profiles, *viz.*, taxonomic profiles indicating the number of marker genes assigned to different taxonomic units or functional profiles indicating the number of sequences assigned to different subsystems or pathways. A user-friendly, graphical interface permits easy exploration of statistical results and generation of publication quality plots for inferring biological relevant features present in a metagenomic profile. STAMP facilitates statistical hypothesis tests to identify features (*e.g.*, taxa or metabolic pathways) that differ significantly between

1. Pairs of profiles (Two Sample)
2. Sets of profiles organized into two groups (Two Groups)
3. Sets of profiles organized into multiple groups (Multiple Groups)

- ❑ **Software Installation**

STAMP is implemented in Python and can be installed in any operating system, *i.e.*, Windows/ MacOS/Linux. Source codes and executable binary file can be downloaded from the following link:

https://github.com/dparks1134/STAMP/releases/tag/v2.1.3



STAMP - Statistical analysis of metagenomic profiles

Parent level:
Profile level:
Unclassified: Retain unclassified reads
Statistical test: ANOVA
Post-hoc test: Tukey-Kramer 0.95
Effect size: Eta-squared
Multiple test correction: No correctior
□ Select specific features
p-value filter (>): 0.05000 ☑
Effect size (<): 0.80 □
Number active features:
Filter: Add all samples where =

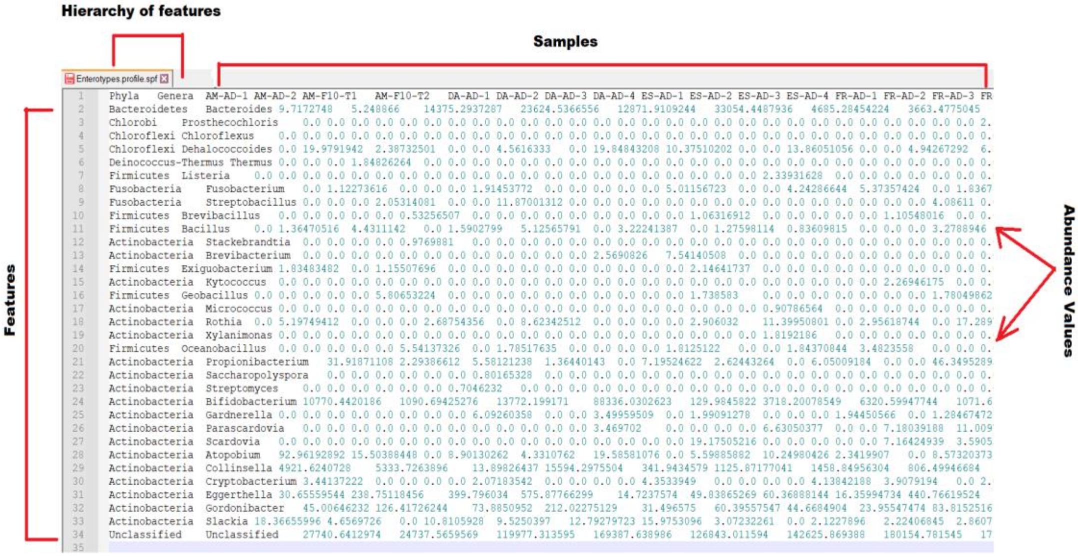
Upon installation of the software, some example datasets also get downloaded in the installation folder. Here, profile and metadata for the dataset *EnterotypeArumugam* is used for the demonstration of this software.

- ❑ **Input files**

STAMP requires 2 input files:

1. Metagenomic profile file
2. Metadata file

1. **Metagenomic profile file**:
   STAMP can analyze both taxonomic and functional profiles. User defined input files should be text files in tab-separated values (TSV) format. It can contain hierarchical profile information for two or more samples. The first row of the file contains headers for each column. First few columns indicate the hierarchical structure of a feature in an arrangement of the highest to the lowest level. There are no restrictions on the depth of the hierarchy but it must form a strict tree structure. Reads that have an unknown classification at any point in the hierarchy should be marked as unclassified (case insensitive). The parent of a classified child in the hierarchy must also be classified. Other columns contain abundance values of features in different samples.

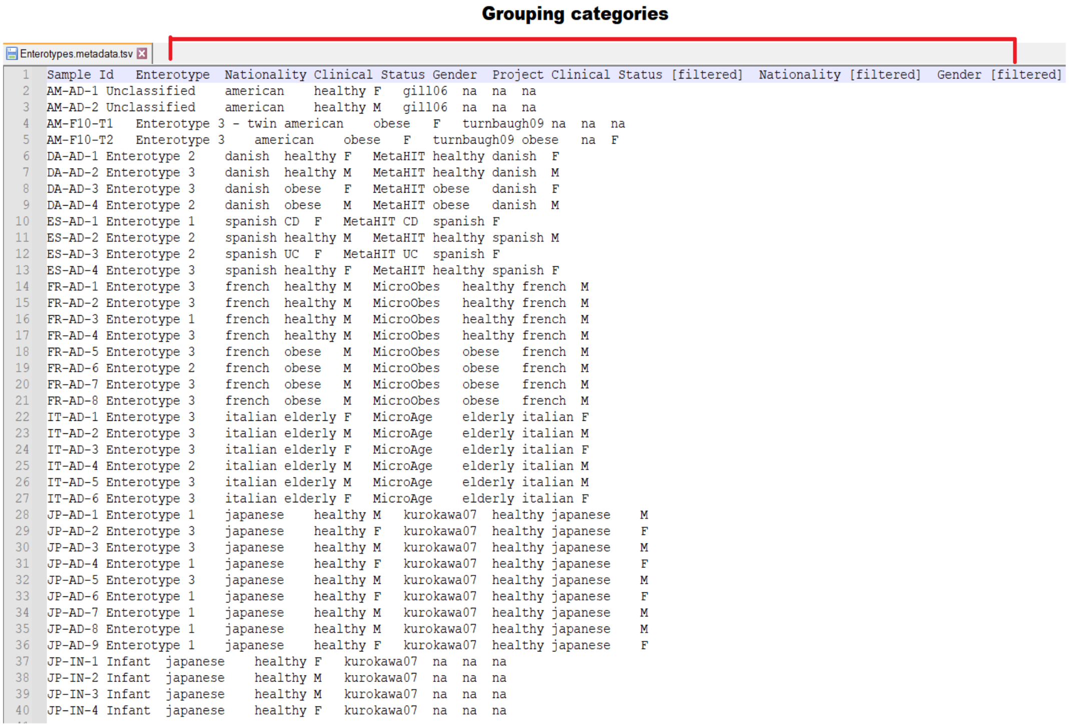


```
Phyla	Genera	AM-AD-1 AM-AD-2 AM-F10-T1   AM-F10-T2   DA-AD-1 DA-AD-2 DA-AD-3 DA-AD-4 ES-AD-1 ES-AD-2 ES-AD-3 ES-AD-4 FR-AD-1 FR-AD-2 FR-AD-3 FR
Bacteroidetes	Bacteroides 9.7172748	5.248866	14375.2937287	23624.5366556	12871.9109244	33054.4487936	4685.28454224	3663.4775045
Chlorobi	Prosthecochloris	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2.
Chloroflexi Chloroflexus	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Chloroflexi Dehalococcoides 0.0 19.9791942	2.38732501	0.0 0.0 4.5616333	0.0 19.84843208 10.37510202 0.0 0.0 13.86051056 0.0 0.0 4.94267292	6.
Deinococcus-Thermus Thermus 0.0 0.0 0.0 1.84826264	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Firmicutes	Listeria	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2.33931628	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Fusobacteria	Fusobacterium	0.0 1.12273616	0.0 0.0 0.0 1.91453772	0.0 0.0 0.0 0.0 0.0 5.01156723	0.0 0.0 4.24286644	5.37357424	0.0 1.8367
Fusobacteria	Streptobacillus 0.0 0.0 0.0 2.05314081	0.0 0.0 11.87001312 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 4.08611 0.
Firmicutes	Brevibacillus	0.0 0.0 0.0 0.0 0.0 0.53256507	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.06316912	0.0 0.0 0.0 0.0 0.0 0.0 1.10548016	0.0 0.
Firmicutes	Bacillus	0.0 1.36470516	4.4311142	0.0 1.5902799	5.12565791	0.0 3.22241387	0.0 1.27598114	0.83609815	0.0 0.0 0.0 3.2788946
Actinobacteria	Stackebrandtia	0.0 0.0 0.0 0.0 0.9769881	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Actinobacteria	Brevibacterium	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2.5690826	7.54140508	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Firmicutes	Exiguobacterium 1.83483482	0.0 1.15507696	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2.14641737	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Actinobacteria	Kytococcus	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 2.26946175	0.0 0.
Firmicutes	Geobacillus 0.0 0.0 0.0 0.0 0.0 5.80653224	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.738583	0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.78049862
Actinobacteria	Micrococcus 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.90786564	0.0 0.0 0.0 0.0 0.0 0.0 0.
Actinobacteria	Rothia	0.0 5.19749412	0.0 0.0 0.0 2.68754356	0.0 8.62342512	0.0 0.0 0.0 0.0 2.906032	11.39950801 0.0 2.95618744	0.0 17.289
Actinobacteria	Xylanimonas 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.8192186	0.0 0.0 0.0 0.0 0.0 0.0 0.
Firmicutes	Oceanobacillus	0.0 0.0 0.0 0.0 0.0 5.54137326	0.0 1.78517635	0.0 0.0 0.0 0.0 1.8125122	0.0 0.0 1.84370844	3.4823558	0.0 0.0 0.
Actinobacteria	Propionibacterium	31.91871108 2.29386612	5.58121238	1.36440143	0.0 7.19524622	2.62443264	0.0 6.05009184	0.0 0.0 46.3495289
Actinobacteria	Saccharopolyspora	0.0 0.0 0.0 0.0 0.0 0.0 0.80165328	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Actinobacteria	Streptomyces	0.0 0.0 0.0 0.0 0.0 0.0 0.7046232	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.
Actinobacteria	Bifidobacterium 10770.4420186	1090.69425276	13772.199171	88336.0302623	129.9845822 3718.20078549	6320.59947744	1071.6
Actinobacteria	Gardnerella 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 6.09260358	0.0 0.0 3.49959509	0.0 1.99091278	0.0 0.0 0.0 1.94450566	0.0 1.28467472
Actinobacteria	Parascardovia	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 3.469702	0.0 0.0 0.0 0.0 6.63050377	0.0 0.0 7.18039188	11.009
Actinobacteria	Scardovia	0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 19.17505216 0.0 0.0 0.0 0.0 0.0 7.16424939	3.5905
Actinobacteria	Atopobium	92.96192892 15.50388448 0.0 8.90130262	4.3310762	19.58581076 0.0 5.59885882	10.24980426 2.3419907	0.0 8.57320373
Actinobacteria	Collinsella 4921.6240728	5333.7263896	13.89826437 15594.2975504	341.9434579 1125.87177041	1458.84956304	806.49946684
Actinobacteria	Cryptobacterium 3.44137222	0.0 0.0 0.0 0.0 2.07183542	0.0 0.0 0.0 0.0 4.3533949	0.0 0.0 0.0 0.0 4.13842188	3.9079194	0.0 2.
Actinobacteria	Eggerthella 30.65559544 238.75118456	399.796034	575.87766299	14.7237574	49.83865269 60.36888144 16.35994734 440.76619524
Actinobacteria	Gordonibacter	45.00646232 126.41726244	73.8850952	212.02275129	31.496575	60.39557547 44.6684904	23.95547474 83.8152516
Actinobacteria	Slackia 18.36655996 4.6569726	0.0 10.8105928	9.5250397	12.79279723 15.9753096	3.07232261	0.0 2.1227896	2.22406845	2.8607
Unclassified	Unclassified	27740.6412974	24737.5659569	119977.313595	169387.638986	126843.011594	142625.869388	180154.781545	17
```

STAMP can analyze taxonomic or functional profiles obtained from MG-RAST software in *.tsv* format. First column of this MG-RAST profile is the *metagenome* column. To perform statistical analysis using STAMP, MG-RAST profile needs to be converted into a STAMP compatible profile (*.spf*) using: File → Create STAMP profile from... → MG-RAST profile

Similarly, taxonomic and functional profiles from BIOM, Rita, CoMet and mothur can also be analyzed using STAMP. It can directly process abundance profiles for multiple samples obtained from the JGI IMG/M web portal. COG profiles from IMG/M do not contain information about which COG category or higher level class a COG belongs to. STAMP can add this information using: Append COG categories to IMG/M profile.

2. **Metadata file**:
   STAMP requires additional data associated with each sample to perform statistical analysis of metagenomic samples organized in two or more groups. These additional information are provided in a metadata file in *.tsv* format. First column of this file indicates Sample Ids. Other columns provide information about various grouping categories and corresponding values.



```
Sample Id   Enterotype   Nationality Clinical Status Gender  Project Clinical Status [filtered]  Nationality [filtered]  Gender [filtered]
AM-AD-1 Unclassified    american    healthy F   gill06  na  na  na
AM-AD-2 Unclassified    american    healthy M   gill06  na  na  na
AM-F10-T1   Enterotype 3 - twin american    obese   F   turnbaugh09 na  na  na
AM-F10-T2   Enterotype 3    american    obese   F   turnbaugh09 obese   na  F
DA-AD-1 Enterotype 2    danish  healthy F   MetaHIT healthy danish  F
DA-AD-2 Enterotype 3    danish  healthy M   MetaHIT healthy danish  M
DA-AD-3 Enterotype 3    danish  obese   F   MetaHIT obese   danish  F
DA-AD-4 Enterotype 2    danish  obese   M   MetaHIT obese   danish  M
ES-AD-1 Enterotype 1    spanish CD  F   MetaHIT CD  spanish F
ES-AD-2 Enterotype 2    spanish healthy M   MetaHIT healthy spanish M
ES-AD-3 Enterotype 2    spanish UC  F   MetaHIT UC  spanish F
ES-AD-4 Enterotype 3    spanish healthy F   MetaHIT healthy spanish F
FR-AD-1 Enterotype 3    french  healthy M   MicroObes   healthy french  M
FR-AD-2 Enterotype 3    french  healthy M   MicroObes   healthy french  M
FR-AD-3 Enterotype 1    french  healthy M   MicroObes   healthy french  M
FR-AD-4 Enterotype 3    french  healthy M   MicroObes   healthy french  M
FR-AD-5 Enterotype 3    french  obese   M   MicroObes   obese   french  M
FR-AD-6 Enterotype 2    french  obese   M   MicroObes   obese   french  M
FR-AD-7 Enterotype 3    french  obese   M   MicroObes   obese   french  M
FR-AD-8 Enterotype 3    french  obese   M   MicroObes   obese   french  M
IT-AD-1 Enterotype 3    italian elderly F   MicroAge    elderly italian F
IT-AD-2 Enterotype 3    italian elderly M   MicroAge    elderly italian M
IT-AD-3 Enterotype 3    italian elderly F   MicroAge    elderly italian F
IT-AD-4 Enterotype 2    italian elderly M   MicroAge    elderly italian M
IT-AD-5 Enterotype 3    italian elderly M   MicroAge    elderly italian M
IT-AD-6 Enterotype 3    italian elderly F   MicroAge    elderly italian F
JP-AD-1 Enterotype 1    japanese    healthy M   kurokawa07  healthy japanese    M
JP-AD-2 Enterotype 3    japanese    healthy F   kurokawa07  healthy japanese    F
JP-AD-3 Enterotype 3    japanese    healthy M   kurokawa07  healthy japanese    M
JP-AD-4 Enterotype 1    japanese    healthy F   kurokawa07  healthy japanese    F
JP-AD-5 Enterotype 3    japanese    healthy M   kurokawa07  healthy japanese    M
JP-AD-6 Enterotype 1    japanese    healthy F   kurokawa07  healthy japanese    F
JP-AD-7 Enterotype 1    japanese    healthy M   kurokawa07  healthy japanese    M
JP-AD-8 Enterotype 1    japanese    healthy M   kurokawa07  healthy japanese    M
JP-AD-9 Enterotype 1    japanese    healthy F   kurokawa07  healthy japanese    F
JP-IN-1 Infant  japanese    healthy F   kurokawa07  na  na  na
JP-IN-2 Infant  japanese    healthy M   kurokawa07  na  na  na
JP-IN-3 Infant  japanese    healthy M   kurokawa07  na  na  na
JP-IN-4 Infant  japanese    healthy F   kurokawa07  na  na  na
```

If metadata file is not provided, STAMP assumes all samples contained in a single group and performs only “Two Sample” tests.

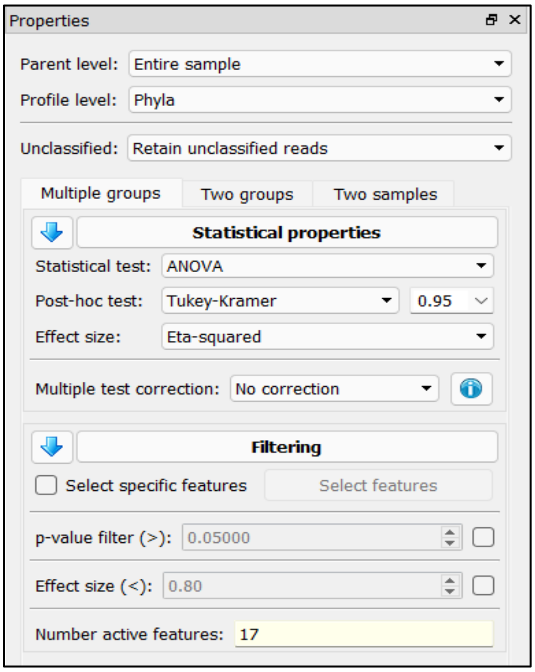
## ❑ Analyzing Metagenomic Profiles:

Upload both profile file and metadata file to the STAMP software to perform various statistical analysis for multiple groups/ two groups/ two samples.

### ❖ Statistical Analysis for Multiple Groups

Statistical properties can be set through the Properties window. It helps to set a number of properties related to performing statistical tests:

- **Parent Level**: The proportion of sequences assigned to a feature will be calculated relative to the total number of sequences assigned to its parent category. By default, it is set as *Entire sample*.
- **Profile Level**: The hierarchical level at which statistical tests will be performed. It facilitates analysis of metagenomic profile at different depths of the hierarchy.
- **Unclassified**: Unclassified sequences can be handled in 3 ways: a) retained in the profile (Retain unclassified reads), removed from the profile (Remove unclassified reads), or removed from consideration except when calculating a profile (Use only for calculating frequency profiles).
- **Statistical Properties**: The statistical test, post-hoc test along with the confidence interval width, effect size, and multiple test correction method to use can be specified in this section. A list of methods provided in STAMP for analyzing multiple groups is given in Table 3.



Parent level: Entire sample

Profile level: Phyla

Unclassified: Retain unclassified reads

Statistical test: ANOVA

Post-hoc test: Tukey-Kramer 0.95

Effect size: Eta-squared

Multiple test correction: No correction

□ Select specific features

p-value filter (>): 0.05000 □

Effect size (<): 0.80 □

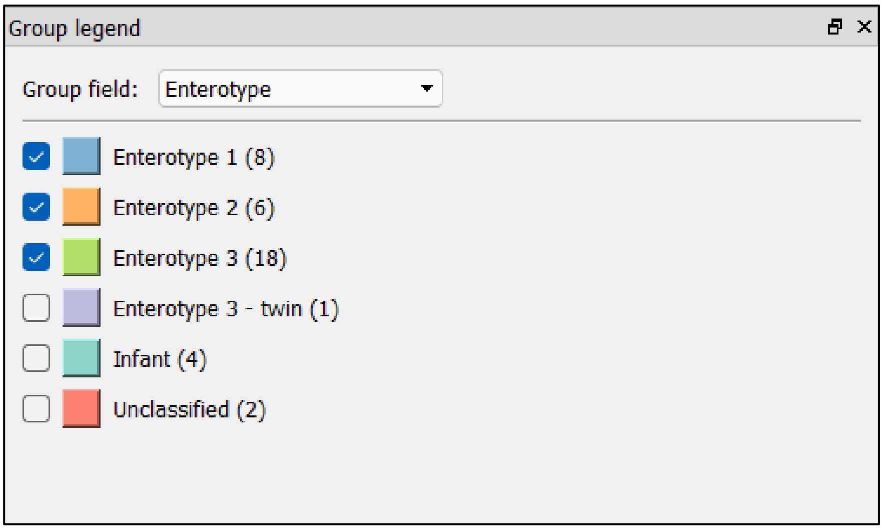
Number active features: 17

- **Filtering**: This section provides a number of filters for identifying features that satisfy a set of criteria (*i.e.*, desired p-value and effect size).

**Table 3: Multiple groups statistical techniques available in STAMP**

| Statistical hypothesis tests | Comments | References |
|---|---|---|
| **ANOVA** | An analysis of variance (ANOVA) is a method for testing whether or not the means of several groups are all equal. It can be seen as a generalization of the t-test to more than two groups. | Bluman, 2007 |
| Kruskal-Wallis H-test | A non-parametric method for testing whether or not the median of several groups are all equal. It considers the rank order of each sample and not the actual proportion of sequences associated with a feature. This has the benefit of not assuming the data is normally distributed. Each group must contain at least 5 samples to apply this test. | Bluman, 2007 |
| **Post-hoc tests** | | |
| **Games-Howell** | Used to determine which means are significantly different when an ANOVA produces a significant p-value. This post-hoc test is designed for use when variances and group sizes are unequal. It is preferable to Tukey-Kramer when variances are unequal and group sizes are small, but it more computationally expensive. | |
| Scheffé | A general post-hoc test for considering all possible contrasts unlike the Tukey-Kramer method which considers only pairs of means. Currently, STAMP only considers pairs of means so the Tukey-Kramer method is preferred. In general, this test is highly conservative. | |
| **Tukey-Kramer** | Used to determine which means are significantly different when an ANOVA produces a significant p-value. It considers all possible pairs of means while controlling the familywise error rate (*i.e.*, accounting for multiple comparisons). In general, we recommend using the Games-Howell post-hoc test when reporting final results and the Tukey-Kramer method for exploratory analysis since it is less computationally intensive. The Tukey-Kramer may also be preferred as it is more widely used and known amongst researchers. | Bluman, 2007 |
| Welch's (uncorrected) | Simple performs Welch's t-test on each possible pair of means. No effort is made to control the familywise error rate. | |
| **Multiple test correction methods** | | |
| Benjamini-Hochberg FDR | Initial proposal for controlling false discovery rate instead of the familywise error. Step-down procedure. | Benjamini and Hochberg, 1995 |
| Bonferroni | Classic method for controlling the familywise error. Often criticized as being too conservative. | Adbi, 2007 |
| Šidák | Less common method for controlling the familywise error rate. Uniformly more powerful than Bonferroni, but requires the assumption that individual tests are independent. | Adbi, 2007 |
| **Storey's FDR** | Recent method used to control the false discovery rate. More powerful than the Benjamini-Hochberg method. Requires estimating certain parameters and is more computationally expensive than the Benjamini-Hochberg approach. | Storey and Tibshirani, 2003<br>Storey *et al.*, 2004 |

- ❖ **Graphical exploration of results:**
  Statistical analysis results can be graphically represented with the help of various plots. The Group legend window helps to select the particular grouping category for which we want to explore the results.



Group legend

Group field: Enterotype

☑ 🟦 Enterotype 1 (8)

☑ 🟧 Enterotype 2 (6)

☑ 🟩 Enterotype 3 (18)

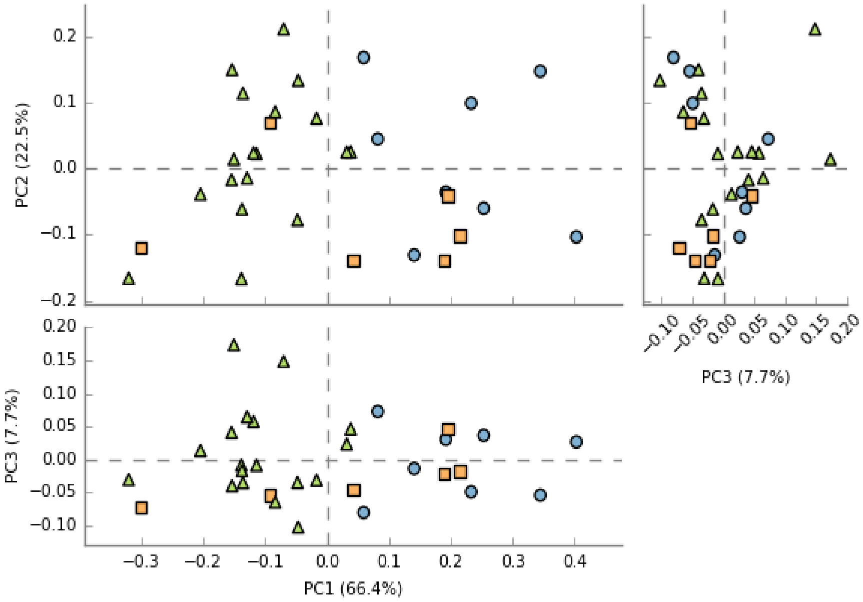
□ 🟪 Enterotype 3 - twin (1)

□ 🟦 Infant (4)

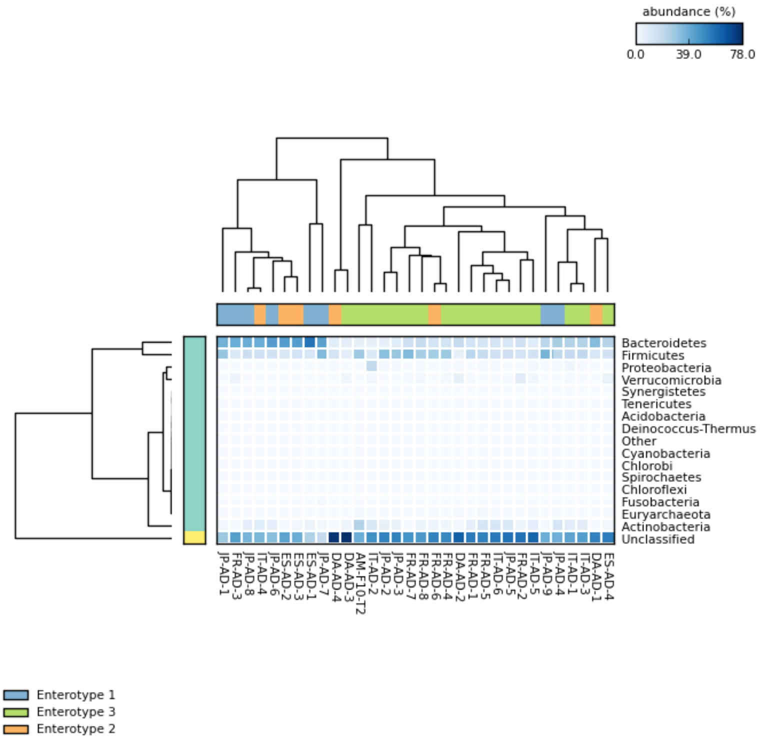
□ 🟥 Unclassified (2)

The following plots can be generated for exploring the analysis results of multiple groups:

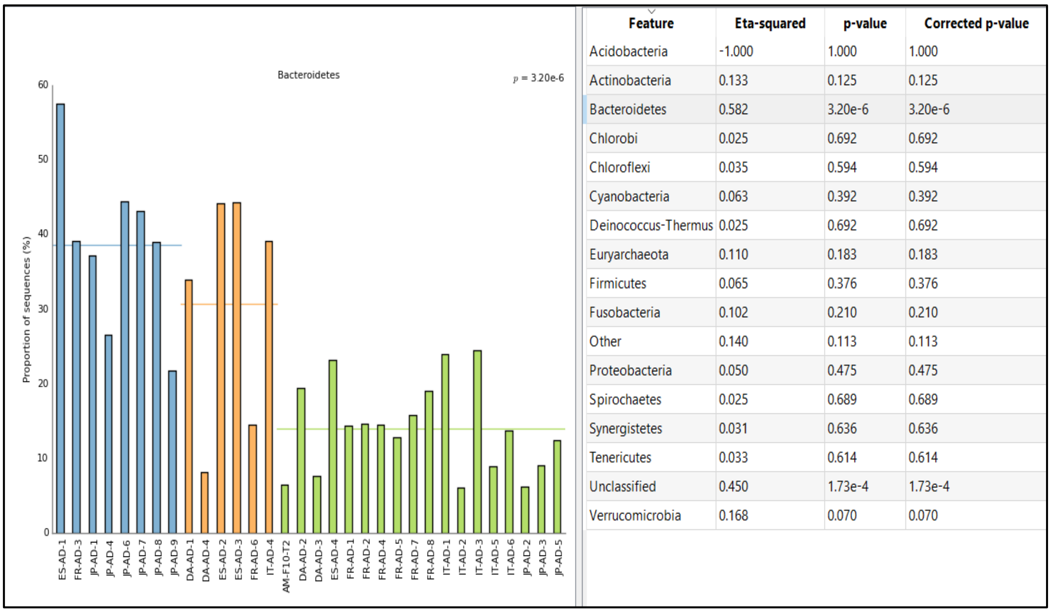
- **PCA plot**: Principal component analysis (PCA) plot of the samples. Clicking on a marker within the plot indicates the sample represented by the marker. Markers of different colours belong to different groups.



- **Heatmap plot**: It represents the proportion of sequences assigned to each feature in every sample. Dendrograms can be shown along the sides of the heatmap and are used to cluster both the features and samples.

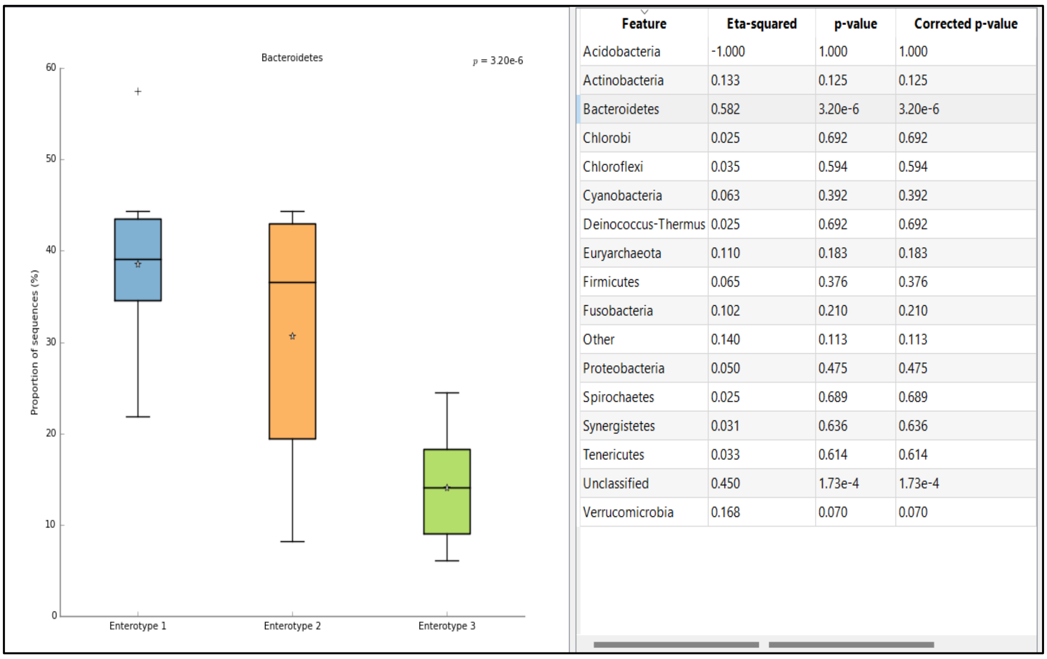


- **Bar plot**: Bar plot represents the proportion of sequences assigned to a particular feature in every sample.



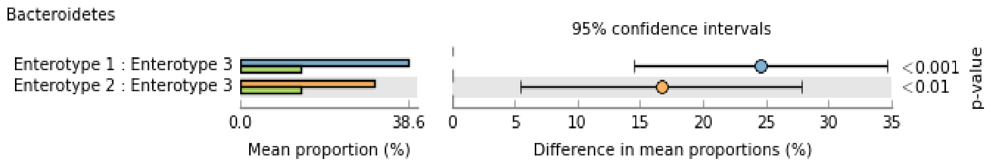
| Feature | Eta-squared | p-value | Corrected p-value |
|---|---|---|---|
| Acidobacteria | -1.000 | 1.000 | 1.000 |
| Actinobacteria | 0.133 | 0.125 | 0.125 |
| Bacteroidetes | 0.582 | 3.20e-6 | 3.20e-6 |
| Chlorobi | 0.025 | 0.692 | 0.692 |
| Chloroflexi | 0.035 | 0.594 | 0.594 |
| Cyanobacteria | 0.063 | 0.392 | 0.392 |
| Deinococcus-Thermus | 0.025 | 0.692 | 0.692 |
| Euryarchaeota | 0.110 | 0.183 | 0.183 |
| Firmicutes | 0.065 | 0.376 | 0.376 |
| Fusobacteria | 0.102 | 0.210 | 0.210 |
| Other | 0.140 | 0.113 | 0.113 |
| Proteobacteria | 0.050 | 0.475 | 0.475 |
| Spirochaetes | 0.025 | 0.689 | 0.689 |
| Synergistetes | 0.031 | 0.636 | 0.636 |
| Tenericutes | 0.033 | 0.614 | 0.614 |
| Unclassified | 0.450 | 1.73e-4 | 1.73e-4 |
| Verrucomicrobia | 0.168 | 0.070 | 0.070 |

- **Box plot**: It is similar to a bar plot. Box plot provides a more concise summary of the distribution of sequence proportions of a feature in various groups. The box-and-whiskers graphics show the median of the data as a line, the mean of the data as a star, the 25th and 75th percentiles of the data as the top and bottom of the box, and uses whiskers to indicate the most extreme data point within 1.5*(75th – 25th percentile) of the median. Data points outside of the whiskers are shown as crosses.



| Feature | Eta-squared | p-value | Corrected p-value |
|---|---|---|---|
| Acidobacteria | -1.000 | 1.000 | 1.000 |
| Actinobacteria | 0.133 | 0.125 | 0.125 |
| Bacteroidetes | 0.582 | 3.20e-6 | 3.20e-6 |
| Chlorobi | 0.025 | 0.692 | 0.692 |
| Chloroflexi | 0.035 | 0.594 | 0.594 |
| Cyanobacteria | 0.063 | 0.392 | 0.392 |
| Deinococcus-Thermus | 0.025 | 0.692 | 0.692 |
| Euryarchaeota | 0.110 | 0.183 | 0.183 |
| Firmicutes | 0.065 | 0.376 | 0.376 |
| Fusobacteria | 0.102 | 0.210 | 0.210 |
| Other | 0.140 | 0.113 | 0.113 |
| Proteobacteria | 0.050 | 0.475 | 0.475 |
| Spirochaetes | 0.025 | 0.689 | 0.689 |
| Synergistetes | 0.031 | 0.636 | 0.636 |
| Tenericutes | 0.033 | 0.614 | 0.614 |
| Unclassified | 0.450 | 1.73e-4 | 1.73e-4 |
| Verrucomicrobia | 0.168 | 0.070 | 0.070 |

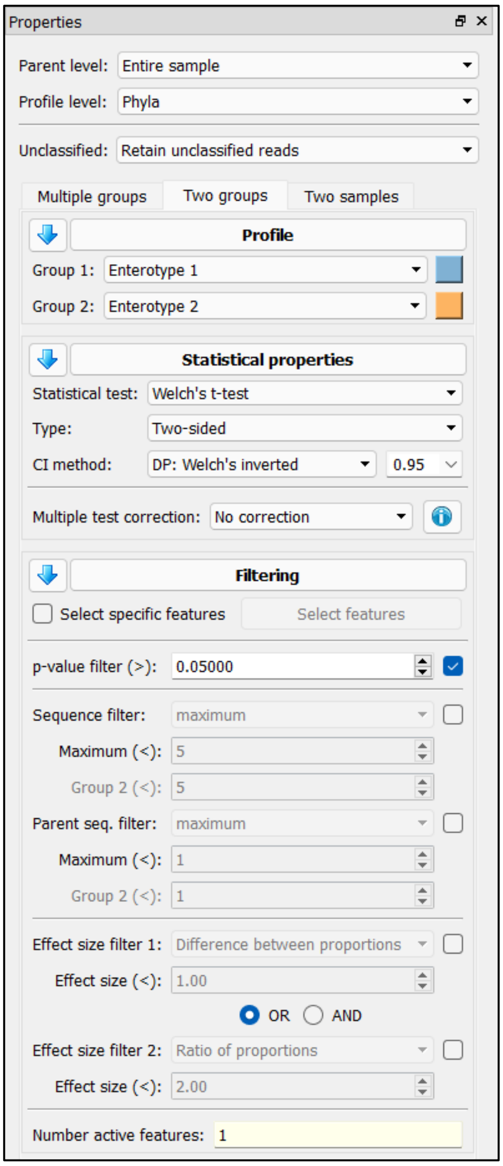
- **Post hoc plot**: Upon rejection of the null hypothesis, post hoc tests are performed to identify which pairs of groups are differing significantly from each other. Post hoc plot shows the results of such a test. It provides p-value and effect size measure for each pair of groups for a particular feature.



Each of these plots provides a number of customization options. To customize a plot, click the Configure plot button below the plot. Plots can also be sent to a new window using the Send plot to window command under the View menu. This allows multiple plots to be viewed at once. Plots can be saved in raster (PNG) and vector (PDF, PS, EPS, SVG) formats (File → Save plot).

## ❖ Statistical Analysis for Two Groups

To analyze a pair of groups, click on the Two groups tab in the Properties window. In the Profile section, we have to specify which pair of groups will be analyzed. Data points of these 2 groups will be represented by 2 different colours. Groupings are determined by the value of the Group field present in the Group legend window. Here, the filtering section provides a large number of filters for identifying features that satisfy a set of criteria.



Properties

Parent level: Entire sample

Profile level: Phyla

Unclassified: Retain unclassified reads

Multiple groups | Two groups | Two samples

**Profile**

Group 1: Enterotype 1 🟦

Group 2: Enterotype 2 🟧

**Statistical properties**

Statistical test: Welch's t-test

Type: Two-sided

CI method: DP: Welch's inverted 0.95

Multiple test correction: No correction

**Filtering**

□ Select specific features  Select features

p-value filter (>): 0.05000 ☑

Sequence filter: maximum □

Maximum (<): 5

Group 2 (<): 5

Parent seq. filter: maximum □

Maximum (<): 1

Group 2 (<): 1

Effect size filter 1: Difference between proportions □

Effect size (<): 1.00

● OR ○ AND

Effect size filter 2: Ratio of proportions □

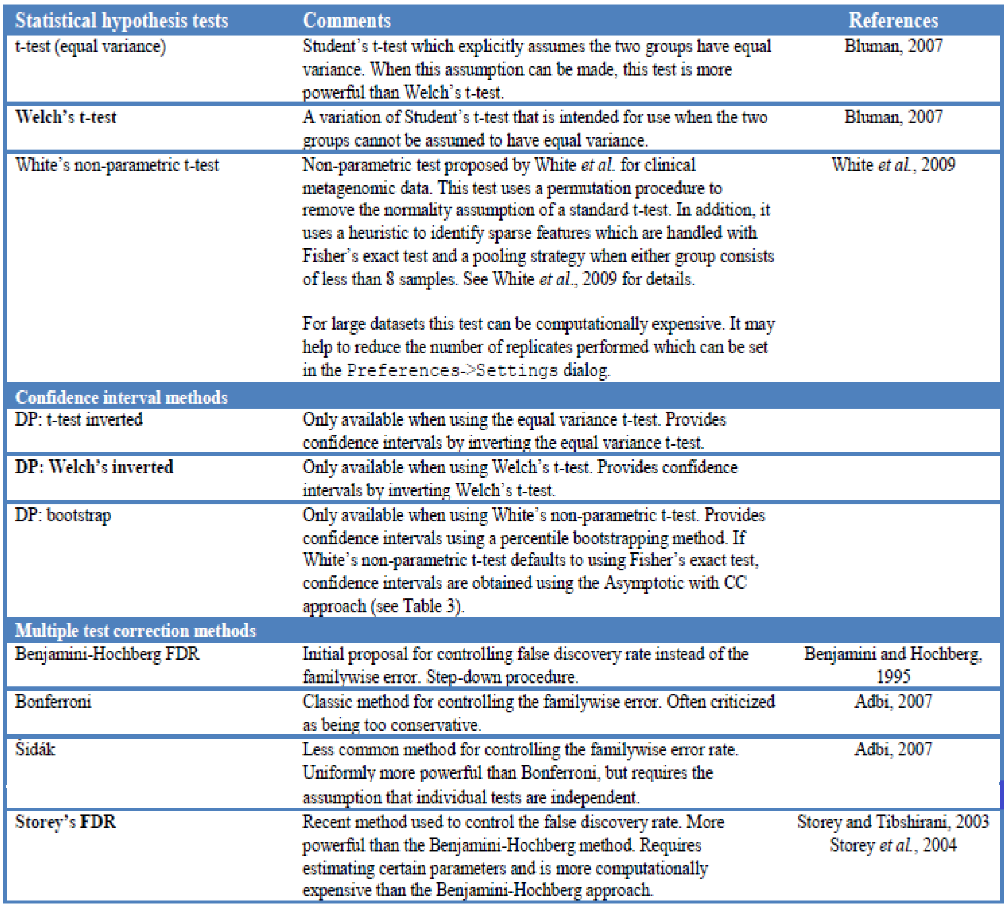
Effect size (<): 2.00

Number active features: 1

Sequence filter removes features that have been assigned fewer than the specified number of sequences. Parent sequence filter does the filtering of sequence counts within parental categories. Effect size filters remove features with small effect sizes. Here, two different effect size statistics are used. It allows one to filter features based on both absolute (*i.e.*, difference between proportions) and relative (*i.e.*, ratio of proportions) measure of effect size.

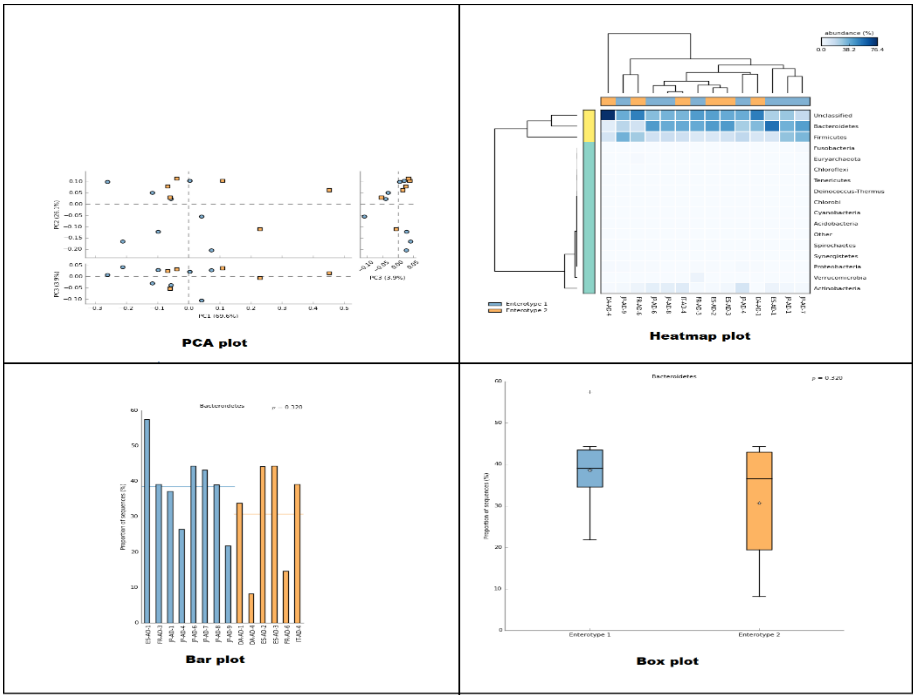
A list of methods for statistical analysis of metagenomic profiles present in two groups is given in Table 4.

### Table 4: Two groups statistical techniques available in STAMP



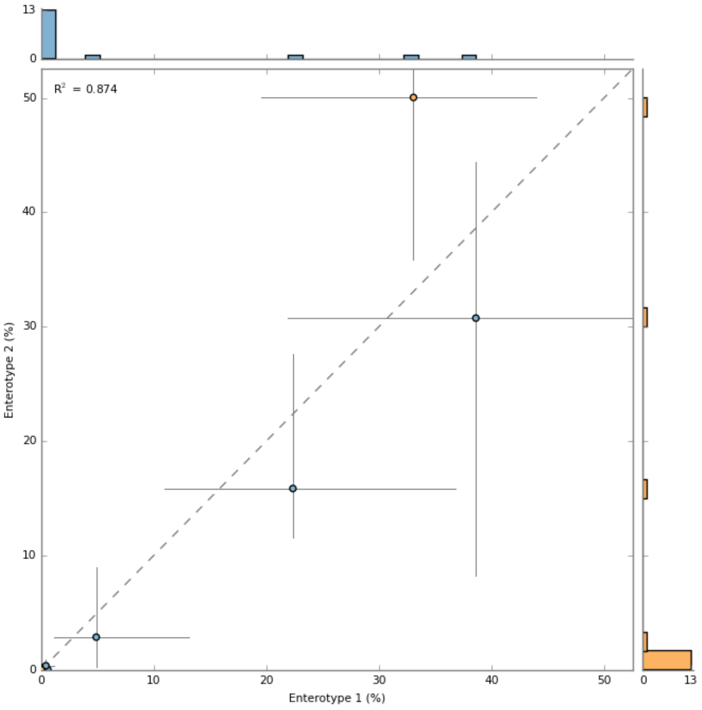
| Statistical hypothesis tests | Comments | References |
|---|---|---|
| t-test (equal variance) | Student's t-test which explicitly assumes the two groups have equal variance. When this assumption can be made, this test is more powerful than Welch's t-test. | Bluman, 2007 |
| Welch's t-test | A variation of Student's t-test that is intended for use when the two groups cannot be assumed to have equal variance. | Bluman, 2007 |
| White's non-parametric t-test | Non-parametric test proposed by White *et al.* for clinical metagenomic data. This test uses a permutation procedure to remove the normality assumption of a standard t-test. In addition, it uses a heuristic to identify sparse features which are handled with Fisher's exact test and a pooling strategy when either group consists of less than 8 samples. See White *et al.*, 2009 for details.<br>For large datasets this test can be computationally expensive. It may help to reduce the number of replicates performed which can be set in the `Preferences->Settings` dialog. | White *et al.*, 2009 |
| **Confidence interval methods** | | |
| DP: t-test inverted | Only available when using the equal variance t-test. Provides confidence intervals by inverting the equal variance t-test. | |
| DP: Welch's inverted | Only available when using Welch's t-test. Provides confidence intervals by inverting Welch's t-test. | |
| DP: bootstrap | Only available when using White's non-parametric t-test. Provides confidence intervals using a percentile bootstrapping method. If White's non-parametric t-test defaults to using Fisher's exact test, confidence intervals are obtained using the Asymptotic with CC approach (see Table 3). | |
| **Multiple test correction methods** | | |
| Benjamini-Hochberg FDR | Initial proposal for controlling false discovery rate instead of the familywise error. Step-down procedure. | Benjamini and Hochberg, 1995 |
| Bonferroni | Classic method for controlling the familywise error. Often criticized as being too conservative. | Adbi, 2007 |
| Šidák | Less common method for controlling the familywise error rate. Uniformly more powerful than Bonferroni, but requires the assumption that individual tests are independent. | Adbi, 2007 |
| Storey's FDR | Recent method used to control the false discovery rate. More powerful than the Benjamini-Hochberg method. Requires estimating certain parameters and is more computationally expensive than the Benjamini-Hochberg approach. | Storey and Tibshirani, 2003<br>Storey *et al.*, 2004 |

- **Graphical exploration of results:**
  Similar to multiple groups, here, bar plot, box plot, PCA plot and heatmap plot can be generated to explore the result of statistical analysis for two groups.

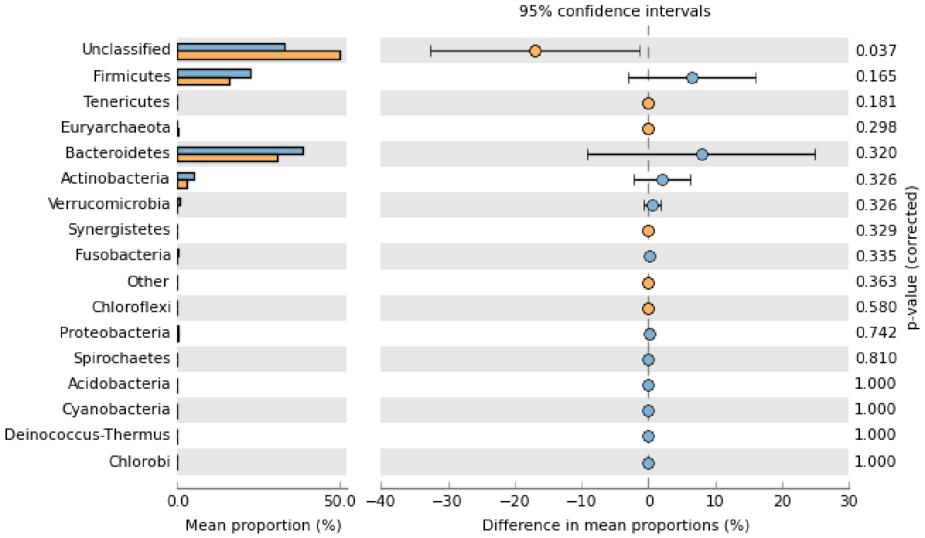


Other plots:

- **Scatter plot**:
  It indicates the mean proportion of sequences within each group which are assigned to each feature. This plot is useful for identifying features that are clearly enriched in one of the two groups. The spread of the data within each group can be shown in various ways (*e.g.*, standard deviation, minimum and maximum proportions).

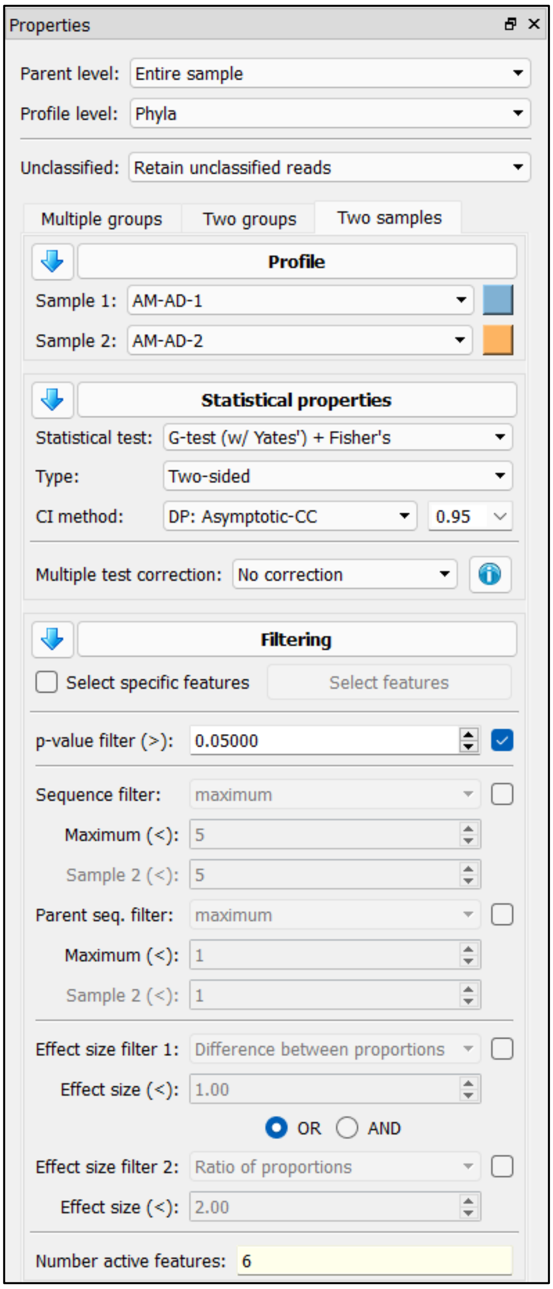


- **Extended error bar plot**:
  It indicates the difference in mean proportion between two groups along with the associated confidence interval of this effect size and the p-value of the specified statistical test. In addition, a bar plot indicates the proportion of sequences assigned to a feature in each group of samples.



- **Statistical Analysis for Two Samples**

To analyze a pair of samples, click on the Two samples tab in the Properties window. The Profile section is used to specify which pair of samples will be analyzed. Data points (features) belonging to these 2 samples will be represented by 2 different colours.



Parent level: Entire sample

Profile level: Phyla

Unclassified: Retain unclassified reads

Sample 1: AM-AD-1 🟦

Sample 2: AM-AD-2 🟧

Statistical test: G-test (w/ Yates') + Fisher's

Type: Two-sided

CI method: DP: Asymptotic-CC 0.95

Multiple test correction: No correction

□ Select specific features

p-value filter (>): 0.05000 ☑

Sequence filter: maximum □

Maximum (<): 5

Sample 2 (<): 5

Parent seq. filter: maximum □

Maximum (<): 1

Sample 2 (<): 1

Effect size filter 1: Difference between proportions □

Effect size (<): 1.00

● OR ○ AND

Effect size filter 2: Ratio of proportions □

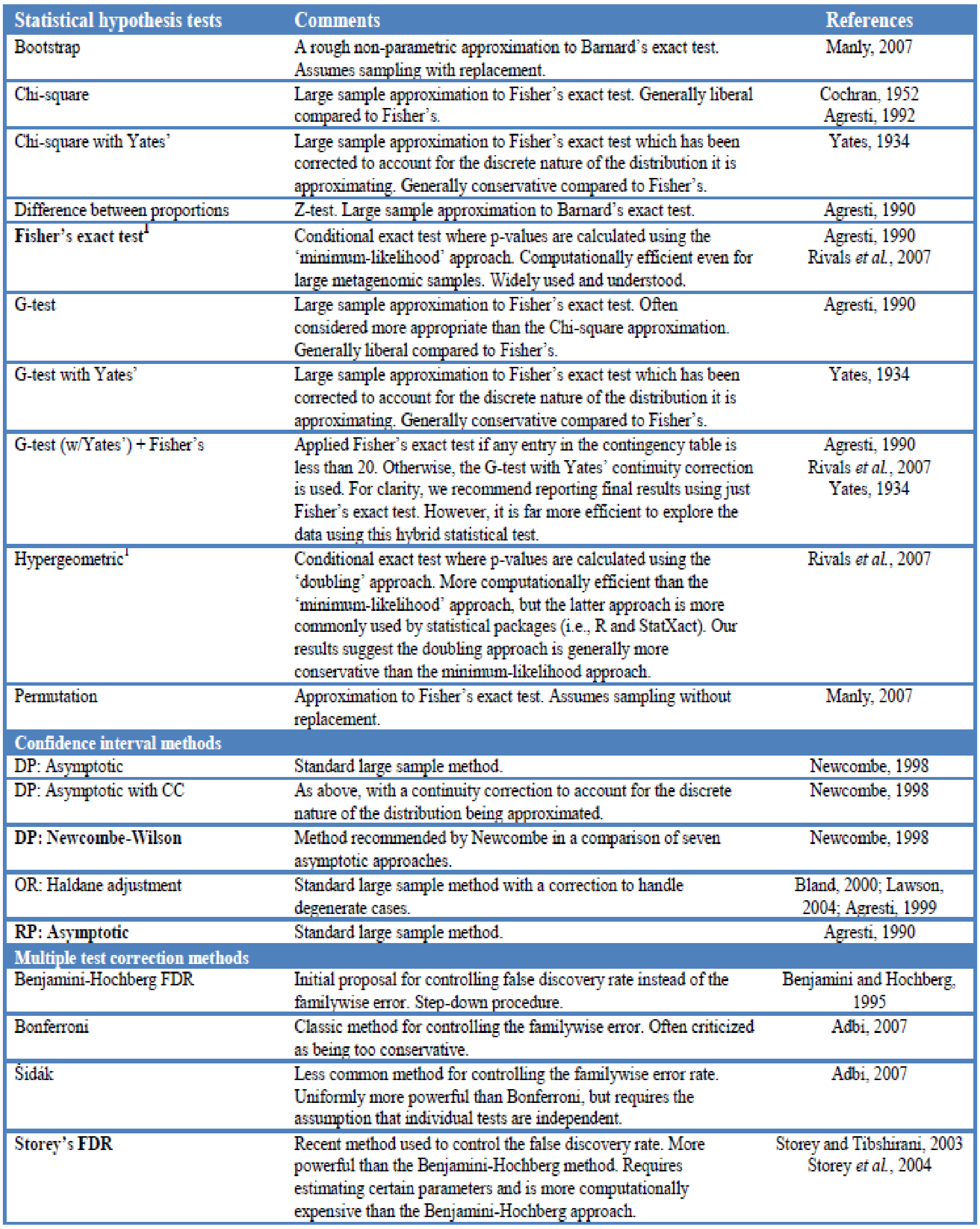
Effect size (<): 2.00

Number active features: 6

Similar to the previous analyses, various statistical properties and filtering criteria can be explicitly mention for the analysis of metagenomic profiles belonging to two different samples.

A list of statistical techniques for the analysis of metagenomic profiles belonging to two different samples is given in Table 5.

## Table 5: Two samples statistical techniques available in STAMP



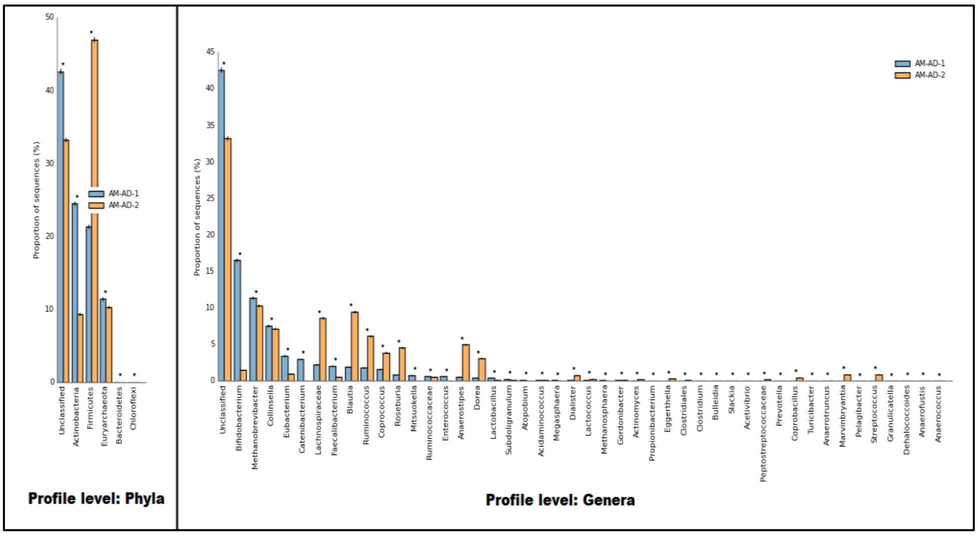
| Statistical hypothesis tests | Comments | References |
|---|---|---|
| Bootstrap | A rough non-parametric approximation to Barnard's exact test. Assumes sampling with replacement. | Manly, 2007 |
| Chi-square | Large sample approximation to Fisher's exact test. Generally liberal compared to Fisher's. | Cochran, 1952<br>Agresti, 1992 |
| Chi-square with Yates' | Large sample approximation to Fisher's exact test which has been corrected to account for the discrete nature of the distribution it is approximating. Generally conservative compared to Fisher's. | Yates, 1934 |
| Difference between proportions | Z-test. Large sample approximation to Barnard's exact test. | Agresti, 1990 |
| **Fisher's exact test**[1] | Conditional exact test where p-values are calculated using the 'minimum-likelihood' approach. Computationally efficient even for large metagenomic samples. Widely used and understood. | Agresti, 1990<br>Rivals *et al.*, 2007 |
| G-test | Large sample approximation to Fisher's exact test. Often considered more appropriate than the Chi-square approximation. Generally liberal compared to Fisher's. | Agresti, 1990 |
| G-test with Yates' | Large sample approximation to Fisher's exact test which has been corrected to account for the discrete nature of the distribution it is approximating. Generally conservative compared to Fisher's. | Yates, 1934 |
| G-test (w/Yates') + Fisher's | Applied Fisher's exact test if any entry in the contingency table is less than 20. Otherwise, the G-test with Yates' continuity correction is used. For clarity, we recommend reporting final results using just Fisher's exact test. However, it is far more efficient to explore the data using this hybrid statistical test. | Agresti, 1990<br>Rivals *et al.*, 2007<br>Yates, 1934 |
| Hypergeometric[1] | Conditional exact test where p-values are calculated using the 'doubling' approach. More computationally efficient than the 'minimum-likelihood' approach, but the latter approach is more commonly used by statistical packages (i.e., R and StatXact). Our results suggest the doubling approach is generally more conservative than the minimum-likelihood approach. | Rivals *et al.*, 2007 |
| Permutation | Approximation to Fisher's exact test. Assumes sampling without replacement. | Manly, 2007 |
| **Confidence interval methods** | | |
| DP: Asymptotic | Standard large sample method. | Newcombe, 1998 |
| DP: Asymptotic with CC | As above, with a continuity correction to account for the discrete nature of the distribution being approximated. | Newcombe, 1998 |
| **DP: Newcombe-Wilson** | Method recommended by Newcombe in a comparison of seven asymptotic approaches. | Newcombe, 1998 |
| OR: Haldane adjustment | Standard large sample method with a correction to handle degenerate cases. | Bland, 2000; Lawson, 2004; Agresti, 1999 |
| **RP: Asymptotic** | Standard large sample method. | Agresti, 1990 |
| **Multiple test correction methods** | | |
| Benjamini-Hochberg FDR | Initial proposal for controlling false discovery rate instead of the familywise error. Step-down procedure. | Benjamini and Hochberg, 1995 |
| Bonferroni | Classic method for controlling the familywise error. Often criticized as being too conservative. | Adbi, 2007 |
| Šidák | Less common method for controlling the familywise error rate. Uniformly more powerful than Bonferroni, but requires the assumption that individual tests are independent. | Adbi, 2007 |
| **Storey's FDR** | Recent method used to control the false discovery rate. More powerful than the Benjamini-Hochberg method. Requires estimating certain parameters and is more computationally expensive than the Benjamini-Hochberg approach. | Storey and Tibshirani, 2003<br>Storey *et al.*, 2004 |

- **Graphical exploration of results:**

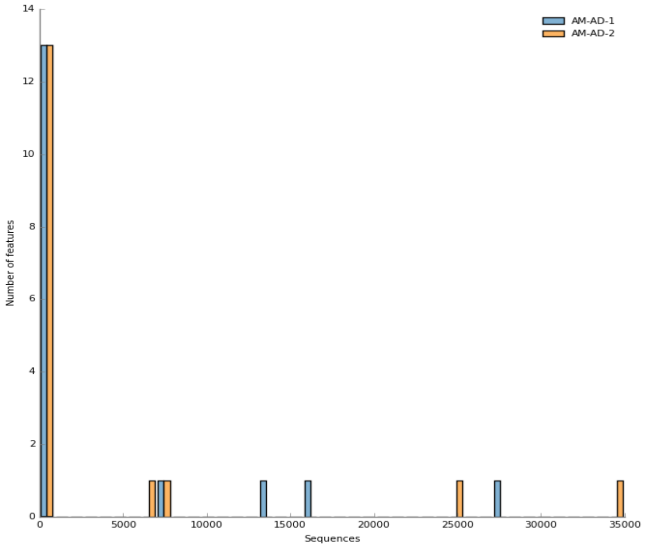
  Similar to the statistical analysis for two groups, here, bar plot, scatter plot and extended error bar plot can be generated to explore the result of statistical analysis of metagenomic profiles belonging to two different samples.

  Other plots:

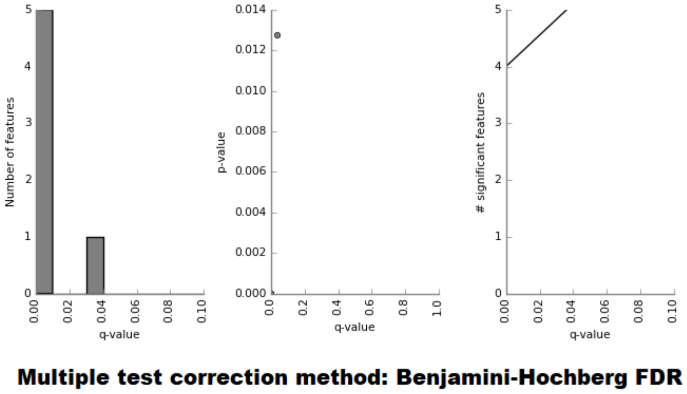
  - **Profile bar plot**: It is a grouped bar plot indicating the proportion of sequences assigned to each feature in the two selected samples. It is recommended for investigating higher hierarchical levels of a profile where the number of features is relatively small. Confidence intervals for each proportion are calculated using the Wilson score method.



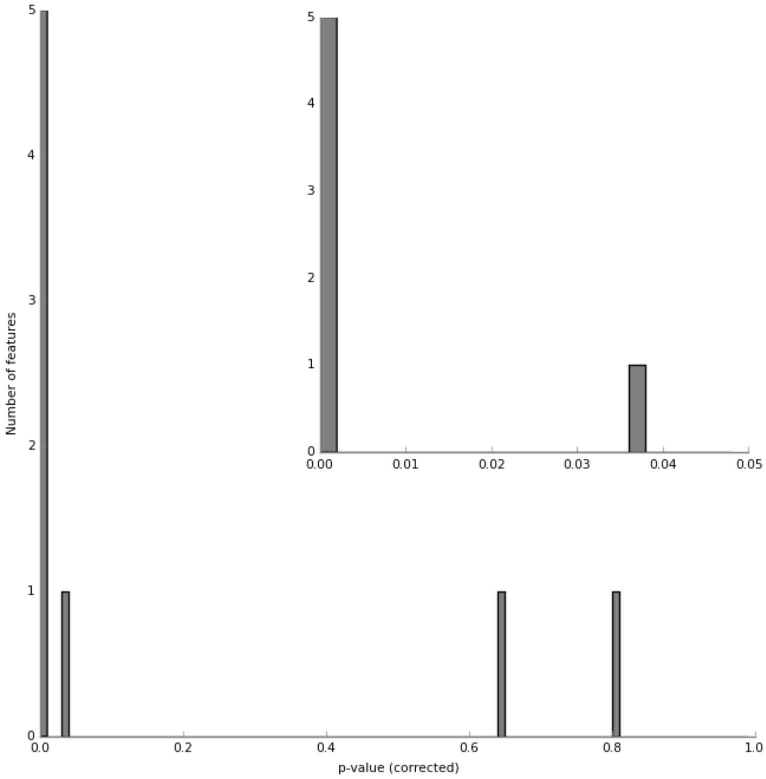
- **Sequence histogram**: It gives a general overview of the number of sequences assigned to each feature in both the samples.



- **Multiple comparison plots**: It can be used to analyze the results of applying a multiple test correction technique, *e.g.*, Benjamini-Hochberg FDR.



- **p-value histogram**: It shows the distribution of p-values and corrected p-values (*i.e.*, number of features corresponding to a particular p-value) in a metagenomic profile.



## References:

Abdi, H. (2007). *Encyclopedia of Measurement and Statistics*. Sage, Thousand Oaks, CA.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, **57**, 289 – 300.

Parks, D. H. and Beiko, R. G. (2010). Identifying biologically relevant differences between metagenomic communities. *Bioinformatics*, **26**, 715 – 721.

Storey, J. D. and Tibshirani, R. (2003). Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences of the United States of America*, **100**, 9440 – 9445.

# Protein Structure Prediction

**Sunil Kumar**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

Protein structure prediction is one of the most significant technologies pursued by computational structural biologist and theoretical chemist. It has the aim of determining the three-dimensional structure of proteins from their amino acid sequences. In other words, this is expressed as the prediction of protein tertiary structure from primary structure.

The practical role of protein structure prediction is now more important than ever. Massive amounts of protein sequence data have been derived from modern large-scale DNA sequencing efforts such as the Human Genome Project. But, the output of experimentally determined protein structures, by time-consuming and relatively expensive X-ray crystallography or NMR spectroscopy, is lagging far behind the output of protein sequences.

Due to exponentially improving computer power, and new algorithms, much progress is being made to overcome these factors by the many research groups that are interested in the task. Prediction of structures for small proteins is now a perfectly realistic goal. A wide range of approaches are routinely applied for such predictions. These approaches may be classified into two broad classes; *ab initio* modeling and comparative or homology modeling.

## *Ab initio* Method

*Ab initio*- or *de novo*- protein modeling methods seek to build three-dimensional protein models "from scratch", i.e., based on physicochemical principles rather than (directly) on previously solved structures. There are many possible procedures that either attempt to mimic protein folding or apply some stochastic method to search possible solutions (i.e., global optimization of a suitable energy function). These procedures tend to require vast computational resources, and have thus only been carried out for tiny proteins. To attempt to predict protein structure *de novo* for larger proteins, we will need better algorithms and larger computational resources like those afforded by either powerful supercomputers (such as Blue Gene or MDGRAPE-3).

## Comparative protein modeling

Comparative protein modeling uses previously solved structures as starting points, or templates. This is effective because it appears that although the number of actual proteins is vast, there is a limited set of tertiary structural motifs to which most proteins belong. It has been suggested that there are only around 2000 distinct protein folds in nature, though there are many millions of different proteins.

These methods may also be split into two groups:

- **Homology modeling** is based on the reasonable assumption that two homologous proteins will share very similar structures. Because a protein's fold is more evolutionarily conserved than its amino acid sequence, a target sequence can be modeled with reasonable accuracy on a very distantly related template, provided that the relationship between target and template can be discerned through sequence alignment. It has been suggested that the primary bottleneck in comparative modeling arises from difficulties in alignment rather than from errors in structure prediction given a known-good alignment. Homology modeling is most accurate when the target and template have similar sequences.
- Protein Threading scans the amino acid sequence of an unknown structure against a database of solved structures. In each case, a scoring function is used to assess the compatibility of the sequence to the structure, thus yielding possible three-dimensional models. This type of method is also known as **3D-1D fold recognition** due to its compatibility analysis between three-dimensional structures and linear protein sequences. This method has also given rise to methods performing an **inverse folding search** by evaluating the compatibility of a given structure with a large database of sequences, thus predicting which sequences have the potential to produce a given fold.

### Homology Modeling: General Procedures

The steps to creating a homology model are as follows:

1) Identify homologous proteins and determine the extent of their sequence similarity with one another and the unknown.

2) Align the sequences.
3) Identify structurally conserved and structurally variable regions.
4) Generate coordinates for core (structurally conserved) residues of the unknown structure from those of the known structure(s).
5) Generate conformations for the loops (structurally variable) in the unknown structure.
6) Build the side-chain conformations.
7) Refine and evaluate the unknown structure.

## 1) Identifying Homologues

Several computerized search methods are available to assist in identifying homologues. In most cases of homology modeling, we have the sequence of a protein for which we want to model the three-dimensional structure (the unknown or target). We then apply sequence search methods to identify proteins with which the unknown has some degree of sequence similarity and for which the three-dimensional structures are available (the templates). We then assume that these proteins are homologous with our unknown and use the three-dimensional structures of these proteins to develop a model of the structure of our unknown. Ideally, one should have several homologues with which to develop a homology model, but modeling can be done with only one known structure.

## 2) Aligning Sequences

A critical step in the development of a homology model is the alignment of the unknown sequence with the homologues. Many methods are available for sequence alignment. Factors to be considered when performing an alignment are-

1) Which algorithm to use for sequence alignment,
2) Which scoring method to apply, and
3) Whether and how to assign gap penalties.

**Algorithms for Alignments**

Sequence alignments generally are based on the dynamic programming algorithm of Needleman and Wunsch. Current methods include FASTA, Smith-Waterman, and BLASTP, with the last method differing from the first two in not allowing gaps.

### Scoring Alignments

Scoring of alignments typically involves construction of a 20x20 matrix in which identical amino acids and those of similar character (i.e., conservative substitutions) may be scored higher than those of different character. Four general types of scoring have been applied to alignments:

**Identity:** considers only identical residues

**Genetic Code:** considers the number of base changes in DNA or RNA to interconvert the codons for the amino acids

**Chemical Similarity:** considers the physico-chemical properties (e.g., polarity, size, charge) with greater weight given to alignment of similar properties

**Observed Substitutions:** considers substitution frequencies observed in alignments of sequences. The substitution schemes are generally considered to be the best methods for scoring alignments. These methods are based on an analysis of the frequency with which a given amino acid is observed to be replaced by other amino acids among proteins for which the sequences can be aligned.

### PAM Matrices

One of the first substitution scoring schemes to be developed was the Dayhoff mutation data matrix. Dayhoff and co-workers developed this method during analysis of the evolution of proteins. The mutation probability matrix that they derived gives the probability of one amino acid mutating to a second amino acid within a particular evolutionary time. The scoring schemes are denoted PAM (Percentage of Acceptable point Mutations) followed by a number. For example, if alignments were scored using PAM40 and PAM250, the lower PAM matrix would recognize short alignments of highly similar sequences and the higher PAM matrix would find longer, weaker local alignments

### BLOSUM Matrices

The PAM substitution matrix is based on substitution frequencies from global alignments of very similar sequences. Henikoff and Henikoff extended this approach by developing substitution matrices using local multiple alignments of more distantly related sequences. A database was assembled that contained multiple alignments (without gaps) of short regions of related sequences. These sequences were clustered into groups (blocks) based on their similarity at some threshold value of percentage identity. Blocks substitution matrices (BLOSUM) were derived based on substitution frequencies for all pairs of amino acids within a group. The different BLOSUM matrices were obtained by varying the threshold. For example, a BLOSUM80 matrix is derived using a threshold of 80% identity.

### Evaluating the Alignment

The final aspect of sequence alignment that should be considered is evaluation of the accuracy of the alignment. The best way to assess the accuracy is to compare alignments from sequence comparisons with alignments from protein three-dimensional structures. Of course this assessment is possible only if you are working with a family of proteins for which three-dimensional structures are known for at least two members of the family. In fact, this approach to evaluation of alignments can be applied during the alignment process.

## 3) Identification of Structurally Conserved and Structurally Variable Regions

After the known structures are aligned, they are examined to identify the structurally conserved regions (SCRs) from which an average structure, or framework, can be constructed for these regions of the proteins. Variable regions (VRs), in which each of the known structures may differ in conformation, also must be identified because special techniques must be applied to model these regions of the unknown protein.

When only one known structure is available for homology modeling, it is more difficult to identify the SCRs. Based on analyses of other homologues for which multiple structures are available; we know that the SCRs generally correspond to the elements of secondary structure, such as alpha-helices and beta-sheets, and to ligand- and substrate-binding sites. Thus, these regions are used as the SCRs in the cases where only one structure is available. The VRs usually lie on the surface of the proteins and form the loops where the main chain turns.

### 4) **Generate coordinates for core (structurally conserved) residues of the unknown structure from those of the known structure(s)**

When generating coordinates for the unknown structure, one needs to model main chain atoms and side chain atoms, both in SCRs and VRs.

For the SCRs, it is straightforward to generate the coordinates of the main chain atoms of the unknown structure from those of the known structure(s). Side chain coordinates are copied if the residue type in the unknown is identical or very similar to that in the known homologues. For other side chain coordinates one can apply a side chain rotamer library in a systematic approach to explore possible side chain conformations. It may be desirable to weight the contribution of each homologue in each SCR based on the extent of similarity with the unknown. In the event that some coordinates in the unknown are undefined in the SCRs, regularization can be used to build and relax both main chain and side chain atoms in those regions. Note that this procedure should be used only if the region of undefined atoms is one or two residues in length.

### 5) Generate conformations for the loops (structurally variable) in the unknown structure

For the VRs, a variety of approaches may be applied in assigning coordinates to the unknown. These regions will correspond most often to the loops on the surface of the protein. If a loop in one of the known structures is a good model for that of the unknown, then the main chain coordinates of that known structure can be copied. Side chain coordinates of residues that are similar in length and character also may be copied. Rotamer libraries can be used to define other side chain coordinates.

When a good model for a loop cannot be found among the known structures, one can search fragment databases for loops in other proteins that may provide a suitable model for the unknown. A residue range is chosen to include the undefined loop as well as a few residues (e.g., three) on either side of the loop for which coordinates have been defined. Fragments are examined for their ability to fit in the undefined region without making bad contacts with other atoms and to overlap well with the residues on either side of the loop. The loop may then be subjected to conformational searching to identify low energy conformers if desired. Coordinates for side chain atoms in these loop regions may be copied if residues are similar, though it is likely that considerable application of side chain rotamer libraries will be required to define coordinates in these regions.

## 6) Evaluation and Refinement of the Structure

For a homology model from any source, it is important to demonstrate that the structural features of the model are reasonable in terms of what is know about protein structures in general. That is, researchers have analyzed three-dimensional structures of proteins from which basic principles of protein structure and folding have been developed. Several programs are available to assist in this analysis of correctness of a homology model. The criteria for analysis of correctness can include:

1) Main chain conformations in acceptable regions of the Ramachandran map.
2) Planar peptide bonds.
3) Side chain conformations that correspond to those in the rotamer library
4) Hydrogen-bonding of polar atoms if they are buried
5) Proper environments for hydrophobic and hydrophilic residues
6) No bad atom-atom contacts
7) No holes inside the structure.

Programs that provide structure analysis along with output includePROCHEK and 3D-Profiler. PROCHECK is based on an analysis of (phi, psi) angles, peptide bond planarity, bond lengths, bond angles, hydrogen-bond geometry, and side-chain conformations of known protein structures as a function of atomic resolution. Thus, the expected values of these parameters are known and can be compared to a modeled structure based on the atomic resolution of the structures from which the model was developed. 3D-profiler compares a homology model to its sequence using a 3D profile. The profile is based on the statistical preferences of each of the 20 amino acids for particular environments within the protein. Each residue position in a 3D model can be characterized by its environment. Preferred environments for amino acids are derived from known three-dimensional structures and are defined by three parameters: (1) the area of each residue that is buried, (2) the fraction of side-chain area that is covered by polar atoms (*i.e.*, O and N), and (3) the local secondary structure. Based on these environment variables, a 3D structure is converted into a 1D profile that describes each residue in the folded protein structure. Examination of these profiles reveals which regions of a sequence appear to be folded correctly and which do not.

Once any irregularities have been resolved, the entire structure may then be subjected to further refinement. This process may consist of energy minimization with restraints, especially for the SCRs. The restraints then may be gradually removed for subsequent minimizations. It also may be advantageous to apply molecular dynamics in conjunction with energy minimization. For any of these refinement procedures, the structure should be solvated, using for example crystallographic waters from the known homologues, a solvent shell, or a periodic box of pre-equilibrated water molecules.

## Databases of Structures from Homology Modeling

Databases are now available that contain large numbers of protein structures that have been obtained by comparative (homology) modeling. Two of these databases are listed here:

1) **ModBase** - It is a query able database of annotated protein structure models. The models are derived by Modpipe,an automated modeling pipeline relying on the programs PSI-BLAST and MODELLER.The database also includes fold assignments and alignments on which the models were based.MODBASE contains theoretically calculated models, which may contain significant errors, not experimentally determined structures.

2) **3DCrunch -** It is a large scale modeling project that aims to submit all entries from protein sequence databases to SWISS-MODEL. Currently the database contains 64,000 entries.

## Automated Web-Based Homology Modeling

Web-based tools are now available to generate models of protein 3-dimensional structures using comparative modeling techniques.

1) **SWISS-MODEL -** It is a fully automated protein structure homology-modeling server, accessible via the ExPASy web server, or from the program Deep View (Swiss Pdb-Viewer). The purpose of this server is to make Protein Modeling accessible to all biochemists and molecular biologists World Wide. The present version of the server is 3.5 and is under constant improvement and debugging. SWISS-MODEL was initiated in 1993 by Manuel Peitsch
2) **WHAT IF** - It is available on EMBL servers, includes three components, one to generate the homology models, one to evaluate the quality of the homology models, and one to evaluate models of proteins for which the structure is already known, thereby providing for evaluation of the quality of the modeling program.

**Source:-**

1) http://en.wikipedia.org/wiki/Homology_modeling
2) http://en.wikipedia.org/wiki/Protein_structure_prediction
3) http://cmbi.kun.nl/gvteach/hommod/index.shtml
4) http://bioinfo.se/kurser/swell/homology.html
5) Sali A, Blundell TL. (1993). Comparative protein modelling by satisfaction of spatial restraints. *J Mol Biol* 234(3):779-815
6) Fiser A, Sali A. (2003). ModLoop: automated modeling of loops in protein structures. *Bioinformatics* 19(18):2500-2510
7) John B, Sali A. (2003). Comparative protein structure modeling by iterative alignment, model building and model assessment. *Nucleic Acids Res* 31(14):3982-3992

# Molecular Docking

**Sunil Kumar**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

**Objective:**
To find the interaction between the protein and a ligand molecule by performing docking studies.

**Theory**
A molecule is a small chemical element that is made up of two or more atoms held together by chemical bonds. A molecule can be composed of either single kind of element (e.g. $H_2$) or different kinds of elements (e.g. $CO_2$). Molecules can be found in both living things and non living things. A drug is a small molecule that can interact, bind and control the function of biological receptors that helps to cure a disease. Receptors are proteins that interact with other biological molecules to maintain various cellular functions in plants. Enzymes, hormone receptors, cell signaling receptors, neurotransmitter receptors etc. are some important receptors in plants.

Drug designing is a process of designing a drug molecule that can interact and bind to a target. Receptors are molecules which can be seen on the surface of the cell which receives signals and can be defined as a molecule which recognizes a small molecule, which on binding triggers a cellular process. In an unbounded state receptor, functionalities of the receptor remain silent. Hence this definition says that receptor binds specifically to a particular ligand or vice versa, but in some cases high concentrations of ligands will binds to a multiple receptor sites.

Drug receptors usually remain without endogenous ligand. The receptors for these drugs molecules can be an enzyme, an ion channels, proteins, nucleic acids etc. Hence the drug molecule will go and cross link the DNA and stops DNA replication. Receptors for endogenous regulatory ligands are hormones, growth factors etc. Hence the function of these receptors is to sense the ligands and to initiate the response. For example, Aspirin is a small pain killer drug molecule which contains nine carbon atoms, eight hydrogen atoms and four oxygen atoms. Design of the molecules should be complementary in shape and charge to the target.

Molecular modeling includes computational techniques that are used to model a molecule. Drug designing by using these modeling techniques is referred to as computer-aided drug design. Computer based drug design is a fast, automatic, very low cost process. It can be done either by Ligand based drug design or Structure based drug design. Ligand based drug design purely based on the model which is going to bind to the target, defining of pharmacophoric regions are necessary for the molecule in order to bind the target but Structure based drug design is based on the 3 dimensional structure of the target. If any target is not available it can be created by using homology modeling. Using the structure of the target predict the drug molecules binding affinity to the target. Building a molecule using computer techniques is a very important step in drug deigning. There are so many computational tools available for building a molecule.

After modeling a molecule, check where the ligand get docked onto the receptor, and check whether the ligand fits for the target molecule and go for Docking studies.

**Protein ligand interaction:**

Proteins are the fundamental units of all living cells and play a vital role in various cellular functions. Each protein has specific function in plants. The structure of the protein determines its function. The binding of a protein with other molecules is very specific to carry out its function properly. For this reason every protein has a particular structure. A molecule is a small chemical element that is made up of two or more atoms held together by chemical bonds. A drug is a small molecule that can interact, bind and control the function of biological receptors that helps to cure a disease.

Protein–ligand interactions are essential for all processes happening in living organisms. Ligand-mediated signal transmission through molecular complementary is essential to all life processes; these chemical interactions comprises biological recognition at molecular level. The evolution of the protein functions depends on the development of specific sites which are designed to bind ligand molecules. Ligand binding capacity is important for the regulation of biological functions. Protein-Ligand interactions occur through the molecular mechanics involving the conformational changes among low affinity and high affinity states. Ligand binding interactions changes the protein state and protein function.

**Key concepts of protein ligand interaction:**

1. Every biological reaction is initiated by protein-ligand interaction step. Such reactions never involve in the binding of single ligand or single step.
2. Binding of two or more ligands to a same protein indicates mutual interaction.
3. Ligand binding plays an important role in regulation of biological function.
4. Ligand binding may leads to the conformational changes in proteins.
5. Ligand and macromolecule interaction provides the strength of the interaction.

**What is Docking?**

Docking is a method which predicts the preferred orientation of one molecule to another molecule when they are bound together to form a stable complex. Molecular docking can be referred as “lock and key” model. Here the protein can be called as a lock and the ligand can be called as key, which describes the best fit orientation of the ligand which it goes and binds to a particular protein.

To perform a docking, first one may require a protein molecule. The protein structures and ligands are the inputs for the docking.

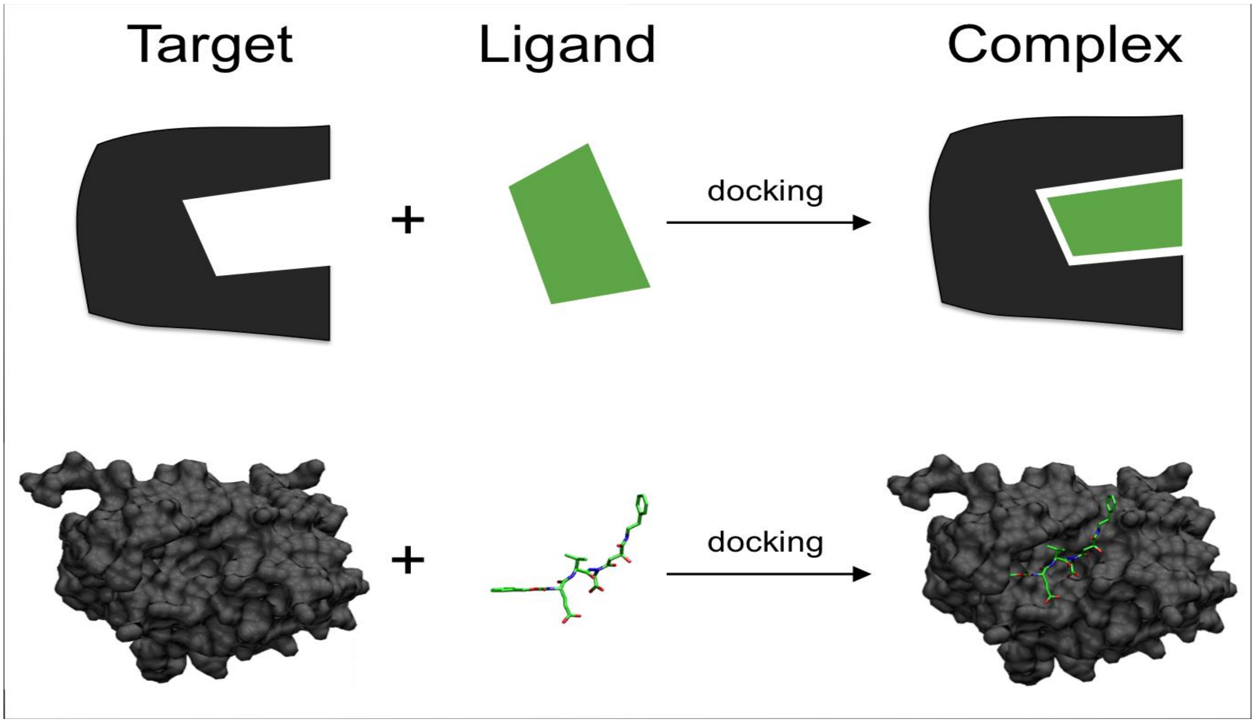


Figure1: Example of Docking

**Docking can be based on two separate platforms.**

**1. Search algorithm**

Search algorithm creates an optimum number of configurations that includes the binding modes which are determined experimentally. Configurations are evaluated using scoring functions to differentiate the binding modes from the other modes.

**The common search algorithms are:**

1. Monte Carlo methods
2. Genetic algorithms
3. Fragment-based methods
4. Point complimentary methods
5. Tabu searches
6. Systematic searches
7. Molecular dynamics.

**2. Scoring function:**

Scoring functions are developed to find the interactions between the protein- protein interactions and protein-DNA interactions. Scoring methods are the mathematical methods used to predict the strength of interaction between two molecules.

**Steps for Docking:**

1. **Preparation of the Protein molecule :**
   Download the protein structure to the working directory. Remove the water molecules and add hydrogens to the molecule to satisfy the valances of the molecule. X-ray crystallographic structures cannot resolove the hydrogen, so in most of the PDB structures hydrogens are absent. Remove the disulphide and trisulphide bonds of a protein using AutoDock. After the preparation of the molecules, molecules has to be minimized.

2. **Preparation of ligand molecules :**
   Prepare a ligand molecule which is going to bind to the target add hydrogen atoms to the molecule and filter the unwanted molecules based on their properties like water and small ions. If the stereoisomers are missing from the Molecule it requires adding stereo chemical information. Optimize the geometry of the molecule. Take the molecule for docking studies.

3. **Surface representation:**
   Take a receptor and ligand molecule for studies, receptor as a static and ligand molecule as flexible. Find the Surface of the molecules by using geometric features of the molecules. Grid points are used to find the surface area.

4. **Feature calculation**
   Features are the methods which are used to find the potential docking sites that are derived from surface representation.

5. **Docking**
   It is important to find the cavities on the surface of the receptor in protein Ligand interaction.

6. **Evaluation of Docking result:**
   Dock the each individual parts, docking of each segments gives the total score.

**Types of Docking:**

**Rigid Docking**: In a rigid molecular docking the molecules are referred as rigid objects they cannot change their shape during the docking

**Flexible Docking**: In a flexible docking the molecules are referred as flexible objects that they can change their shapes according to the ligand and the target during docking process.

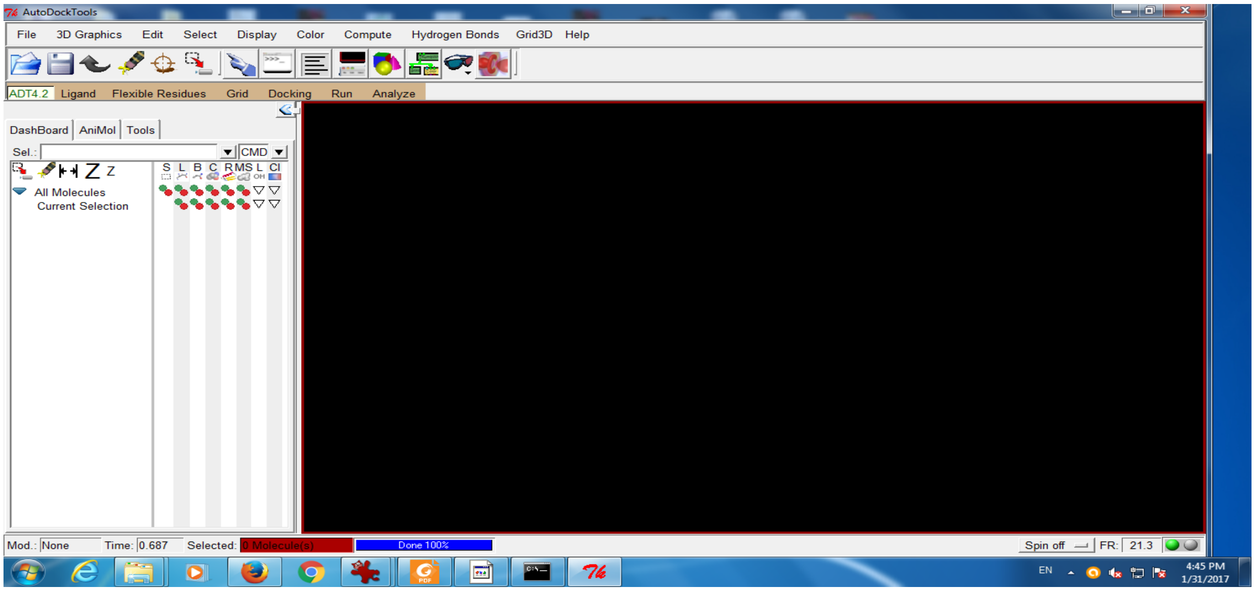
## AutoDock:

AutoDock is a docking tool, which is designed to predict the behavior of the small molecules and helps user to perform the docking of ligands to a set of grids which describes the target, once docking completes result can visualize in 3D view. AutoDock 4 is freely available under the GNU General Public License. AutoDock uses a Monte Carlo simulation with a rapid energy evaluation using grid based molecular affinity potentials. It is given a volume around the protein, the rotatable bonds for the substrate, and an arbitrary starting configuration, and the procedure produces a relatively unbiased docking.

Different applications of AutoDock:

1. Structure based drug design.
2. X-ray crystallography
3. Lead optimization
4. Combinatorial library design
5. Protein-Protein docking.
6. Chemical mechanism studies.

## Home page of AutoDock:



Sel.:

Mod.: None Time: 0.687 Selected: 0 Molecule(s) Done 100% Spin off FR: 21.3

## Procedure

Here one can perform rigid docking where the protein and the ligand molecule are non flexible. Here phosphatidyl-inositol-3-kinases (PDB ID -1E7U) is used as an example for receptor and its ligand KWT. Autodock Tools can be used to prepare PDBQT molecules of the receptor and ligand with PDBQT format, in which PDB format contains partial charges (“Q”) and atom types (“T”).

1. Open the Autodock software by clicking on Autodock icon from your desktop. (Figure 1).

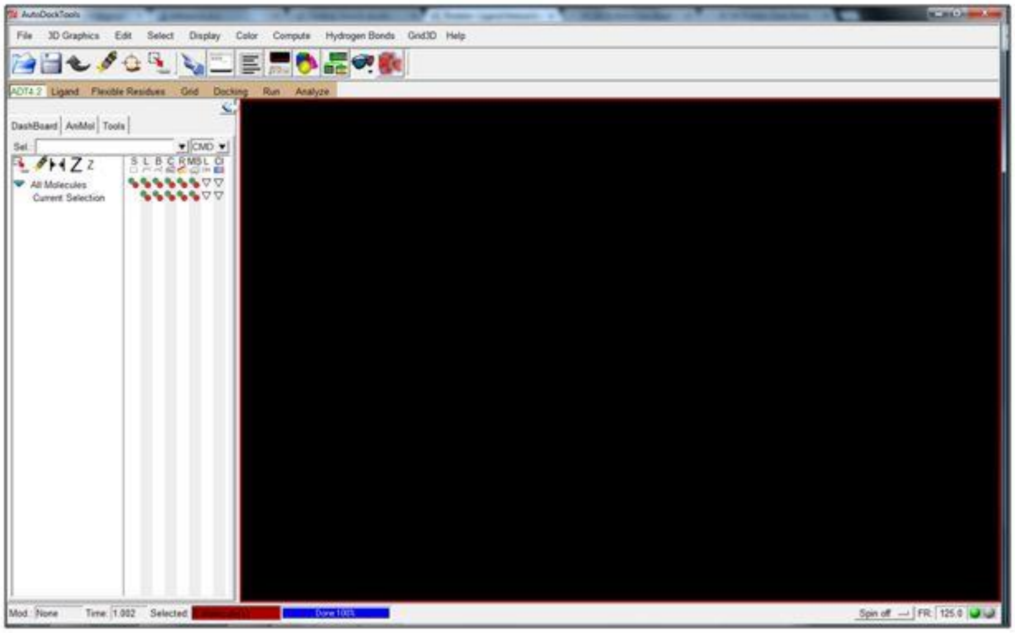


Figure 1: AutoDock GUI

2. Read the downloaded PDB molecule 1E7U in the work space panel by clicking on the tab “File“ and then select “Read molecules” as shown in Figure 2.

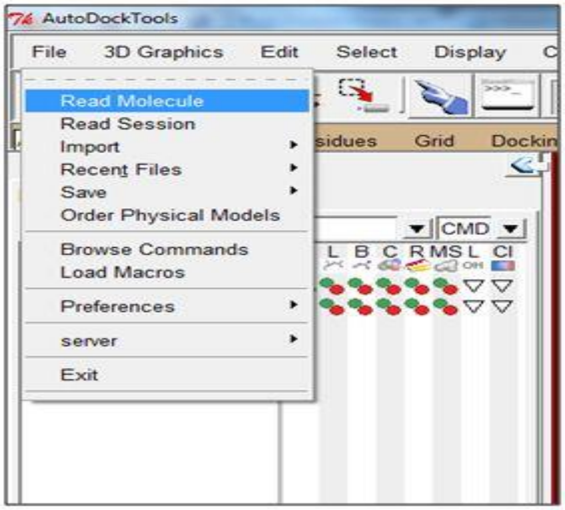


Figure 2: To read a molecule

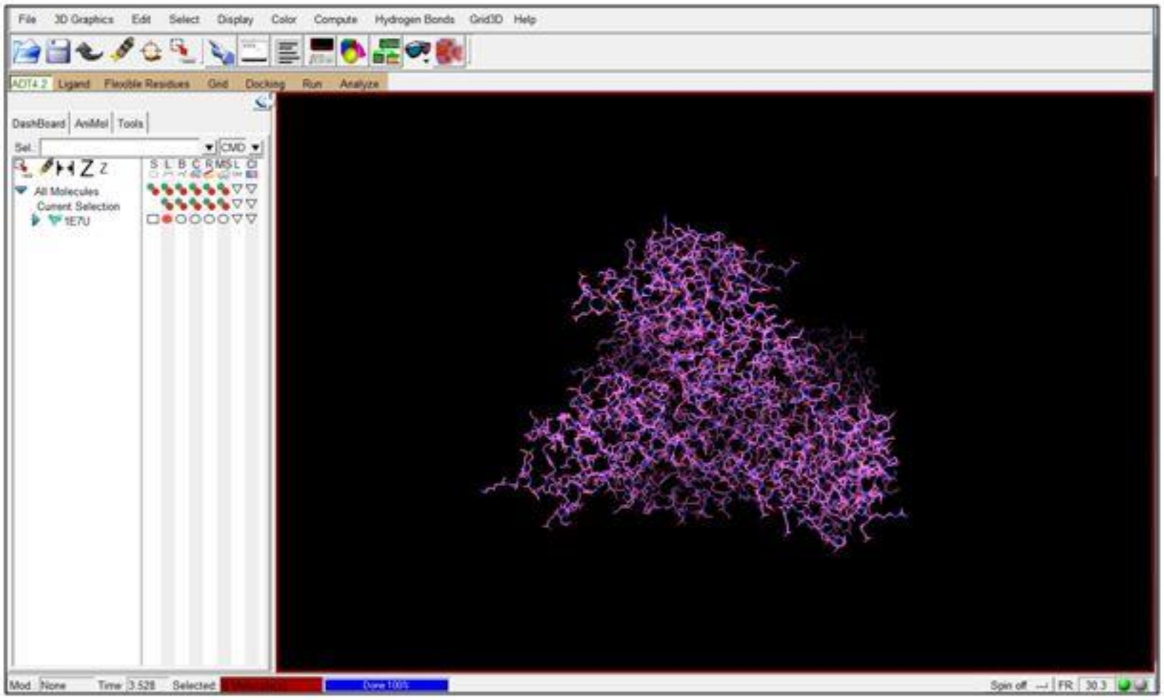


Figure 3: 1E7U

3. PDB files can have errors such as missing atoms, chain breaks, water molecules etc. which is needed to be corrected. Select all water molecules which obstruct the accuracy of docking procedure.

4. Click on the “Edit” tab and select “Delete Water” to delete the water molecules from the receptor molecule as shown in Figure 4.

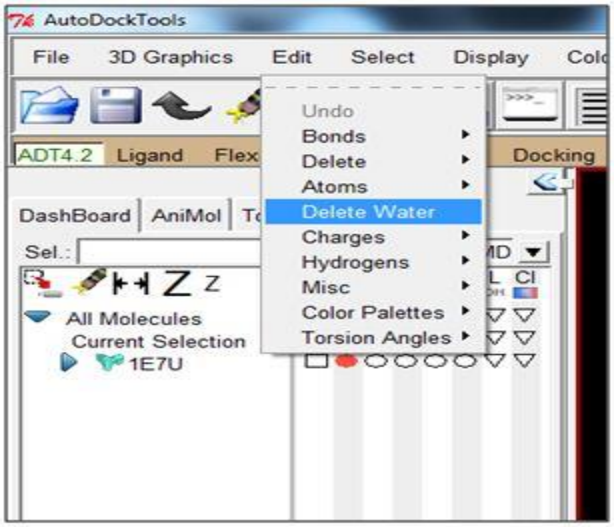


Figure 4: Deleting water molecule

5. For adding Hydrogens to satisfy valency, Click on the "Edit" tab and select "Hydrogen" and then select "Add" option as shown in Figure 5.

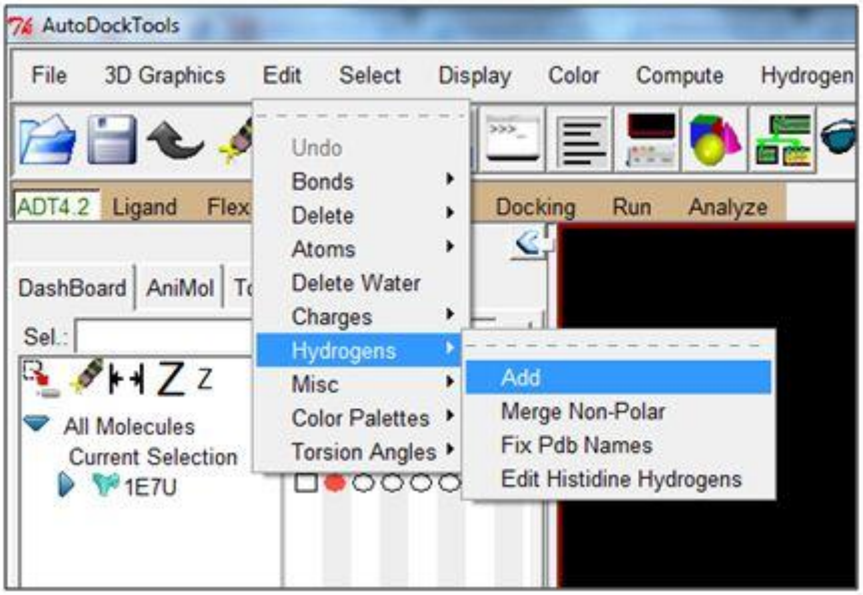
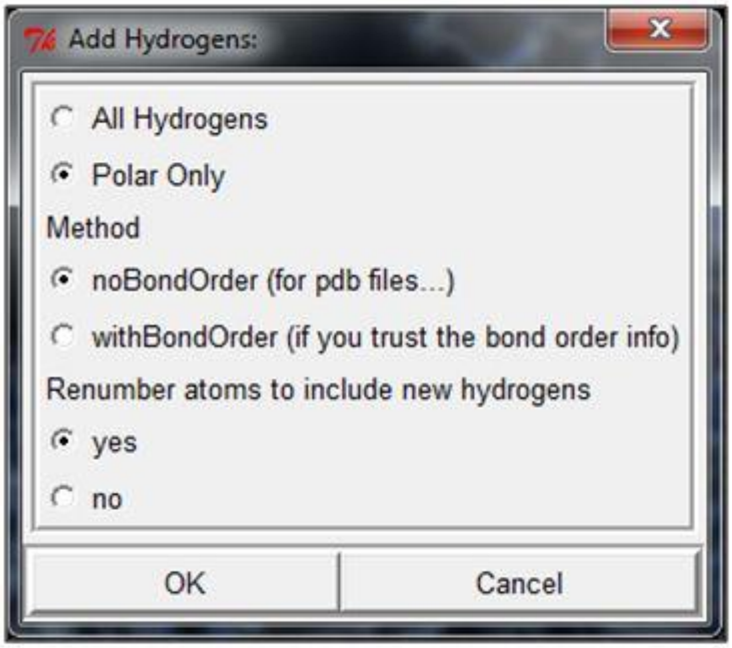


Figure 5: Adding Hydrogen to the receptor

6. Now select "Polar Only" -> "noBondOrder"->"Yes" respectively and then click on the "Ok" option  as shown in Figure 6.


Add Hydrogens:

○ All Hydrogens

● Polar Only

Method

● noBondOrder (for pdb files...)

○ withBondOrder (if you trust the bond order info)

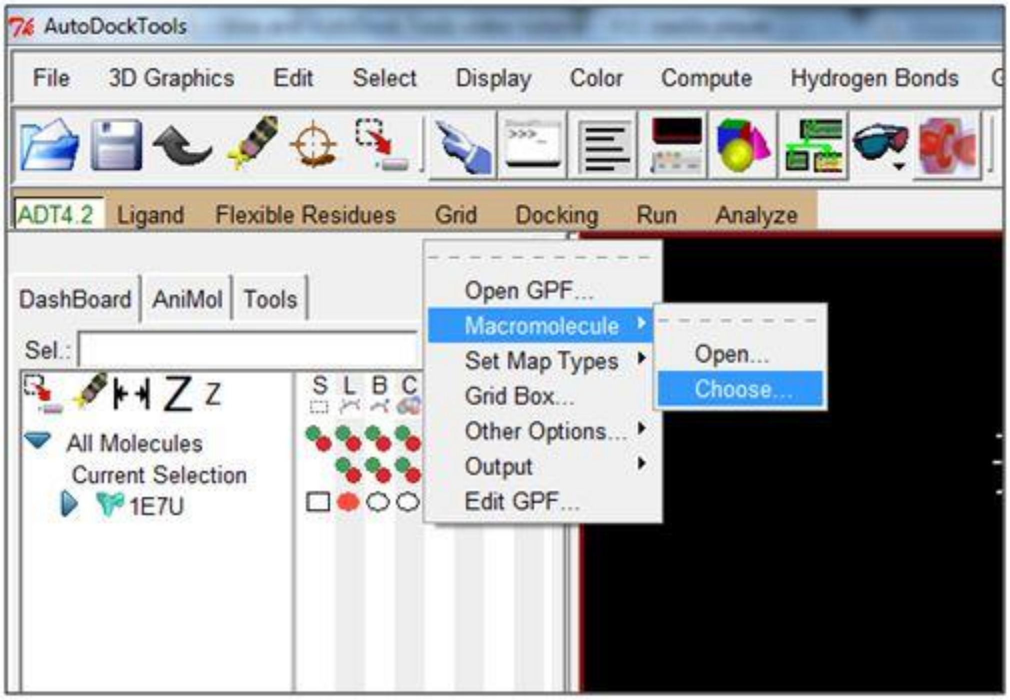
Renumber atoms to include new hydrogens

● yes

○ no

Figure 6: Adding Hydrogen

7. Click on the “Grid” option and select “Macromolecules” and select Choose option for selecting the molecule  as shown in Figure 7 and 8.



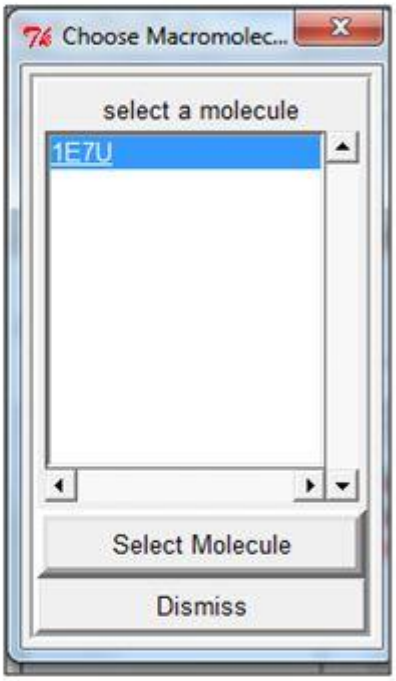
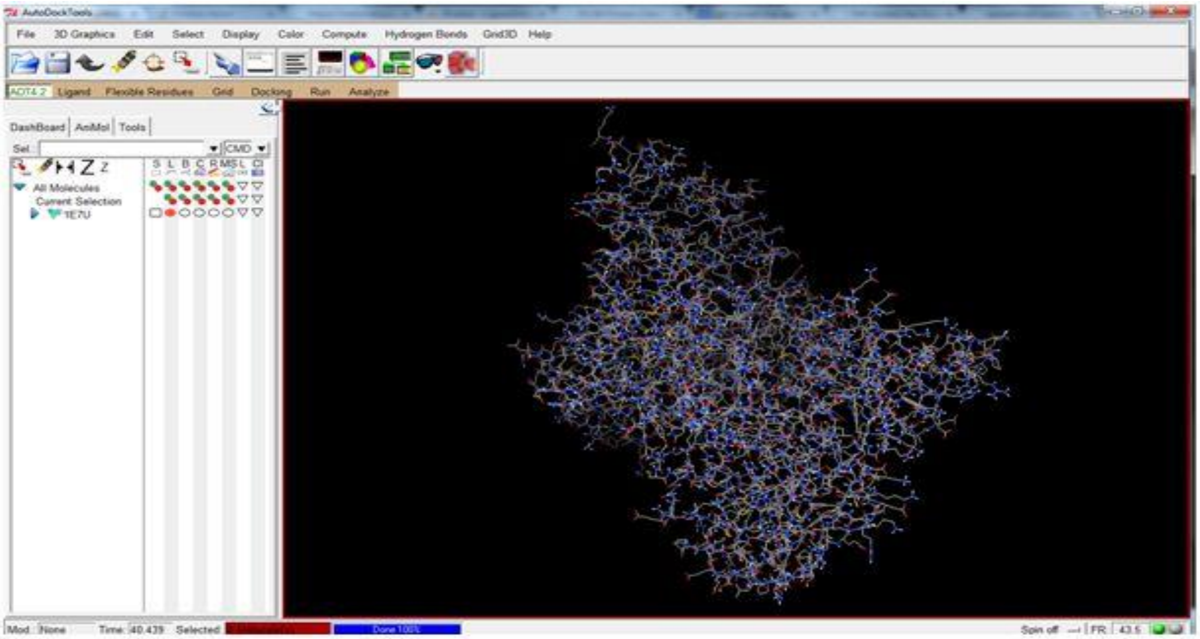
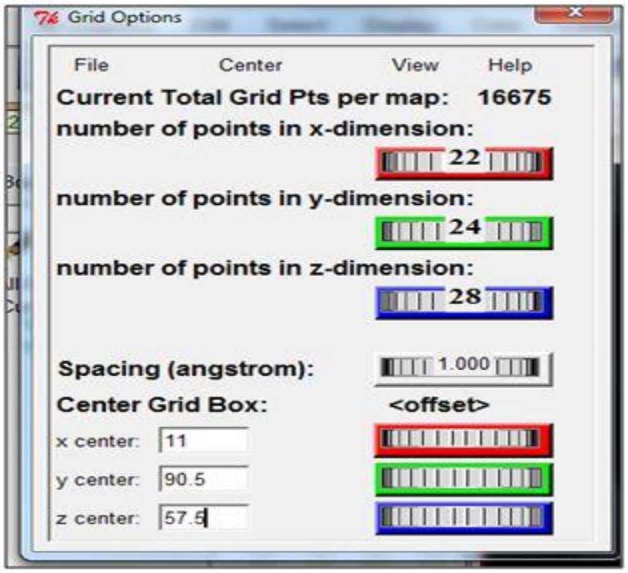


Figure 7 and 8: Selecting the receptor molecule for applying grid

8. By clicking on the respective molecule will display the details of non bonded atoms, non polar hydrogen atoms and non integral charge on the molecule. After that save the molecule in PDBQT format.(Figure 9)



9. To set grid parameters, go to “Grid” -> “Grid Box”  as shown in Figure 10. A “Grid Option” message appears which helps the user to change the grid point per map in all positions. It sets the 3D space for better binding conformation as shown in the figure. The maximum value that can be given by the Autogrid is 126.



Current Total Grid Pts per map: 16675

number of points in x-dimension: 22

number of points in y-dimension: 24

number of points in z-dimension: 28

Spacing (angstrom): 1.000

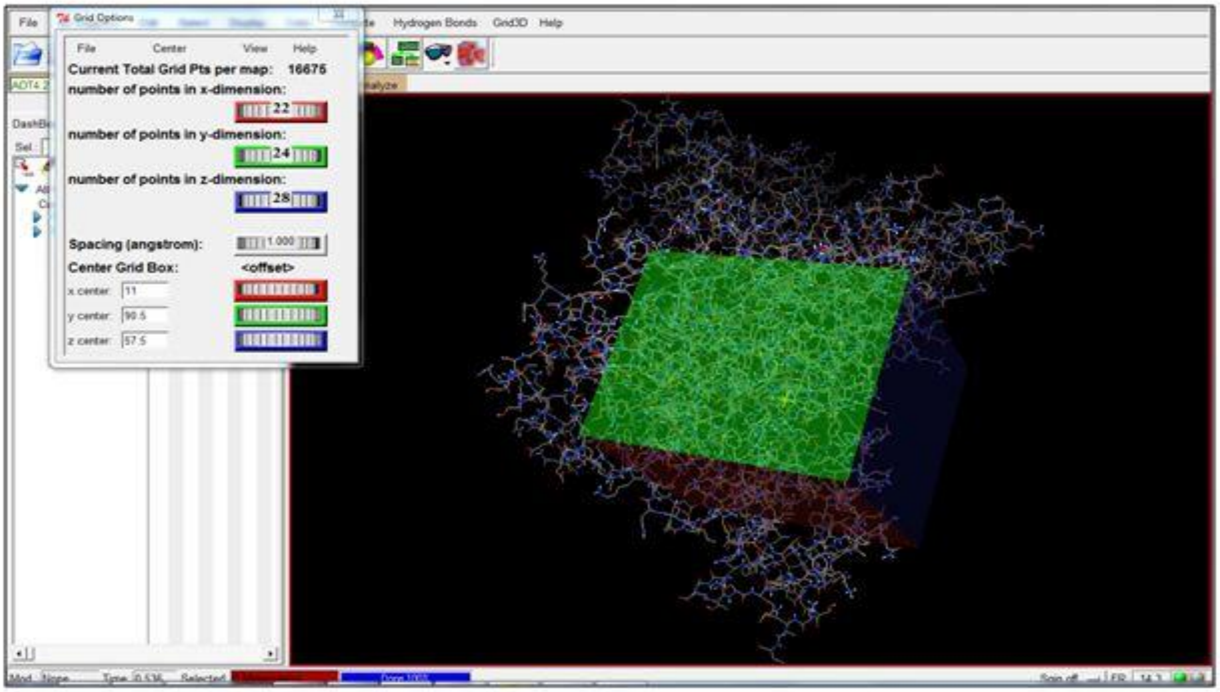
Center Grid Box: <offset>

x center: 11

y center: 90.5

z center: 57.5

Figure 10: Grid Option box



Current Total Grid Pts per map: 16675

number of points in x-dimension: 22

number of points in y-dimension: 24

number of points in z-dimension: 28

Spacing (angstrom): 1.000

Center Grid Box: <offset>

x center: 11

y center: 90.5

z center: 57.5

Figure 11: Assigning 3D space for better binding conformation

10. Next step is to prepare the ligand molecule for docking. Open the ligand miolecule by clicking on the "Ligand" option and select "Input" and click on "Open". Select the downloaded molecule and open it in the work space panel as shown in Figure 12.

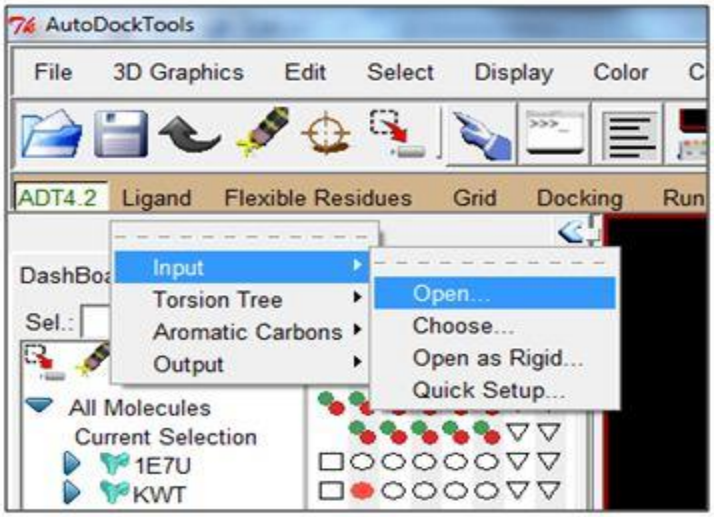


Figure 12: Reading ligand molecule

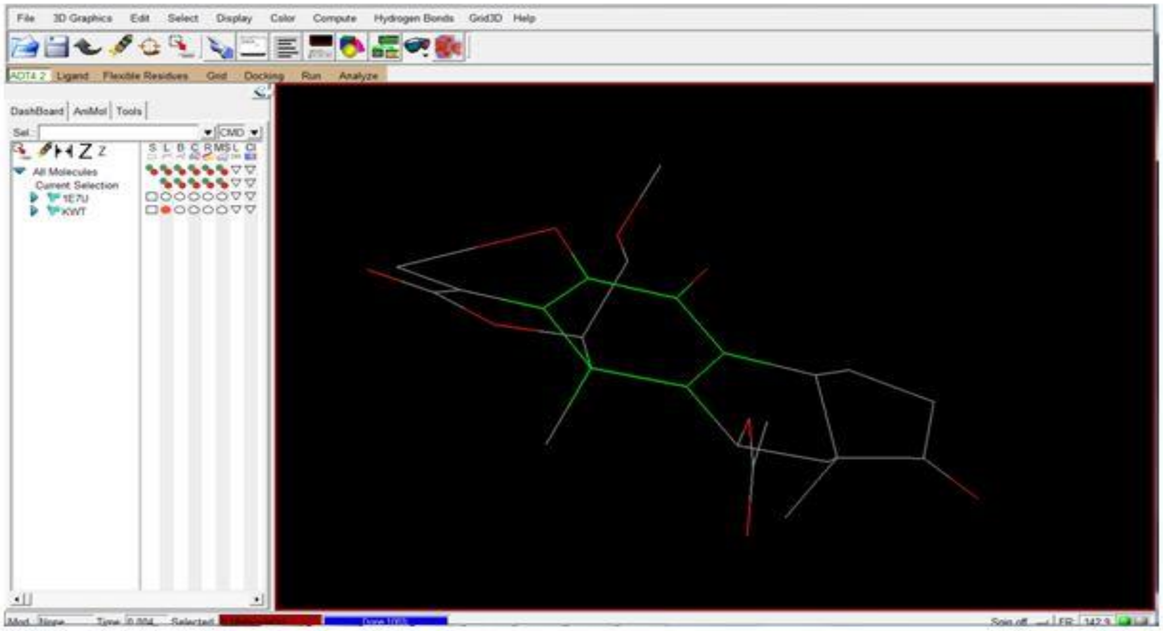


Figure 13: KWT opened in work space panel

11. The receptor molecule and ligand molecule can be viewed separately by clicking on dashboard which is displayed on the left side of the work space panel. By selecting the required molecule will display it in work space panel. The other options will enable us to view in other formats too  as shown in Figure 14.

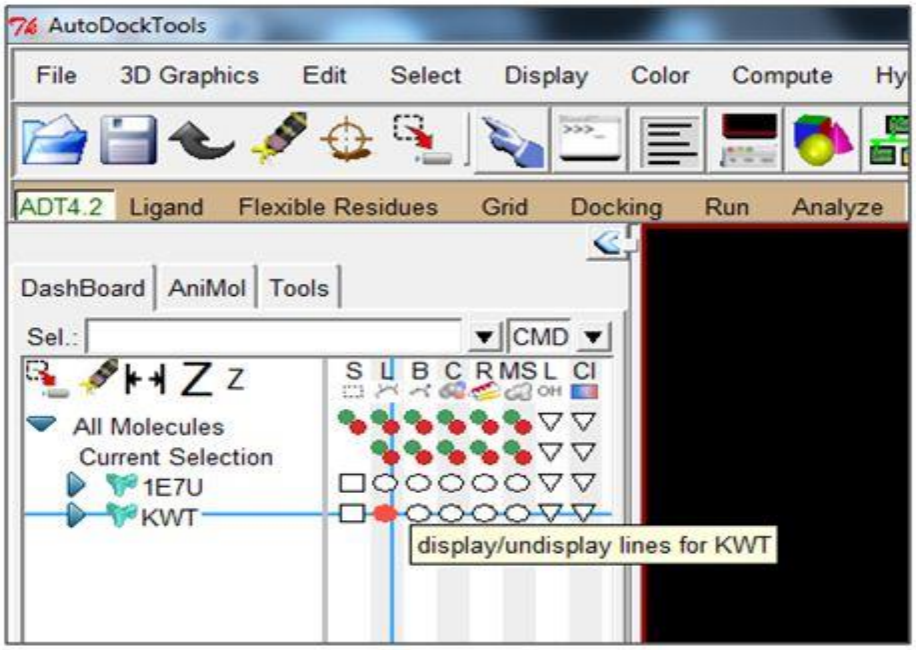


Figure 14: Dashboard with other options

12. To choose Torsions, click on the “ligand “ -> “Torsion Tree” ->”Choose Torsions” which will display the number of rotatable bonds. The rotatable bonds is displayed in green color, non-rotatable bonds in magenta color and unrotatable bonds in red color. To make a non - rotatable bond to rotatable, click on the bond itself  as shown in Figure 15.

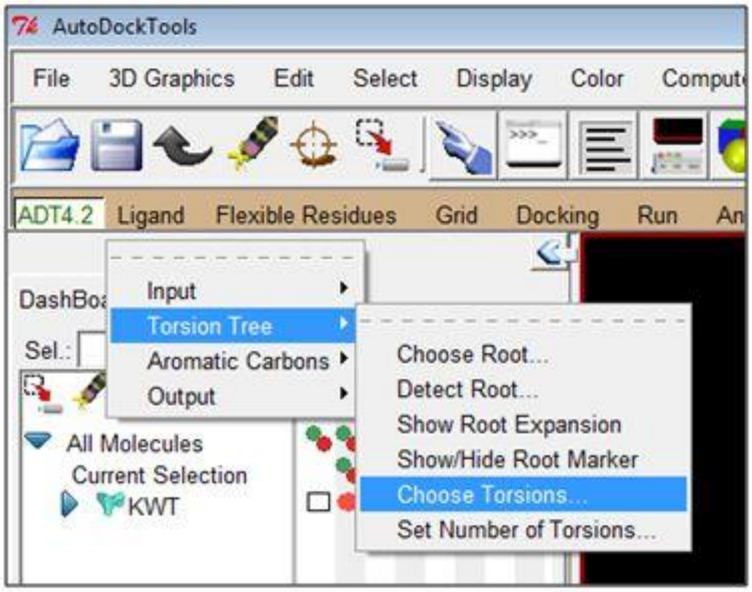


Figure 15: Selecting torsions to view rotatable bonds

13. The output can be saved inPDBQT format. For that click on the “Ligand” -> “Output” ->”Save as PDBQT” , so that it can be saved along with the receptor molecule in the same folder itself  as shown in Figure 16.

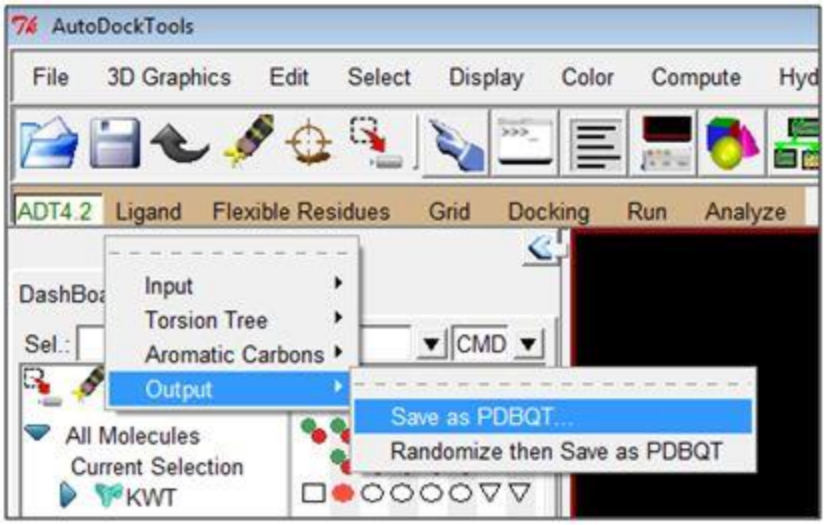
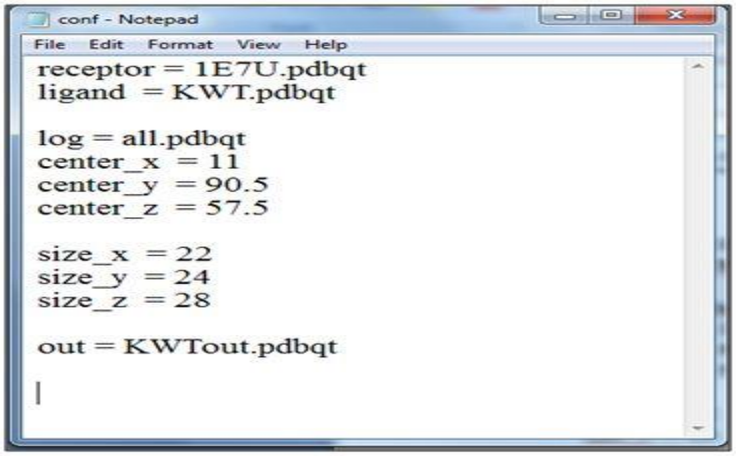


Figure 16: Output saved as PDBQT format

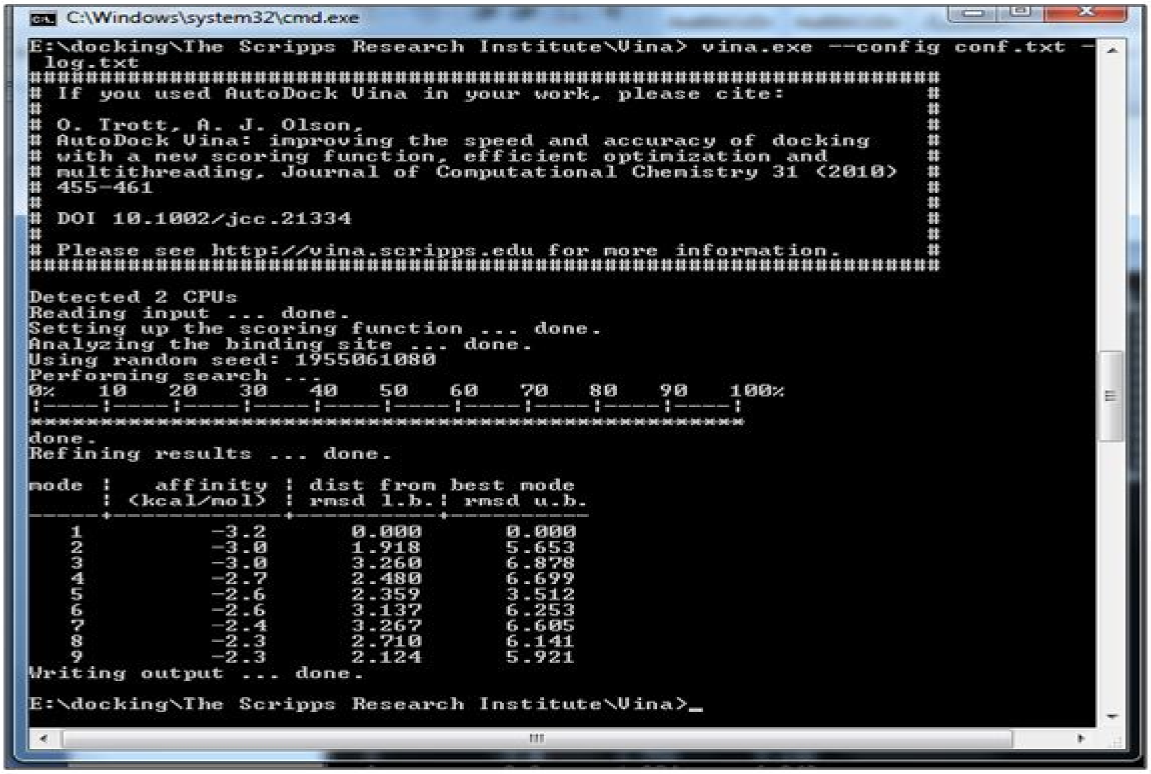
14. For running the Vina program, command prompt is used, “vina help” prints the different options necessary for running the program. It includes commands for receptor, ligand and so on. The configuration file is wriiten in a text document with the following format as shown in Figure 17.



```
receptor = 1E7U.pdbqt
ligand  = KWT.pdbqt

log = all.pdbqt
center_x  = 11
center_y  = 90.5
center_z  = 57.5

size_x  = 22
size_y  = 24
size_z  = 28

out = KWTout.pdbqt
```

Figure 17: Configuartion file saved as a text document

15. For running Autodock Vina, vina.exe --config conf.txt --log log.txt can be used as the script as shown in figure 14, which will create an outout file of the ligand and a log file along with other files. (Figure 18)



```
E:\docking\The Scripps Research Institute\Vina> vina.exe --config conf.txt -
 log.txt
#################################################################
# If you used AutoDock Vina in your work, please cite:          #
#                                                               #
# O. Trott, A. J. Olson,                                        #
# AutoDock Vina: improving the speed and accuracy of docking    #
# with a new scoring function, efficient optimization and       #
# multithreading, Journal of Computational Chemistry 31 (2010)  #
# 455-461                                                       #
#                                                               #
# DOI 10.1002/jcc.21334                                         #
#                                                               #
# Please see http://vina.scripps.edu for more information.      #
#################################################################

Detected 2 CPUs
Reading input ... done.
Setting up the scoring function ... done.
Analyzing the binding site ... done.
Using random seed: 1955061080
Performing search ...
0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
***************************************************
done.
Refining results ... done.

mode |   affinity | dist from best mode
     | (kcal/mol) | rmsd l.b.| rmsd u.b.
-----+------------+----------+----------
   1         -3.2      0.000      0.000
   2         -3.0      1.918      5.653
   3         -3.0      3.260      6.878
   4         -2.7      2.480      6.699
   5         -2.6      2.359      3.512
   6         -2.6      3.137      6.253
   7         -2.4      3.267      6.605
   8         -2.3      2.710      6.141
   9         -2.3      2.124      5.921
Writing output ... done.

E:\docking\The Scripps Research Institute\Vina>_
```

Figure 18: Output in Command prompt

**Reference:**

This Experiment uses: Trott, O. and Olson, A. J. (2010), AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. J. Comput. Chem., 31: 455–461. doi: 10.1002/jcc.21334, onlinelibrary.wiley.com/doi/10.1002/jcc.21334/abstract

**Webliography:**

1. Autodock Vina : vina.scripps.edu/

2. Autodock Vina Download : mgltools.scripps.edu/

3. metavo.metacentrum.cz/en/docs/aplikace/software/Autodock-vina.html

4. Autodock Vina Manual: vina.scripps.edu/manual.html

**Videos:**

1. Autodock Vina Tutorial: vina.scripps.edu/tutorial.html

# Molecular Dynamics and Simulation

**Sneha Murmu, U. B. Angadi and Sudhir Srivastava**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

Molecular dynamics (MD) simulation is a computational technique used to study the behavior of atoms and molecules over time. It is based on the laws of classical mechanics, which describe how particles move and interact with each other under the influence of forces. In an MD simulation, the positions, velocities, and accelerations of the atoms or molecules are calculated at each time step, and the system is evolved forward in time.

The basic principle of MD simulation is based on the integration of Newton's second law of motion, which states that the force acting on an object is proportional to its mass times its acceleration. In MD, the forces acting on each atom or particle are calculated using a force field, which describes the interactions between the atoms or particles in the system. The force field is typically based on empirical or theoretical models, which consider the van der Waals forces, electrostatic interactions, and bonded interactions such as covalent bonds, hydrogen bonds, and torsional angles. The motion of the atoms or particles is then simulated using numerical integration of Newton's equations of motion. This process involves calculating the position and velocity of each atom or particle at each time step, based on the forces acting on it, and then updating the forces based on the new positions and velocities.

MD simulations can provide detailed information on the structure, dynamics, and thermodynamics of a system. They can be used to study the behavior of molecules, proteins, and materials in different environments, such as solvents, membranes, or under mechanical stress. MD simulations can also be used to predict the behavior of systems under different conditions or to explore the effects of mutations or drug interactions on protein structures.

## Force Fields

Force fields are critical components of molecular dynamics (MD) simulations. They provide a mathematical description of the interatomic or intermolecular forces that govern the behavior of the simulated system. Force fields specify the potential energy and its corresponding force as a function of the coordinates of the atoms or molecules, which is used to calculate the motion of the system over time. They are mathematical models that include parameters for the bond stretching, bond bending, torsion, and non-bonded interactions between atoms (Figure 1). The accuracy of the force field determines the accuracy of the MD simulations.

There are two primary types of force fields used in molecular dynamics simulations: classical and quantum mechanical. Classical force fields are most commonly used in biomolecular simulations and are based on a set of mathematical functions and empirical parameters to describe the interactions between atoms. These force fields are computationally efficient and can simulate systems up to millions of atoms. Quantum mechanical force fields, on the other hand, consider the electronic structure of atoms and molecules and are computationally more intensive but can provide higher accuracy in describing the system.

A functional form for a force field (also called Potential Energy Function) that can be used to model single molecule or assemblies of atoms and / or molecules is as shown below:

$$\psi(\mathbf{r}^N) = \sum_{bonds} \frac{k_i}{2}(l_i - l_{i,0})^2 + \sum_{angles} \frac{k_i}{2}(\theta_i - \theta_{i,0})^2 + \sum_{torsions} \frac{V_n}{2}(1 + cos(n\omega - \gamma)) + \sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(4\varepsilon_{ij}\left[\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{6} + \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}}\right]\right) \quad \text{… Equation 1}$$

$\psi(r^N)$ denotes the potential energy, which is a function of the positions (r) of N particles (usually atoms).

The first term in the Equation 1 models the interaction between pairs of bonded atoms, here modelled by a harmonic potential that gives the increase in energy as the bond length $l_i$ deviates from the reference value $l_{i,0}$. The second component is a summation over all valence angles in the molecule, modelled using a harmonic potential. A valence bond angle is the angle formed between three atoms A-B-C in which A and C are both bonded to B. The third component is a torsional potential that models how the energy changes as a bond rotates. The fourth component is the non-bonded term. It is calculated between all pairs of atoms (i and j) that are in different molecules or that are the same molecule but separated by at least three bonds (1, n relationship where n ≥ 4). In a simple force field, the non-bonded term is modelled using a Coulomb potential term for electrostatic interactions and a Lennard-Jones potential for van der Waals interactions.

The first three are the components of covalent (or bonded) contribution and the last one is the component of non-covalent (or non-bonded) contribution.

**A simple form of the above equation:**

A potential function or force field calculates the molecular system's potential energy (E) in a given conformation as a sum of individual energy terms,

$E = E_{\text{Covalent}} + E_{\text{Non-covalent}}$ … Equation 2

where, $E_{\text{Covalent}} = E_{\text{bond}} + E_{\text{angle}} + E_{\text{dihedral}}$

$E_{\text{Non-covalent}} = E_{\text{electrostatic}} + E_{\text{van der Waals}}$

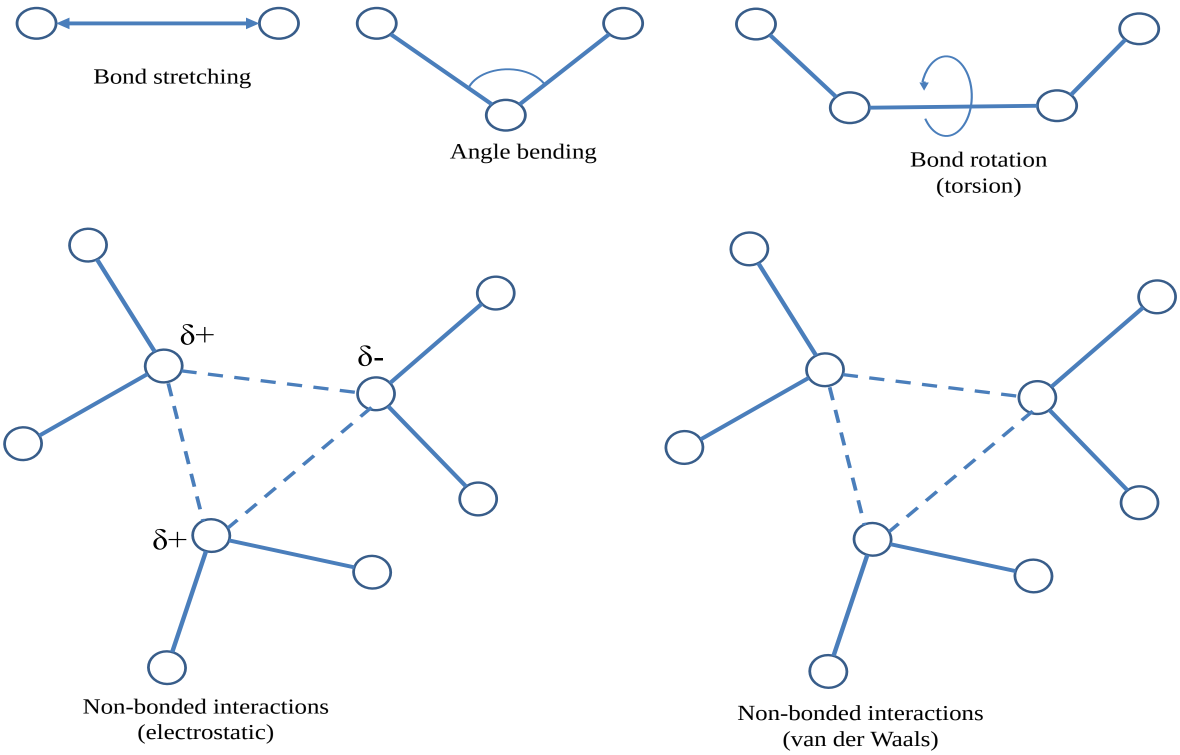


**Figure 1:** Schematic representation of bonded (upper row) and non-bonded (lower row) components contributing to a molecular mechanics force field.

There are several different force fields that have been developed over the years, each with its own strengths and limitations. Here are some examples:

CHARMM (Brooks et al., 2009): The Chemistry at Harvard Macromolecular Mechanics (CHARMM) force field is widely used for biomolecular simulations. It includes parameters for all of the major types of interactions, including covalent bonds, angles, dihedrals, van der Waals forces, and electrostatics. It is known for its accuracy in reproducing protein structures and dynamics.

AMBER (Case et al., 2010): The Assisted Model Building with Energy Refinement (AMBER) force field is also widely used in biomolecular simulations. It includes parameters for bond stretching, bond bending, torsion, and non-bonded interactions, and is known for its accuracy in reproducing experimental structures and dynamics.

OPLS (Damm et al., 1997): The Optimized Potentials for Liquid Simulations (OPLS) force field was originally developed for liquid simulations, but has also been used in biomolecular simulations. It includes parameters for bond stretching, bond bending, torsion, and non-bonded interactions, and is known for its accuracy in reproducing thermodynamic properties of liquids.

GROMOS (Scott et al., 1999): The Groningen Molecular Simulation (GROMOS) force field is widely used in simulations of small molecules and peptides. It includes parameters for bond stretching, bond bending, torsion, and non-bonded interactions, and is known for its accuracy in reproducing thermodynamic properties of small molecules.

### Conclusion

In summary, the principle of molecular dynamics simulation is based on the integration of classical mechanics, which involves calculating the positions, velocities, and forces of all atoms or particles in a system as a function of time. MD simulations can provide detailed information on the structure, dynamics, and thermodynamics of a system and can be used to study a wide range of molecular and material systems.

**************************************Practical**************************************

The purpose of this hands-on is to provide an introduction to the fundamental commands needed to set up, run, and analyze MD simulations using a suitable simulation tool. GROMACS which is one of the most popular Molecular Dynamics (MD) simulation software, will be used for the practical session. Before starting with the steps of typical MD simulation, let us have a quick look on how to install GROMACS in linux (here, Ubuntu).

### Installation

To install GROMACS, we need the following software installed on our system:

i. C & C++ Compiler which comes built-in with Ubuntu.
ii. CMake – A linux software to make binaries
iii. BuildEssential – It is a reference for all the packages needed to compile a package.
iv. FFTW Library: a library used by Gromacs to compute discrete Fourier transform

v. DeRegressionTest Package

Following are commands to install above mentioned pre-requisites:

```
sudo apt-get update

sudo apt-get upgrade

sudo apt-get install cmake

sudo apt-get install build-essential

wget http://gerrit.gromacs.org/download/regressiontests-5.1.1.tar.gz

tar xvzf regressiontests-5.1.1.tar.gz

sudo apt-get install libfftw3-dev

wget ftp://ftp.gromacs.org/pub/gromacs/gromacs-5.1.1.tar.gz

tar xvzf gromacs-5.1.1.tar.gz

cd gromacs-5.1.1/

mkdir build

cd build

sudo cmake .. -DGMX_BUILD_OWN_FFTW=OFF -DREGRESSIONTEST_DOWNLOAD=OFF -DCMAKE_C_COMPILER=gcc -DREGRESSIONTEST_DOWNLOAD=ON

make

make check

sudo make install

source /usr/local/gromacs/bin/GMXRC
```

If the execution of above commands was successful, the installation is complete. You may check the version of your Gromacs with a command to make sure the installation finished as expected.

gmx pdb2gmx --versionource /usr/local/gromacs/bin/GMXRC

**MD Simulation protoco**l

Following steps are involved in simulating a protein structure.

- Create initial state

i. Generate topology of protein
ii. Add box and solvation to the system
iii. Add ions to the solved system

- Introduction to the interaction potentials

iv. Energy minimization

- Predict how the particles move

v. Equilibration of system
vi. MD Production run

Now, we will see how to perform each step in more details. For the purpose of demonstrating simulation of protein, a small protein structure of ubiquitin (PDB code 1UBQ) was downloaded from RCSB PDB.

**1. Generate topology**

The obtained protein structure must be checked for the following things:

- Remove the water molecules if present
- Non-standard residues like heteroatoms must be removed
- Residues with missing atoms must be fixed beforehand

If water molecules are present, we can simply use the grep command to search for "HOH" in the PDB file and then remove them. The following command can be used for removing water molecules:

*grep -v HOH 1UBQ.pdb > 1UBQ_clean.pdb*

The next step is to use the pdb2gmx module of GROMACS. The pdb2gmx module generates three files:

□ The topology for the molecule.

□ A position restraint file.

□ A post-processed structure file.

The topology (topol.top by default) contains all the information necessary to define the molecule within a simulation. This information includes nonbonded parameters as well as bonded parameters. The following command was used to execute pdb2gmx:

```
gmx pdb2gmx -f 1UBQ_clean.pdb -o 1UBQ_processed.gro -water spce
```

The structure is processed by pdb2gmx, and we are prompted to choose a force field. We will use the all-atom OPLS force field, so ‘15’ was typed at the command prompt

The force field will contain the information that will be written to the topology.

## 2. Solvation

To simulate proteins and other molecules we need to define the box dimensions around the protein and fill in the box with solvent. The box was defined using the following command:

```
gmx editconf -f 1UBQ_processed.gro -o 1UBQ_newbox.gro -c -d 1.0 -bt cubic
```

-c : centers the protein in the box

-d 1.0 : places the protein at least 1.0 nm from the box edge

-bt cubic : The box type is defined as a cube

Specifying a solute-box distance of 1.0 nm will mean that there are at least 2.0 nm between any two periodic images of a protein. This distance will be sufficient for just about any cut off scheme commonly used in simulations.

The box is filled with solvent (water) by using the command below:

```
gmx solvate -cp 1UBQ_newbox.gro -cs spc216.gro -o 1UBQ_solv.gro -p topol.top
```

-cp : this parameter takes as input the configuration of the protein which is contained in the output file obtained from the previous step

-cs : configuration of the solvent is part of the standard GROMACS installation. We are using spc216.gro, which is a generic equilibrated 3-point solvent model.

### 3. Adding Ions

Neutralizing a system is a practice carried out for obtaining correct electrostatic values during the simulation. This is done because under periodic boundary and using PME electrostatics - the system has to be neutral. Therefore, we are adding ions to neutralization purpose only. The tool for adding ions within GROMACS is called genion which reads through the topology and replace water molecules with the ions that the user specifies. The input is called a run input file, which has an extension of. tpr. The .tpr file contains all the parameters for all of the atoms in the system.ed by the GROMACS grompp module (GROMACS pre-processor).

Assemble .tpr file with the following command:

```
gmx grompp -f ions.mdp -c 1UBQ_solv.gro -p topol.top -o ions.tpr
```

Now we have an atomic-level description of our system in the binary file ions.tpr. We will pass this file to genion:

```
gmx genion -s ions.tpr -o 1UBQ_solv_ions.gro -p topol.top -pname NA -nname CL -neutral
```

-s : input file given as structure/state file (.tpr file)

-pname and -nname : define the positive and negative ion names

-neutral : add only the ions necessary to neutralize the net charge on the protein by adding the correct number of negative ions (in this case will add 8 Cl- ions to offset the +8 charge on the protein)

### 4. Energy minimization (EM)

EM is done to ensure there that the system has no steric clashes or inappropriate geometry.

First, we need to assemble structure, topology, and simulation parameters into a binary input file (.tpr file):

```
gmx grompp -f minim.mdp -c 1UBQ_solv_ions.gro -p topol.top -o em.tpr
```

Here, minim.mdp is the file containing information regarding molecular dynamics parameter. It is not inherently present in the GROMACS distribution; hence it needs to be created before the execution of above command. An mdp file contain following parameters,

```
; minim.mdp - used as input into grompp to generate em.tpr
; Parameters describing what to do, when to stop and what to save
integrator  = steep         ; Algorithm (steep = steepest descent minimization)
emtol      = 1000.0      ; Stop minimization when the maximum force < 1000.0 kJ/mol/nm
emstep      = 0.01          ; Minimization step size
nsteps      = 50000         ; Maximum number of (minimization) steps to perform

; Parameters describing how to find the neighbors of each atom and how to calculate
the interactions
nstlist         = 1         ; Frequency to update the neighbor list and long range forces
cutoff-scheme   = Verlet    ; Buffered neighbor searching
ns_type         = grid      ; Method to determine neighbor list (simple, grid)
coulombtype     = PME       ; Treatment of long range electrostatic interactions
rcoulomb        = 1.0       ; Short-range electrostatic cut-off
rvdw            = 1.0       ; Short-range Van der Waals cut-off
pbc             = xyz       ; Periodic Boundary Conditions in all 3 dimensions
```

Next, we have to invoke mdrun to carry out the EM:

```
gmx mdrun -v -deffnm em
```

The output em.edr file contains all of the energy terms that GROMACS collects during EM. We can analyze any .edr file using the GROMACS energy module:

```
gmx energy -f em.edr -o potential.xvg
```

At the prompt, type "10 0" to select Potential (10); zero (0) terminates input. The average of Epot is shown, and a file called "potential.xvg" is written. To plot this data, we need the Xmgrace plotting tool.

**5. Equilibration**

Since the objective of MD simulation is to study the dynamics of a particular system, we have to suit the *in-silico* environment of our simulation system as close as possible to the real system (e.g. experimental job in wet laboratory). Therefore, in equilibration step we optimize the temperature to 300K since we assumed that we do the experimental job at room temperature, and pressure value at 1 atm.

Equilibration will be carried out in two steps. First, an NVT (constant Number of atoms, Volume, and Temperature) simulation will be performed in order to bring the system to the target temperature. Second, an NPT (constant Number of atoms, Pressure, and Temperature) simulation will be performed to allow the system to find the correct density.

### 5. a) Temperature Equilibration

We will call grompp and mdrun just as we did at the EM step and run the following two commands:

```
gmx grompp -f nvt.mdp -c em.gro -r em.gro -p topol.top -o nvt.tpr
```

```
gmx mdrun -deffnm nvt
```

To analyze the temperature progression, using energy we use the command given below:

```
gmx energy -f nvt.edr -o temperature.xvg
```

Type "16 0" at the prompt to select the temperature of the system and exit and the temperature.xvg can be plotted by Xmgrace tool.

### 5. b) Pressure Equilibration

We had included the -t flag to include the checkpoint file from the NVT equilibration. This file contains all the necessary state variables to continue our simulation. To conserve the velocities produced during NVT, we must include this file. The coordinate file (nvt.gro) is the final output of the NVT simulation.

```
gmx grompp -f npt.mdp -c nvt.gro -r nvt.gro -t nvt.cpt -p topol.top -o npt.tpr
```

```
gmx mdrun -deffnm npt
```

To analyze the pressure progression, again by using energy:

```
gmx energy -f npt.edr -o pressure.xvg
```

Type "18 0" at the prompt to select the pressure of the system and exit. ‘pressure.xvg’ file will be created which can be plotted through Xmgrace.

To take a look at density as well using energy, we need to enter "24 0" at the prompt while running the following command:

```
gmx energy -f npt.edr -o density.xvg
```

**6. Production MD**

After running the two equilibration phases, the system is now well equilibrated at desired temperature and pressure. To run the production MD, we will make use of the checkpoint file to grompp and run a 1 ns MD simulation:

```
gmx grompp -f md.mdp -c npt.gro -t npt.cpt -p topol.top -o md_0_1.tpr
```

To execute mdrun:

```
gmx mdrun -deffnm md_0_1
```

**Analysis**

GROMACS comes equipped with many analysis tools, a complete list of which can be found in the manual. Here you will be exposed to a few useful analysis tools: 'rms', 'rmsf', and 'gyrate. But first, it is useful to learn how to process the trajectory file to only keep the components of interest. Use *trjconv*, which is a post-processing tool to strip out coordinates, correct for periodicity, or manually alter the trajectory (time units, frame frequency, etc). trjconv accounts for any periodicity in the system.

```
gmx trjconv -s md_0_1.tpr -f md_0_1.xtc -o md_0_1_noPBC.xtc -pbc mol –center
```

Select 1 ("Protein") as the group to be centered and 0 ("System") for output. Downstream analyses will be conducted on this "corrected" trajectory.

For checking the structural stability GROMACS has a built-in utility for RMSD calculations called rms. Root mean square deviation (RMSD) is used for measuring the difference between the backbones of a protein from its initial structural conformation to its final position. The command to plot rmsd graph is as follows:

```
gmx rms -s md_0_1.tpr -f md_0_1_noPBC.xtc -o rmsd.xvg -tu ns
```

When prompted choose 4 ("Backbone") for both the least-squares fit and the group for RMSD calculation.

The radius of gyration of a protein is a measure of its compactness. If a protein is stably folded, it will likely maintain a relatively steady value of Rg. If a protein unfolds, its Rg will change over time. The command to plot radius of gyration graph is as follows:

```
gmx gyrate -s md_0_1.tpr -f md_0_1_noPBC.xtc -o gyrate.xvg
```

When prompted choose group 1 (Protein) for analysis.

With this, we have now completed molecular dynamics simulation of a protein with GROMACS, and analyzed some of the results.

**References**

Van Der Spoel, D., Lindahl, E., Hess, B., Groenhof, G., Mark, A. E., & Berendsen, H. J. (2005). GROMACS: fast, flexible, and free. Journal of computational chemistry, 26(16), 1701-1718.

Case, D. A., Darden, T. A., Cheatham, T. E., Simmerling, C. L., Wang, J., Duke, R. E., ... & Kollman, P. A. (2008). Amber 10 (No. BOOK). University of California.

Brooks, B. R., Brooks III, C. L., Mackerell Jr, A. D., Nilsson, L., Petrella, R. J., Roux, B., ... & Karplus, M. (2009). CHARMM: the biomolecular simulation program. Journal of computational chemistry, 30(10), 1545-1614.

Damm, W., Frontera, A., Tirado–Rives, J., & Jorgensen, W. L. (1997). OPLS all-atom force field for carbohydrates. Journal of computational chemistry, 18(16), 1955-1970.

Scott, W. R., Hünenberger, P. H., Tironi, I. G., Mark, A. E., Billeter, S. R., Fennen, J., ... & Van Gunsteren, W. F. (1999). The GROMOS biomolecular simulation program package. The Journal of Physical Chemistry A, 103(19), 3596-3607.

# Online Resources of Proteomics Data

**K. K. Chaturvedi and Sudhir Srivastava**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## 1. Introduction

The field of proteomics is based on the systematic, large-scale characterization and analysis of the complete set of proteins produced by a given cell, tissue or organism under a defined set of conditions **[1]**. It covers the exploration of proteomes from the overall level of protein composition, structure, and activity, and is an important component of functional genomics. It was coined in 1994 by then Ph.D. student Marc Wilkins at Macquarie University.

After genomics and transcriptomics, proteomics is the next step in the study of biological systems, but it is more complex than genomics because an organism's genome is more or less constant, whereas proteomes differ from cell to cell and from time to time **[2]**.

Proteins also are subjected to a wide variety of chemical modifications after translation. The most common and widely studied post-translational modifications include phosphorylation and glycosylation. Many of these post-translational modifications are critical to the protein's function. In addition to phosphorylation and ubiquitination, proteins may be subjected to methylation, acetylation, glycosylation, oxidation, and nitrosylation. Some proteins undergo all these modifications, often in time-dependent combinations **[3]**. Proteomics generally refers to the large-scale experimental analysis of proteins and proteomes, but often refers specifically to protein purification and mass spectrometry.

## 2. Mass spectrometry data format

Mass spectrometry (MS) has recently emerged as a major discovery tool in the life sciences. This analytical technique is used to analyze the molecular composition of a biological sample by ionizing the sample or analyze molecules and then measuring the mass-to-charge ratios of the resulting ions. The data from an MS experiment consist of mass spectra that are used to identify, characterize, and quantify the abundance of the molecules of interest **[4]**.

Many open, XML-based data formats have recently been developed by the Trans-Proteomic Pipeline at the Institute for Systems Biology for global data exchange to facilitate integration and comparison of data stored in various databases. These data formats are described here.

### 2.1 JCAMP-DX

This format was one of the earliest attempts to supply a standardized file format for in mass spectrometry based data exchange. It was initially developed for infrared spectrometry. It is an ASCII based format and therefore not very compact although it provides standards for file compression **[5].**

### 2.2 mzData

mzData was the first attempt by the Proteomics Standards Initiative (PSI) from the Human Proteome Organization (HUPO) to create a standardized format for Mass Spectrometry data, primarily as a data exchange and archive format **[6].** The mzData format is quite flexible as it uses controlled vocabulary extensively. This controlled vocabulary could be frequently updated to support new technologies, instruments, and methods of acquiring data while XML schema remains stable.

### 2.3 mzXML

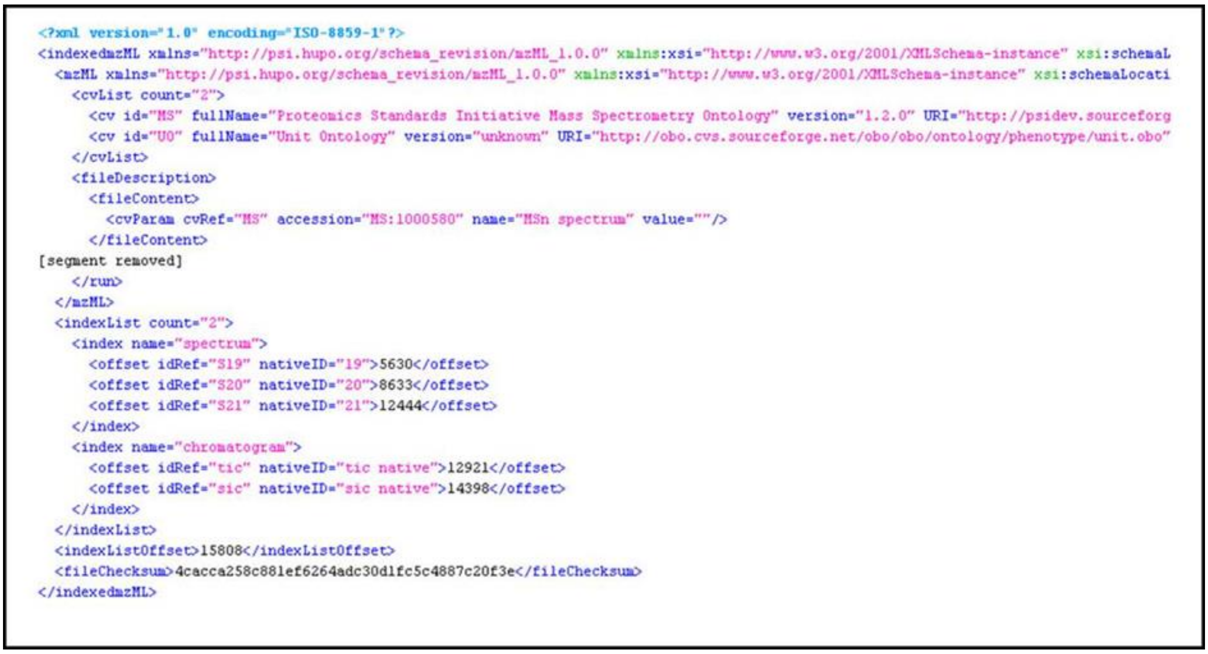
mzXML is a XML (eXtensible Markup Language) based common file format for proteomics mass spectrometric data **[7].** mzXML format was developed at the Institute for Systems Biology (ISB), primarily in order to streamline data processing software. mzXML have a very strict schema with most auxiliary information described in enumerated attributes.

### 2.4 YAFMS

YAFMS (Yet Another Format for Mass **S**pectrometry) is light, serverless, relational database format for proteomics data exchange purposes. Here file format is highly efficient in processing time, as well as in storage space. YAFMS allows data extraction and updates by writing simple SQL queries. Also, this format provides the flexibility to add tables that contain, processed data, deconvolution results, or even images used in publications.**[8].**

### 2.5 mzML

Both mzData and mzXML data formats used to represent same information, therefore HUPO-PSI, the SPC/ISB and instrument vendors made a joint effort to create a unified standard called mzML. It includes the best aspects of both mzData and mzXML data formats and replace these two formats. It was first published in 2008 **[9].**



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<indexedmzML xmlns="http://psi.hupo.org/schema_revision/mzML_1.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaL
  <mzML xmlns="http://psi.hupo.org/schema_revision/mzML_1.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocati
    <cvList count="2">
      <cv id="MS" fullName="Proteomics Standards Initiative Mass Spectrometry Ontology" version="1.2.0" URI="http://psidev.sourceforg
      <cv id="UO" fullName="Unit Ontology" version="unknown" URI="http://obo.cvs.sourceforge.net/obo/obo/ontology/phenotype/unit.obo"
    </cvList>
    <fileDescription>
      <fileContent>
        <cvParam cvRef="MS" accession="MS:1000580" name="MSn spectrum" value=""/>
      </fileContent>
[segment removed]
    </run>
  </mzML>
  <indexList count="2">
    <index name="spectrum">
      <offset idRef="S19" nativeID="19">5630</offset>
      <offset idRef="S20" nativeID="20">8633</offset>
      <offset idRef="S21" nativeID="21">12444</offset>
    </index>
    <index name="chromatogram">
      <offset idRef="tic" nativeID="tic native">12921</offset>
      <offset idRef="sic" nativeID="sic native">14398</offset>
    </index>
  </indexList>
  <indexListOffset>15808</indexListOffset>
  <fileChecksum>4cacca258c881ef6264adc30d1fc5c4887c20f3e</fileChecksum>
</indexedmzML>
```

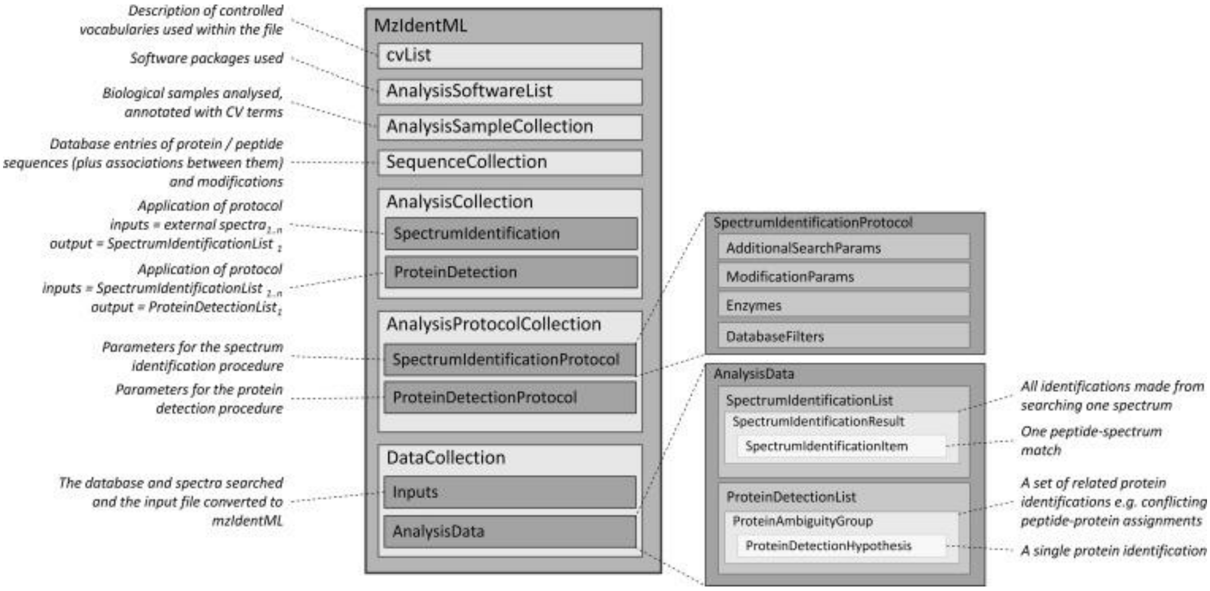
**Figure 1.** Example top and bottom of an mzML document with the middle segment removed for display purposes. The main part of the mzML document is contained within the <mzML></mzML> tags. It is wrapped within an <indexedmzML></indexedmzML> construct, which contains the random access index at the bottom. (Source: Deutsch, 2010)

## 2.6 mzAPI

It is common API (application program interface) proposed by a group of scientists to shift the burden of standards compliance to the instrument manufacturers' existing data access libraries **[10].**

## 2.7 mzIdentML

mzIdentML is one of the standards developed by the Proteomics Informatics working group of the PSI and the mzIdentML 1.0 specification was published in August 2009. The mzIdentML format is XML-based, meaning the files are XML files but with additional structure **[11].** It is a data standard that contains the peptide/protein identification information of a proteomics experiment, but not the quantification information.



Description of controlled vocabularies used within the file
Software packages used
Biological samples analysed, annotated with CV terms
Database entries of protein / peptide sequences (plus associations between them) and modifications
Application of protocol inputs = external spectra$_{1..n}$ output = SpectrumIdentificationList$_1$
Application of protocol inputs = SpectrumIdentificationList$_{1..n}$ output = ProteinDetectionList$_1$
Parameters for the spectrum identification procedure
Parameters for the protein detection procedure
The database and spectra searched and the input file converted to mzIdentML
MzIdentML
cvList
AnalysisSoftwareList
AnalysisSampleCollection
SequenceCollection
AnalysisCollection
SpectrumIdentification
ProteinDetection
AnalysisProtocolCollection
SpectrumIdentificationProtocol
ProteinDetectionProtocol
DataCollection
Inputs
AnalysisData
SpectrumIdentificationProtocol
AdditionalSearchParams
ModificationParams
Enzymes
DatabaseFilters
AnalysisData
SpectrumIdentificationList
SpectrumIdentificationResult
SpectrumIdentificationItem
ProteinDetectionList
ProteinAmbiguityGroup
ProteinDetectionHypothesis
All identifications made from searching one spectrum
One peptide-spectrum match
A set of related protein identifications e.g. conflicting peptide-protein assignments
A single protein identification

**Figure 2.** Detailed structure of mzIdentML file format (Source: Jones et al. 2012)

## 2.8 mzTab

mzTab is also XML-based one of the standards developed by the Proteomics Informatics working group of the PSI **[12].** mzTab files can contain protein, peptide, and small molecule identifications together with experimental metadata and basic quantitative information.

## 2.9 imzML

The imzML standard is used to exchange data from mass spectrometry imaging in a standardized XML file. It splits experimental data into XML and spectral data in a binary file. Both files are linked by a universally unique identifier **[13].**

## 2.10 mzDB

mzDB consists of a standardized and portable server-less single-file database. It relies on the SQLite software library. An optimized 3D indexing approach is adopted, where the LC-MS coordinates (retention time and m/z), along with the precursor m/z for SWATH-MS data, are used to query the database for data extraction. In comparison with XML formats, mzDB saves storage space and improves access times. **[14].**

### 2.11 HDF5

Hierarchical Data Format (HDF) is a set of file formats (HDF4, HDF5) to store and organize large amounts of data ,developed at the National Center for Supercomputing Applications **[15].** The Hierarchical Data Format version 5 (HDF5), is an **open source file format** that supports large, complex, heterogeneous data. HDF5 uses a file directory like structure that allows you to organization of data within the file in many different structured ways.

### 2.12 Toffee

Toffee is an open file format for data-independent acquisition mass spectrometry. It supports HDF5 **[16].**

### 2.13 mzMLb

mzMLb also uses HDF5 backend for raw data storage. It, however, preserves the mzML XML data structure and stays compliant to the existing standard **[17].**

### 2.14 mz5

It is mzML based format, but uses HDF5 backend for reducing storage space requirements and improved read/write speed **[18].**

## 3. Databases for raw data storage, data submission and analysis

Here, we are providing details of important web resources for MS-based proteomics:

**Table 1:** Various MS-based proteomics databases.

| Database Name | Facilities | Link |
|---|---|---|
| PRIDE | Data storage and data submission | http://www.ebi.ac.uk/pride/archive |
| PeptideAtlas | Data storage, data submission and data analysis | http://www.peptideatlas.org |
| Human Proteinpedia | Data storage, data submission and data analysis | http://www.humanproteinpedia.org |
| ProteomicsDB | Data submission and data analysis | https://www.proteomicsdb.org/ |
| MassIVE | Access public datasets, reanalyze spectra, submit data, results comparison and search identifications | https://massive.ucsd.edu |

### 3.1 Human Proteinpedia

Human Proteinpedia is a resource to integrate, store, and share proteomic data **[19].** It is a platform for collecting human proteomic data using a distributed annotation system, which allows the research community to contribute protein annotations. It also provides a panorama of the human proteome.

### 3.2 Proteomics IDEntification (PRIDE) database

The PRoteomics IDEntifications (PRIDE) database (https://www.ebi.ac.uk/pride/) is the world's largest data repository of mass spectrometry-based proteomics data. PRIDE is one of the founding members of the global ProteomeXchange (PX) consortium and an ELIXIR core data resource. It has played an important role in the nascent Human Proteome Project (HPP) **[20]**. It provides a standardised way for submitting mass spectrometry based proteomics data to public-domain repositories and provides access to published experimental data **[21]**.

#### PRIDE resources

#### PRIDE Archive

User can search Original mass spec projects used by PRIDE Peptidome project in the PRIDE Archive. The PRIDE PRoteomics IDEntifications (PRIDE) Archive database is a centralized, standards compliant, public data repository for mass spectrometry proteomics data, including protein and peptide identifications and the corresponding expression values, post-translational modifications and supporting mass spectra evidence (both as raw data and peak list files). Datasets are submitted to ProteomeXchange via PRIDE and are handled by expert bio-curators. All PRIDE public datasets can also be searched in ProteomeCentral, the portal for all ProteomeXchange datasets.

#### PRIDE Archive Spectra

PRIDE Spectra Archive provides direct access to the submitted mass spectra by either selecting peptide or USI Universal Spectrum Identifiers. The USI is multi-part key identifier for identifying mass spectra contained in public data repositories, primarily focused on proteomics).

#### PRIDE Spectrum Libraries

These spectrum libraries are derived from the PRIDE Cluster results. They contain the consensus spectra of all reliable clusters generated from the public experiments in PRIDE Archive. Therefore, they also contain consensus spectra from labelled experiments as well as a wider array of species. These spectral libaries can be read and processed by most spectral libary search tools.

#### PRIDE TOOLS

#### PRIDE Submission Tool

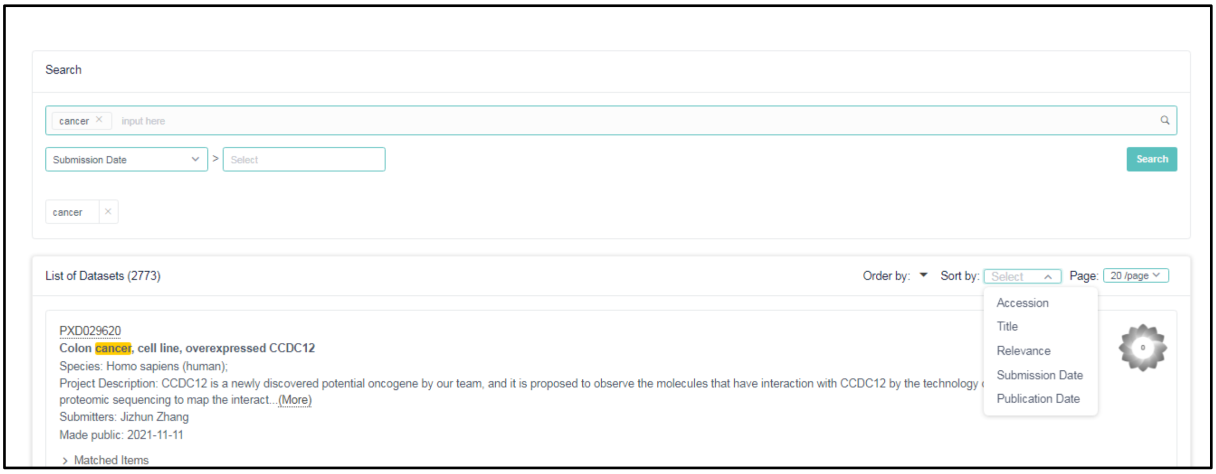
PRIDE Submission Tool enables the user to submit proteomics datasets to PRIDE Archive. **[21]**. Complete Process of Submission of dataset to PRIDE Archive explained in case study.

#### PRIDE Inspector Tool Suite

The PRIDE Inspector Toolsuite is the main tool used to review and download the proteomics data from PRIDE Archive. The stand-alone tool provides different panels or view focuses on a particular aspect of the data **[22]**.

### Dataset search in PRIDE Archive

The search can support dataset identifiers ProteomeXchange dataset (PXD) identifiers or PRIDE assay/experiment numbers, PubMed identifiers, sample details (e.g. organisms, organism part, diseases), instruments, post-translational modifications and any word/phrase included in the title or description of a given dataset.



PXD029620
**Colon cancer, cell line, overexpressed CCDC12**
Species: Homo sapiens (human);
Project Description: CCDC12 is a newly discovered potential oncogene by our team, and it is proposed to observe the molecules that have interaction with CCDC12 by the technology
proteomic sequencing to map the interact...(More)
Submitters: Jizhun Zhang
Made public: 2021-11-11
› Matched Items

**Figure 3.** Search results using dataset identifiers, PubMed identifiers, or sample details (Source: https://www.ebi.ac.uk/pride).

The search terms will be matched against the records in PRIDE Archive and a list of dataset summaries, if any records match, will be shown as a result. A project summary includes the following default information:

1. Project accession (dataset identifier)
2. Project Title
3. Project description (shortened)
4. Organism
5. Project publication date

### Filtering Search Results

Through filtering we can ensure that some information will be present in our search results. The available filters types are: Organism, Organism Part, Diseases, Modification, Instrument, Experiment Type etc.

### 3.3 ProteomicsDB

ProteomicsDB is an in-memory database that was originally created to explore massive amounts of quantitative human mass spectrometry-based proteomics data. ProteomicsDB offers a wide range of data types and use cases across disciplines, including tandem mass spectra, peptide identifications, and peptide proteotypicity values, which can be used as starting points for developing focused mass spectrometry assays [23].

It allows the real-time exploration and retrieval of protein abundance values across different tissues, cell lines, and body fluids via interactive expression heat maps and body maps.

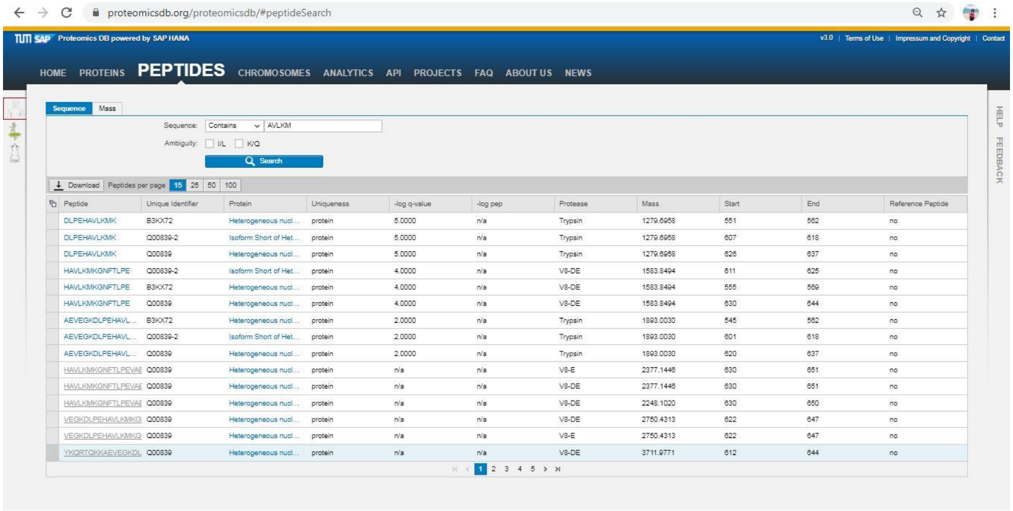
ProteomicsDB supports multiple use cases across different disciplines and covering a wide range of data  e.g. tandem mass spectra, peptide identifications etc. Both experimental and reference spectra can be used for assay development and to validate the identification of so far unobserved proteins [23].

### ProteomicsDB Tools

Data upload: Users can temporarily upload their expression profiles and optionally normalize them to the data stored in ProteomicsDB. Data stored in such sessions are available via ODATA (https://www.odata.org) services within ProteomicsDB and will ultimately allow the integration into any existing analytical pipeline. The first use case which can be highlighted is the comparison of custom expression data to expression data stored in ProteomicsDB. For this to successful, the normalization features available upon upload. By uploading an expression dataset, heat maps will be generated. The heat map allows interactive visualization of expression patterns of multiple groups of proteins.

### Searching Peptides/Proteins

User can enter peptide sequence or mass and will get a list of peptides containing the sequence. Information such as unique identifier, protein name, protease mass, start position and end position can be seen.



proteomicsdb.org/proteomicsdb/#peptideSearch
TUM SAP Proteomics DB powered by SAP HANA
v3.0 | Terms of Use | Impressum and Copyright | Contact
HOME PROTEINS PEPTIDES CHROMOSOMES ANALYTICS API PROJECTS FAQ ABOUT US NEWS
Sequence Mass
HELP FEEDBACK

Sequence: Contains AVLKM

Ambiguity: □ I/L □ K/Q

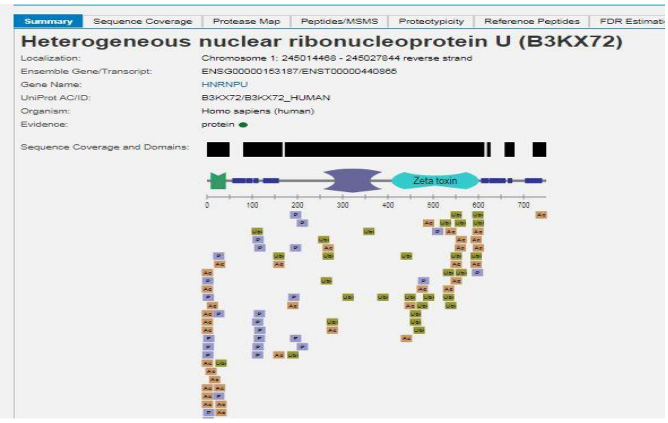
Search

Download Peptides per page 15 25 50 100

| Peptide | Unique Identifier | Protein | Uniqueness | -log q-value | -log pep | Protease | Mass | Start | End | Reference Peptide |
|---|---|---|---|---|---|---|---|---|---|---|
| DLPEHAVLKMK | B3KX72 | Heterogeneous nucl… | protein | 5.0000 | n/a | Trypsin | 1279.6958 | 551 | 562 | no |
| DLPEHAVLKMK | Q00839-2 | Isoform Short of Het… | protein | 5.0000 | n/a | Trypsin | 1279.6958 | 607 | 618 | no |
| DLPEHAVLKMK | Q00839 | Heterogeneous nucl… | protein | 5.0000 | n/a | Trypsin | 1279.6958 | 626 | 637 | no |
| HAVLKMKGNFTLPE | Q00839-2 | Isoform Short of Het… | protein | 4.0000 | n/a | V8-DE | 1583.8494 | 611 | 625 | no |
| HAVLKMKGNFTLPE | B3KX72 | Heterogeneous nucl… | protein | 4.0000 | n/a | V8-DE | 1583.8494 | 555 | 569 | no |
| HAVLKMKGNFTLPE | Q00839 | Heterogeneous nucl… | protein | 4.0000 | n/a | V8-DE | 1583.8494 | 630 | 644 | no |
| AEVEGKDLPEHAVL… | B3KX72 | Heterogeneous nucl… | protein | 2.0000 | n/a | Trypsin | 1893.0030 | 545 | 562 | no |
| AEVEGKDLPEHAVL… | Q00839-2 | Isoform Short of Het… | protein | 2.0000 | n/a | Trypsin | 1893.0030 | 601 | 618 | no |
| AEVEGKDLPEHAVL… | Q00839 | Heterogeneous nucl… | protein | 2.0000 | n/a | Trypsin | 1893.0030 | 620 | 637 | no |
| HAVLKMKGNFTLPEVAE | Q00839 | Heterogeneous nucl… | protein | n/a | n/a | V8-E | 2377.1446 | 630 | 651 | no |
| HAVLKMKGNFTLPEVAE | Q00839 | Heterogeneous nucl… | protein | n/a | n/a | V8-DE | 2377.1446 | 630 | 651 | no |
| HAVLKMKGNFTLPEVAE | Q00839 | Heterogeneous nucl… | protein | n/a | n/a | V8-DE | 2248.1020 | 630 | 650 | no |
| VEGKDLPEHAVLKMKG | Q00839 | Heterogeneous nucl… | protein | n/a | n/a | V8-DE | 2750.4313 | 622 | 647 | no |
| VEGKDLPEHAVLKMKG | Q00839 | Heterogeneous nucl… | protein | n/a | n/a | V8-E | 2750.4313 | 622 | 647 | no |
| YKQRTQKKAEVEGKDL | Q00839 | Heterogeneous nucl… | protein | n/a | n/a | V8-DE | 3711.9771 | 612 | 644 | no |

**Figure 4.** Peptide details (Source: https://www.proteomicsdb.org).

Detailed information can be seen by clicking the protein from the list such as localization, gene name organism name etc.



Summary Sequence Coverage Protease Map Peptides/MSMS Proteotypicity Reference Peptides FDR Estimati

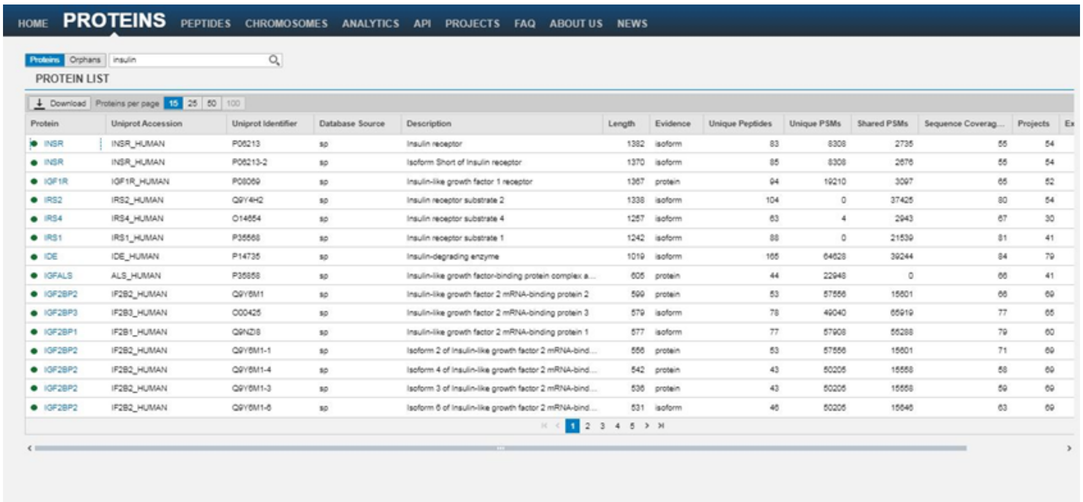
Heterogeneous nuclear ribonucleoprotein U (B3KX72)

| | |
|---|---|
| Localization: | Chromosome 1: 245014468 - 245027844 reverse strand |
| Ensemble Gene/Transcript: | ENSG00000153187/ENST00000440865 |
| Gene Name: | HNRNPU |
| UniProt AC/ID: | B3KX72/B3KX72_HUMAN |
| Organism: | Homo sapiens (human) |
| Evidence: | protein ● |
| Sequence Coverage and Domains: | |

Zeta toxin
0
100
200
300
400
500
600
700
P
Ubi
Ac

**Figure 5.** Protein summary details (Source: https://www.proteomicsdb.org).

Proteins can be searched by name to get the information about accession number, identifier, description about protein, etc.



| Protein | Uniprot Accession | Uniprot Identifier | Database Source | Description | Length | Evidence | Unique Peptides | Unique PSMs | Shared PSMs | Sequence Coverag... | Projects | Ex |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ● INSR | INSR_HUMAN | P06213 | sp | Insulin receptor | 1382 | isoform | 83 | 8308 | 2735 | 55 | 54 | |
| ● INSR | INSR_HUMAN | P06213-2 | sp | Isoform Short of Insulin receptor | 1370 | isoform | 85 | 8308 | 2676 | 55 | 54 | |
| ● IGF1R | IGF1R_HUMAN | P08069 | sp | Insulin-like growth factor 1 receptor | 1367 | protein | 94 | 19210 | 3097 | 65 | 52 | |
| ● IRS2 | IRS2_HUMAN | Q9Y4H2 | sp | Insulin receptor substrate 2 | 1338 | isoform | 104 | 0 | 37425 | 80 | 54 | |
| ● IRS4 | IRS4_HUMAN | O14654 | sp | Insulin receptor substrate 4 | 1257 | isoform | 63 | 4 | 2943 | 67 | 30 | |
| ● IRS1 | IRS1_HUMAN | P35568 | sp | Insulin receptor substrate 1 | 1242 | isoform | 88 | 0 | 21539 | 81 | 41 | |
| ● IDE | IDE_HUMAN | P14735 | sp | Insulin-degrading enzyme | 1019 | isoform | 165 | 64628 | 39244 | 84 | 79 | |
| ● IGFALS | ALS_HUMAN | P35858 | sp | Insulin-like growth factor-binding protein complex a... | 605 | protein | 44 | 22948 | 0 | 66 | 41 | |
| ● IGF2BP2 | IF2B2_HUMAN | Q9Y6M1 | sp | Insulin-like growth factor 2 mRNA-binding protein 2 | 599 | protein | 53 | 57556 | 15601 | 66 | 69 | |
| ● IGF2BP3 | IF2B3_HUMAN | O00425 | sp | Insulin-like growth factor 2 mRNA-binding protein 3 | 579 | isoform | 78 | 49040 | 65919 | 77 | 65 | |
| ● IGF2BP1 | IF2B1_HUMAN | Q9NZI8 | sp | Insulin-like growth factor 2 mRNA-binding protein 1 | 577 | isoform | 77 | 57908 | 56288 | 79 | 60 | |
| ● IGF2BP2 | IF2B2_HUMAN | Q9Y6M1-1 | sp | Isoform 2 of Insulin-like growth factor 2 mRNA-bind... | 556 | protein | 53 | 57556 | 15601 | 71 | 69 | |
| ● IGF2BP2 | IF2B2_HUMAN | Q9Y6M1-4 | sp | Isoform 4 of Insulin-like growth factor 2 mRNA-bind... | 542 | protein | 43 | 50205 | 15558 | 58 | 69 | |
| ● IGF2BP2 | IF2B2_HUMAN | Q9Y6M1-3 | sp | Isoform 3 of Insulin-like growth factor 2 mRNA-bind... | 536 | protein | 43 | 50205 | 15558 | 59 | 69 | |
| ● IGF2BP2 | IF2B2_HUMAN | Q9Y6M1-6 | sp | Isoform 6 of Insulin-like growth factor 2 mRNA-bind... | 531 | isoform | 46 | 50205 | 15646 | 63 | 69 | |

**Figure 6.** Protein details (Source: https://www.proteomicsdb.org).

## 3.4 MassIVE

MassIVE (Mass Spectrometry Interactive Virtual Environment) is a community resource developed by the NIH-funded Center for Computational Mass Spectrometry which ease exchange of mass spectrometry data. Various datasets present in the database can be downloaded, submitted identifications can be searched and result comparison can be done **[24]**.

### MassIVE Tools

### Access Public Datasets

User can Browse publically available datasets or search by dataset metadata (e.g., species, PI, etc.). Datasets are available for download as well as for online browsing of submitted identifications (for complete datasets). Dataset owners can also add missing/requested files, update metadata and add publications to their datasets. Registered users can comment on datasets so others in the community can see updates or find pointers to new analyses of the data.

### Submit Data

User can submit data to share with the community as a MassIVE dataset. Reviewer access credentials and ProteomeXchange identifiers can be requested to meet publication guidelines of proteomics datasets. Workflows are also available to convert raw files (mass spectrometry data) to the open mzML format and to convert from common tab-separated formats (identifications data) into the open mzTab format.

### Search Identifications

All submitted identifications can be searched in complete datasets and dataset reanalyses. Over 300 million peptide-spectrum matches submitted with at most 1% false discovery rate are accessible through this simple interface to search for peptides, proteins and post-translational modifications.

### Reanalyze Spectra

Online MassIVE workflows can be used to reanalyse public datasets for analysis of mass spectrometry data: MSGF+ database search, MSPLIT spectral library search, MODa open modification search, Maestro spectral networks search and MSPLIT-DIA for search of data-independent acquisition (DIA) spectra.

### Result Comparison

User can compare identification results between datasets or against any reanalyses of public data. Venn diagrams are used to compare results at the level of protein, peptide and spectrum identifications. Agreements, disagreements and unique identifications can be interactively inspected for assessment of quality of identifications.

### Share Reanalysis

User can share dataset reanalysis results with the community or reveal new identifications with novel algorithms / analysis pipelines or challenge previously submitted identifications with alternative interpretations for the same data.

### Protein Explorer

Translated evidence and sequence coverage of nearly every human protein, can be explored, as defined by systematic reanalysis of 31 terabytes of public data from >20,000 LC/MS runs and including over 1 million synthetic peptide spectra. Interactive exploration of protein evidence includes coverage maps, functional sites, and full provenance and dataset mapping of every identified peptide.MassIVE Knowledge Base

Browse the community big data derived MassIVE Knowledge Base (MassIVE-KB) peptide spectral libraries. Distilled from 31TB of human proteomics HCD data. Users can peak at the inside of these libraries, browse the source data, and track full provenance of analysis tasks that created these libraries.

### MassIVE quant

MassIVEquant is an extension of the Mass Spectrometry Interactive Virtual Environment (MassIVE) to provide the opportunity for large-scale deposition of data from quantitative mass spectrometry-based proteomic experiments. MassIVEquant is compatible with all mass spectrometry data acquisition types and all computational analysis tools. For each dataset, MassIVEquant systematically stores the raw experimental data, the annotations of the experimental design, the scripts (or descriptions) of every step of the quantitative analysis workflow, and the intermediate input and output files. A branch structure enables MassIVE.quant to store and view alternative reanalyses of the same dataset with various combinations of methods and tools in a way which allows the user to inspect, reproduce or modify any component of the workflow, beginning with well-defined intermediate files. MassIVEquant supports infrastructure to fully automate analysis workflow, or to store, and to browse the intermediate results.

### CoronaMassKB

CoronaMassKB is an open-data community resource for sharing of mass spectrometry data and (re)analysis results for all experiments pertinent to the global SARS-CoV-2 pandemic.

CoronaMassKB is designed for the rapid exchange of data and results among the global community of scientists working towards understanding the biology of SARS-CoV-2/COVID19 and thus accelerating the emergence of effective responses to this global pandemic.

### 3.5 PeptideAtlas

PeptideAtlas is a database that stores various formats of output files and metadata from MS-based experiments , it also allows users to submit raw data. These raw data are periodically analyzed for identification and statistical analysis purposes. The results are made available back to the researchers by web-based presentation systems. PeptideAtlas can help plan targeted proteomics experiments, improve genome annotation, and support data mining projects **[25].** PeptideAtlas is a multi-organism, publicly accessible compendium of peptides identified in a large set of tandem mass spectrometry proteomics experiments **[26]**. Mass spectrometer output files are collected for human, mouse, yeast, and several other organisms, and searched using the latest search engines and protein sequences. All results of sequence and spectral library searching are subsequently processed through the Trans Proteomic Pipeline to derive a probability of correct identification for all results in a uniform manner to insure a high quality database, along with false discovery rates at the whole atlas level. Results may be queried and browsed at the PeptideAtlas web site. The raw data, search results, and full builds can also be downloaded for other uses.

#### PeptideAtlas tools

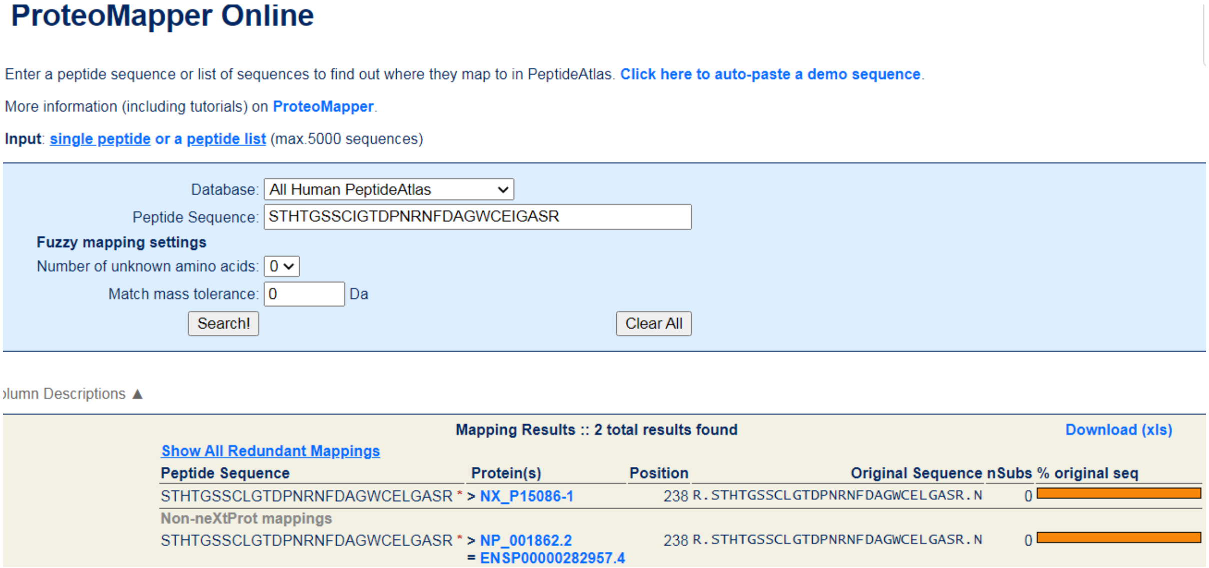
#### PeptideAtlas Tiered Human Integrated Search Proteome (THISP)

There is an automated system that integrates all of the major sources of human protein sequences into a collection of search databases in order to provide well-defined, comprehensive, and often updated human proteomics MS/MS search databases. These databases are tiered into several levels (given below) of complexity from which researchers may choose depending on the goal of the experiment and the data processing resources available **[26]**. On the first of every month, all protein lists are pulled down from their original sources. If any of them have changed, they are integrated and released.

#### ProteoMapper Online

ProteoMapper is a software which efficiently maps observed sequences to all possible variants. There are two components to ProteoMapper: an indexer, and a mapper. A protein sequence database in either FASTA or PEFF format must first be indexed by the indexer. Once the index is built, the mapper can quickly and efficiently map all locations of the input peptide sequence(s) to the proteome. Multiple parallel indices are supported, and input can be in the form of a pepXML file, a simple text file with peptide sequences, or a single sequence via the command-line. There are also options to map using wildcards as well as fuzzy mapping (where one or more amino acids and their positions within the peptide sequence are unknown). User can enter a peptide sequence or list of sequences (maximum upto 5000 sequences) and can select one of the database (All human Peptide Atlas, Yeast, *C. elegans* and Mouse database) **[26]**.

Here, in example below we have taken a peptide sequence *i.e.* STHTGSSCIGTDPNRNFDAGWCEIGASR and searched against All human Peptide Atlas database and found two proteins (NX_P15086-1 and NP_001862.2 ) along with their positions.



ProteoMapper Online

Enter a peptide sequence or list of sequences to find out where they map to in PeptideAtlas. Click here to auto-paste a demo sequence.

More information (including tutorials) on ProteoMapper.

Input: single peptide or a peptide list (max.5000 sequences)

Database: All Human PeptideAtlas

Peptide Sequence: STHTGSSCIGTDPNRNFDAGWCEIGASR

Fuzzy mapping settings

Number of unknown amino acids: 0

Match mass tolerance: 0 Da

Mapping Results :: 2 total results found

| Peptide Sequence | Protein(s) | Position | Original Sequence | nSubs | % original seq |
|---|---|---|---|---|---|
| STHTGSSCLGTDPNRNFDAGWCELGASR * | > NX_P15086-1 | 238 | R.STHTGSSCLGTDPNRNFDAGWCELGASR.N | 0 | |
| Non-neXtProt mappings | | | | | |
| STHTGSSCLGTDPNRNFDAGWCELGASR * | > NP_001862.2<br>= ENSP00000282957.4 | 238 | R.STHTGSSCLGTDPNRNFDAGWCELGASR.N | 0 | |

**Figure 7.** ProteoMapper showing result of mapping of a peptide sequence (Source: http://www.peptideatlas.org/map/).

## CASE STUDY

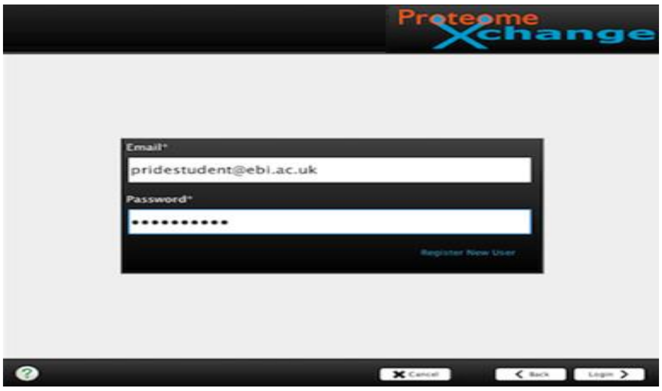
### PRIDE Submission Tool

The stand-alone ProteomeXchange (PX) Submission tool allows the researchers to perform the data submissions to PRIDE Archive.

Here we are describing all the steps to submit proteomics datasets to PRIDE Archive in brief:

### (i) Login Panel

The first step to submit a dataset to PRIDE Archive is to log into PRIDE using an existing account or register as a new user .



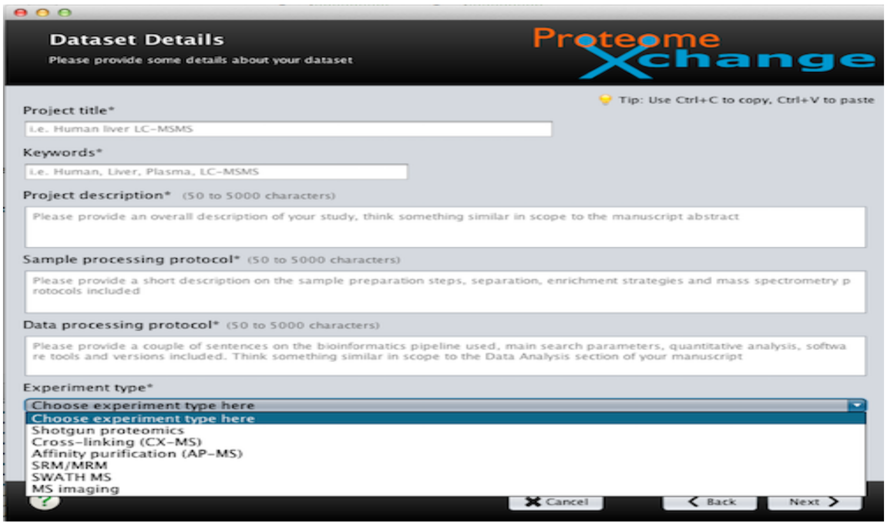
Email*: pridestudent@ebi.ac.uk

Password*: ••••••••••

**Figure 8.** Showing login window (Source: https://www.ebi.ac.uk/pride).

### (ii) Submission Details

Users are to provide some basic details about the uploaded dataset such as the title, a list of keywords (in a comma separated format), and a brief description of the dataset (similar to the abstract of the corresponding publication), a sample processing and a data processing protocol. Also, users have to pick a mass spectrometry experiment type from a drop-down menu (shotgun proteomics, SRM/MRM, CX-MS *etc*) **[24]**.



Dataset Details

Please provide some details about your dataset

Tip: Use Ctrl+C to copy, Ctrl+V to paste

Project title*

i.e. Human liver LC-MSMS

Keywords*

i.e. Human, Liver, Plasma, LC-MSMS

Project description* (50 to 5000 characters)

Please provide an overall description of your study, think something similar in scope to the manuscript abstract

Sample processing protocol* (50 to 5000 characters)

Please provide a short description on the sample preparation steps, separation, enrichment strategies and mass spectrometry p rotocols included

Data processing protocol* (50 to 5000 characters)

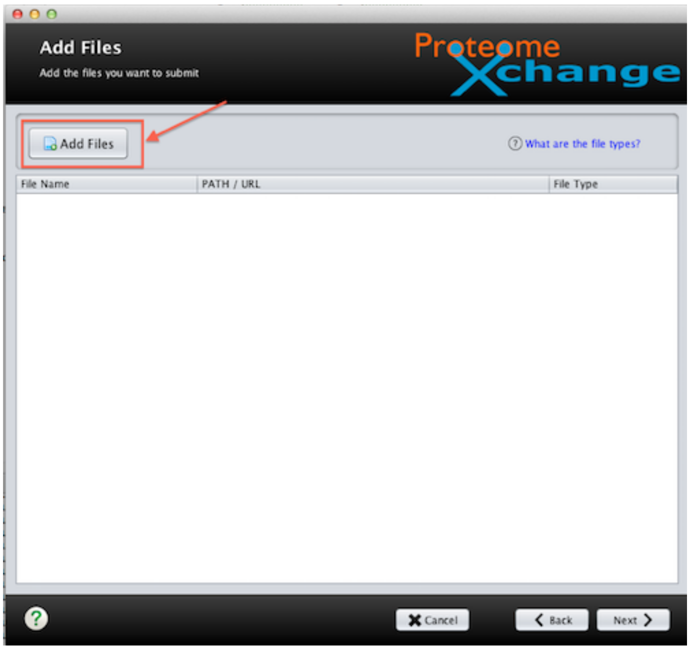
Please provide a couple of sentences on the bioinformatics pipeline used, main search parameters, quantitative analysis, softwa re tools and versions included. Think something similar in scope to the Data Analysis section of your manuscript

Experiment type*

**Figure 9.** Basic details about the uploaded dataset (Source: https://www.ebi.ac.uk/pride).

### (iii) Adding Files and assigning file types

In this stage, user should choose the files to be submitted. Files can be added by clicking on the highlighted button.



Add Files

Add the files you want to submit

**Figure 10.** Showing how to add files (Source: https://www.ebi.ac.uk/pride).

### File formats supported in PRIDE Archive:

**RESULT**: Standard file formats from HUPO-PSI to report peptide/protein identification and quantification results: mzIdentML and mzTab.

There are two relevant PSI file formats:

**mzIdentML**: mzIdentML is a data standard  that contains the peptide/protein identification information of a proteomics experiment, but not the quantification information

**mzTab**: mzTab files can contain protein, peptide, and small molecule identifications together with experimental metadata and basic quantitative information.

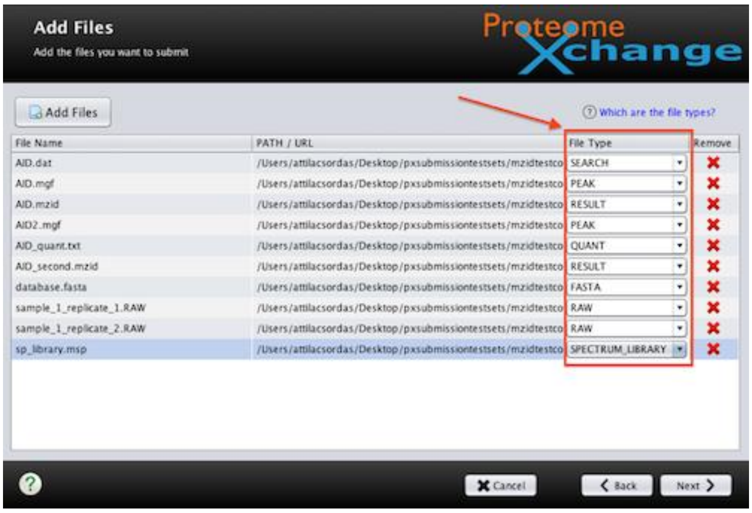
**RAW**: These are original proprietary files (e.g. Thermo RAW).

**SPECTRUM_LIBRARY**: Spectrum libraries used to perform spectrum search.

**PEAK**: The peak file contains the set of MS/MS peaks used for peptide/protein identification (e.g. mgf Mascot generic files).

**SEARCH**: Files from the software analysis tool (e.g. .dat from Mascot).

Submissions that provide RESULT files are called COMPLETE submissions. These files are the one, PRIDE ecosystem (resources, tools) is able to read, write and transform. When a Complete submission is performed using mzIdentML or files mzTab files (identification files), the dataset should contains at least one 'PEAK' list associated with the identification file. mzIdentML only contain the identified peptides/proteins and the corresponding spectra For Quantitative Complete experiments, users should use mzTab files. mzTab is a data standard which represent both identification and quantification data **[24]**.



Add Files

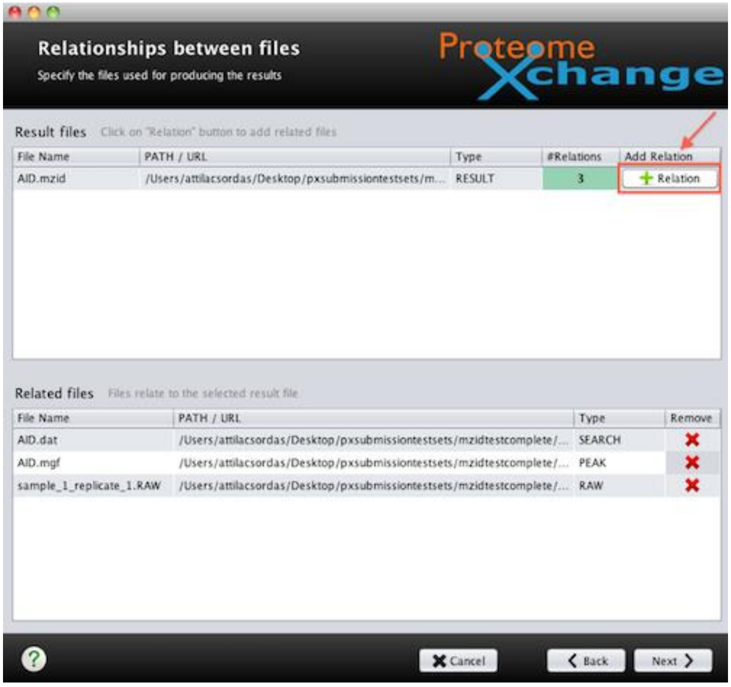
Add the files you want to submit

| File Name | PATH / URL | File Type | Remove |
|---|---|---|---|
| AID.dat | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | SEARCH | ✗ |
| AID.mgf | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | PEAK | ✗ |
| AID.mzid | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | RESULT | ✗ |
| AID2.mgf | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | PEAK | ✗ |
| AID_quant.txt | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | QUANT | ✗ |
| AID_second.mzid | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | RESULT | ✗ |
| database.fasta | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | FASTA | ✗ |
| sample_1_replicate_1.RAW | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | RAW | ✗ |
| sample_1_replicate_2.RAW | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | RAW | ✗ |
| sp_library.msp | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestco | SPECTRUM_LIBRARY | ✗ |

**Figure 11.** Showing different supported file formats (Source: https://www.ebi.ac.uk/pride).

### (iv) Assign the relationships between the submitted files

This mapping step consists of assigning the relations between the 'RESULT' files and the other types of files included in the submission, for example, which 'RAW' (mandatory), 'PEAK' (mandatory for mzIdentML and mzTab), 'SEARCH', 'QUANT', 'FASTA', 'SPECTRUM_LIBRARY', 'GEL' or 'OTHER' files can be linked to a given 'RESULT' file or are associated with it **[24]**.


Relationships between files
Specify the files used for producing the results
ProteomeXchange
Cancel
Back
Next

Result files   Click on "Relation" button to add related files

| File Name | PATH / URL | Type | #Relations | Add Relation |
|---|---|---|---|---|
| AID.mzid | /Users/attilacsordas/Desktop/pxsubmissiontestsets/m... | RESULT | 3 | + Relation |

Related files   Files relate to the selected result file

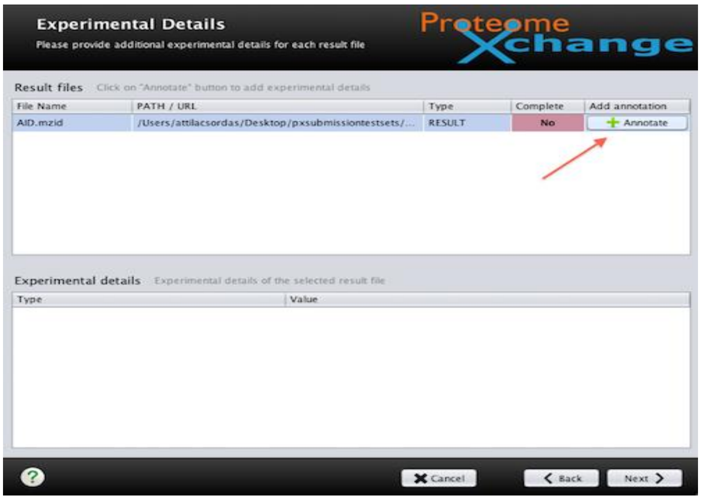
| File Name | PATH / URL | Type | Remove |
|---|---|---|---|
| AID.dat | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestcomplete/... | SEARCH | ✖ |
| AID.mgf | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestcomplete/... | PEAK | ✖ |
| sample_1_replicate_1.RAW | /Users/attilacsordas/Desktop/pxsubmissiontestsets/mzidtestcomplete/... | RAW | ✖ |

**Figure 12.** Mapping relation between result file and other files (Source: https://www.ebi.ac.uk/pride)

By default, the tool makes an attempt to generate the mapping between the 'RESULT' and the other, most importantly RAW' files. If there is one 'RESULT' file found then all the other files will be mapped to this file. But in case if multiple 'RESULT' files found then the tool maps other files with the same name prefix, but without the file extension, to the corresponding 'RESULT' file.

### (v) Additional submission metadata

Additional metadata need to be provided for each 'RESULT' file in the case of a Complete submission, both for mzTab or mzIdentML files.


Experimental Details
Please provide additional experimental details for each result file
ProteomeXchange
Cancel
Back
Next

Result files   Click on "Annotate" button to add experimental details

| File Name | PATH / URL | Type | Complete | Add annotation |
|---|---|---|---|---|
| AID.mzid | /Users/attilacsordas/Desktop/pxsubmissiontestsets/... | RESULT | No | + Annotate |

Experimental details   Experimental details of the selected result file

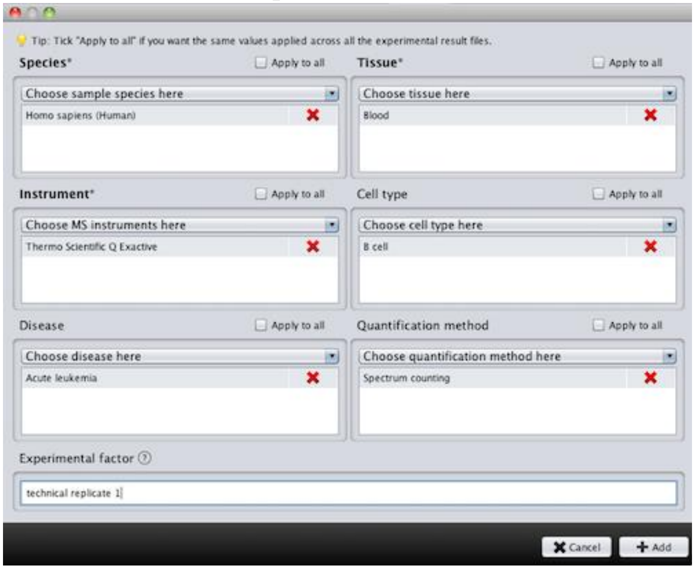
| Type | Value |
|---|---|

**Figure 13.** Annotation data is provided in case of complete submission (Source: https://www.ebi.ac.uk/pride).

User need to click 'Annotate' button for each 'RESULT' file. This information is usually imported automatically in the case of  mzTab file. For mzIdentML files, the information needs to be annotated manually.

The following additional metadata is Mandatory:

- Species: The species of the samples used in a given dataset.
- Tissue: Tissue ("not applicable" should be used in case other types of experiments are performed).
- Instrument information (mass spectrometer).



Tip: Tick "Apply to all" if you want the same values applied across all the experimental result files.

Species* □ Apply to all
Choose sample species here
Homo sapiens (Human) ✖

Tissue* □ Apply to all
Choose tissue here
Blood ✖

Instrument* □ Apply to all
Choose MS instruments here
Thermo Scientific Q Exactive ✖

Cell type □ Apply to all
Choose cell type here
B cell ✖

Disease □ Apply to all
Choose disease here
Acute leukemia ✖

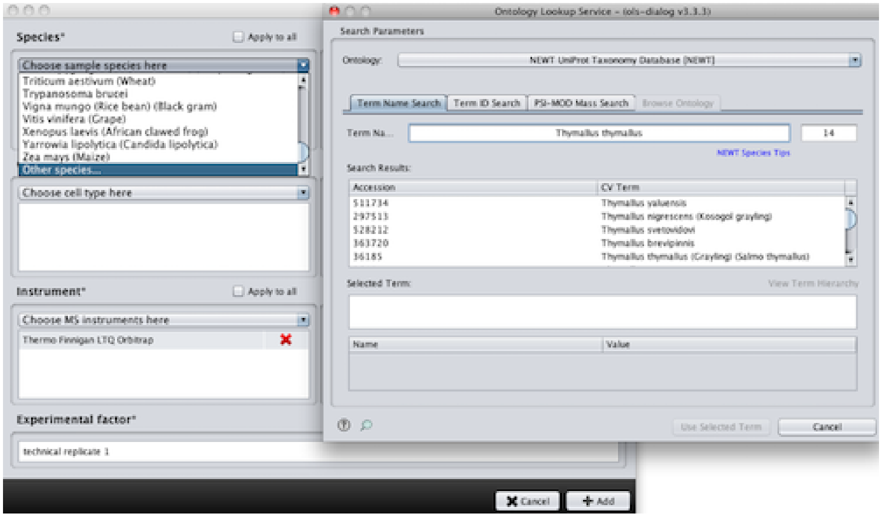
Quantification method □ Apply to all
Choose quantification method here
Spectrum counting ✖

Experimental factor
technical replicate 1

**Figure 14.** Metadata annotation with the drop-down menu (Source: https://www.ebi.ac.uk/pride).

Information should be provided using controlled vocabularies terms from a drop-down menu, providing information about the cell type, disease *and* quantification method etc.

In most cases the metadata annotation is available in the drop-down menu, since the elements of the drop-down menus have been selected based on frequency of these terms in existing datasets. However, sometimes the annotations you are looking for may not be available from the drop-down lists. If that's the case, we need to select the OLS (Ontology Lookup Service) panel and search for the corresponding annotation we want to provide. In the case of the more extensive searches we need to click on the "other" options on the bottom of the drop-down menu. For example, if we have samples coming from e.g. the fish Grayling (*Thymallus thymallus*) this species name is not available from the drop-down list menu. We have to click on other species and search for '*Thymallus thymallus*' in the OLS panel **[24]**.



Species* □ Apply to all
Choose sample species here
Triticum aestivum (Wheat)
Trypanosoma brucei
Vigna mungo (Rice bean) (Black gram)
Vitis vinifera (Grape)
Xenopus laevis (African clawed frog)
Yarrowia lipolytica (Candida lipolytica)
Zea mays (Maize)
Other species...

Choose cell type here

Instrument* □ Apply to all
Choose MS instruments here
Thermo Finnigan LTQ Orbitrap ✖

Experimental factor*
technical replicate 1

Ontology Lookup Service - (ols-dialog v3.3.3)

Search Parameters

Ontology: NEWT UniProt Taxonomy Database [NEWT]

Term Name Search | Term ID Search | PSI-MOD Mass Search | Browse Ontology

Term Na... Thymallus thymallus 14

NEWT Species Tips

Search Results:

| Accession | CV Term |
|---|---|
| 511734 | Thymallus yaluensis |
| 297513 | Thymallus nigrescens (Kosogol grayling) |
| 528212 | Thymallus svetovidovi |
| 363720 | Thymallus brevipinnis |
| 36185 | Thymallus thymallus (Grayling) (Salmo thymallus) |

Selected Term:

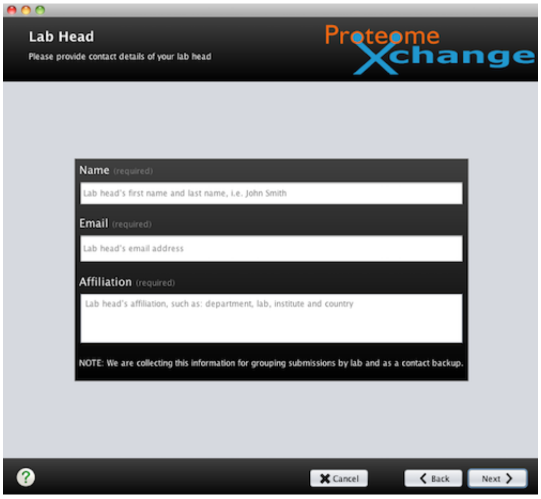
View Term Hierarchy

| Name | Value |
|---|---|

**Figure 15.** Ontology Lookup Service panel providing search for the corresponding annotation (Source: https://www.ebi.ac.uk/pride).

### (vi) Providing contact details for the Lab Head

Details of sender are to be provided for further reference.



Lab Head

Please provide contact details of your lab head

Name (required)

Lab head's first name and last name, i.e. John Smith

Email (required)

Lab head's email address

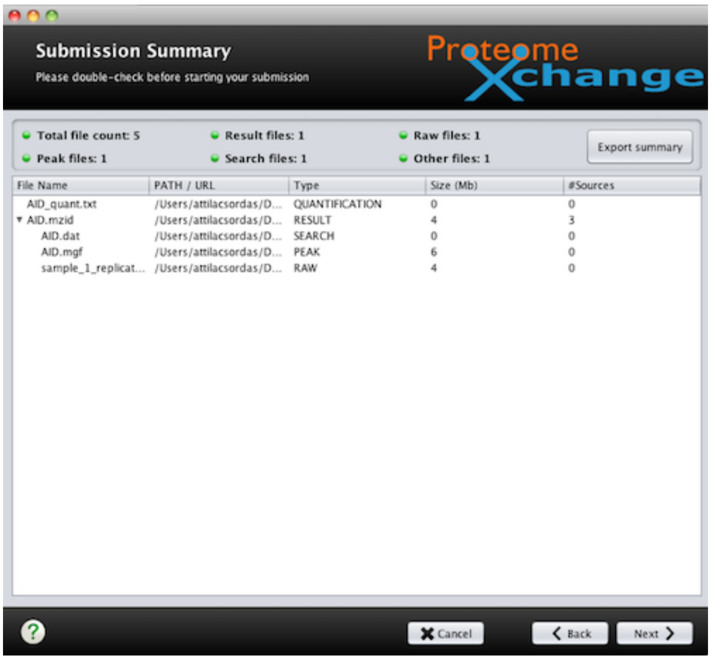
Affiliation (required)

Lab head's affiliation, such as: department, lab, institute and country

NOTE: We are collecting this information for grouping submissions by lab and as a contact backup.

**Figure 16.** Contact details of the sender (Source: https://www.ebi.ac.uk/pride).

This is the final step before the real file upload begins. Before moving on to the upload phase, double-check that the submission summary contains all of the essential files. An example of a mzIdentML-based 'Complete' submission is shown below.



Submission Summary

Please double-check before starting your submission

Total file count: 5
Result files: 1
Raw files: 1
Peak files: 1
Search files: 1
Other files: 1

| File Name | PATH / URL | Type | Size (Mb) | #Sources |
|---|---|---|---|---|
| AID_quant.txt | /Users/attilacsordas/D... | QUANTIFICATION | 0 | 0 |
| ▾ AID.mzid | /Users/attilacsordas/D... | RESULT | 4 | 3 |
| AID.dat | /Users/attilacsordas/D... | SEARCH | 0 | 0 |
| AID.mgf | /Users/attilacsordas/D... | PEAK | 6 | 0 |
| sample_1_replicat... | /Users/attilacsordas/D... | RAW | 4 | 0 |

**Figure 17.** Submission summary mzIdentML based complete submission (Source: https://www.ebi.ac.uk/pride).

### (vii) Uploading all files

Uploading all  files to PRIDE (as part of ProteomeXchange) is the final step. Once the upload is complete, you will receive an email confirming that all of your files have been successfully uploaded and are awaiting validation. By default, dataset will be made publicly available after  manuscript has been accepted, or when submitter instructs to do so or  there is acceptance notification from some journals.

## 4. Discussion

In this chapter, we have listed some commonly used and important proteomics databases which have proved to be very useful for molecular biologists. These resources which include original raw data and the accompanying results have led to high-throughput proteomics research and  large-scale genome annotation efforts. In future, the exchanges of information and metadata between these repositories will become highly relevant, and therefore, the proteomics repositories need to evolve a focused approach to data accessibility among different repositories. Conversely, with the advent of new instruments, new techniques of sample preparation, data analytics, and new forms of data will be continuously generated. It is clear that the amount of data in the currently available repositories is just a small fraction of the actually-generated proteomics data that will eventually become available. Finally in order to benefit the research community, the resources will have to standardize the process and simplify the interface for data submission.

## References

1. Tyers, M. and Mann, M., 2003. From genomics to proteomics. Nature, 422(6928), pp.193-197.
2. Rappsilber, J. and Mann, M., 2002. What does it mean to identify a protein in proteomics?. Trends in biochemical sciences, 27(2), pp.74-78.
3. Khoury, G.A., Baliban, R.C. and Floudas, C.A., 2011. Proteome-wide post-translational modification statistics: frequency analysis and curation of the swiss-prot database. Scientific reports, 1(1), pp.1-5.
4. Martens, Lennart et al. “mzML--a community standard for mass spectrometry data.” Molecular & cellular proteomics : MCP vol. 10,1 (2011): R110.000133. doi:10.1074/mcp.R110.000133 1.1 Deutsch EW (December 2012). "File formats commonly used in mass spectrometry proteomics". Molecular & Cellular Proteomics. 11 (12): 1612–21. doi:10.1074/mcp.R112.019695. PMID 22956731.
5. McDonald, R.S. and Wilks Jr, P.A., 1988. JCAMP-DX: A standard form for exchange of infrared spectra in computer readable form. Applied Spectroscopy, 42(1), pp.151-162.
6. Fischer, B., Neumann, S. and Gatto, L., 2013. A Parser for mzXML, mzData and mzML files.
7. Qing, H. and Xiang, F., 2007. Application of mzXML in mass spectrum data sharing. Computers and Applied Chemistry, 24(12), p.1635.
8. Shah, A.R., Monroe, M.E., Shi, Y., LaMarche, B., Crowell, K., Slysz, G.S., Anderson, G.A. and Smith, R.D., Next generation data exchange format for mass spectrometry.
9. Martens, L., Chambers, M., Sturm, M., Kessner, D., Levander, F., Shofstahl, J., Tang, W.H., Römpp, A., Neumann, S., Pizarro, A.D. and Montecchi-Palazzi, L., 2011. mzML-a community standard for mass spectrometry data. Molecular & Cellular Proteomics, 10(1).
10. Askenazi, M., Parikh, J.R. and Marto, J.A., 2009. mzAPI: a new strategy for efficiently sharing mass spectrometry data. Nature methods, 6(4), pp.240-241.

11. Vizcaíno, J.A., Mayer, G., Perkins, S., Barsnes, H., Vaudel, M., Perez-Riverol, Y., Ternent, T., Uszkoreit, J., Eisenacher, M., Fischer, L. and Rappsilber, J., 2017. The mzIdentML data standard version 1.2, supporting advances in proteome informatics. Molecular & cellular proteomics, 16(7), pp.1275-1285.

12. Hoffmann, N., Rein, J., Sachsenberg, T., Hartler, J., Haug, K., Mayer, G., Alka, O., Dayalan, S., Pearce, J.T., Rocca-Serra, P. and Qi, D., 2019. mzTab-M: a data standard for sharing quantitative results in mass spectrometry metabolomics. Analytical chemistry, 91(5), pp.3302-3310.

13. Schramm, T., Hester, Z., Klinkert, I., Both, J.P., Heeren, R.M., Brunelle, A., Laprévote, O., Desbenoit, N., Robbe, M.F., Stoeckli, M. and Spengler, B., 2012. imzML—a common data format for the flexible exchange and processing of mass spectrometry imaging data. Journal of proteomics, 75(16), pp.5106-5110.

14. Bouyssie, D., Dubois, M., Nasso, S., de Peredo, A.G., Burlet-Schiltz, O., Aebersold, R. and Monsarrat, B., 2015. mzDB: a file format using multiple indexing strategies for the efficient analysis of large LC-MS/MS and SWATH-MS data sets. Molecular & Cellular Proteomics, 14(3), pp.771-781.

15. Askenazi, M., Ben Hamidane, H. and Graumann, J., 2017. The arc of Mass Spectrometry Exchange Formats is long, but it bends toward HDF5. Mass spectrometry reviews, 36(5), pp.668-673.

16. Tully, B., 2020. Toffee–a highly efficient, lossless file format for DIA-MS. Scientific reports, 10(1), pp.1-13.

17. Bhamber, R.S., Jankevics, A., Deutsch, E.W., Jones, A.R. and Dowsey, A.W., 2020. mzMLb: a future-proof raw mass spectrometry data format based on standards-compliant mzML and optimized for speed and storage requirements. Journal of proteome research, 20(1), pp.172-183.

18. Wilhelm, M., Kirchner, M., Steen, J.A. and Steen, H., 2012. mz5: space-and time-efficient storage of mass spectrometry data sets. Molecular & Cellular Proteomics, 11(1), pp.O111-011379.

19. Kandasamy K., Keerthikumar S., Goel R., Mathivanan S., Patankar N., Shafreen B. Human proteinpedia: a unified discovery resource for proteomics research. Nucleic Acids Res. 2009;37:D773–D781.

20. Vizcaino J.A., Cote R.G., Csordas A., Dianes J.A., Fabregat A., Foster J.M. The PRoteomics IDEntifications (PRIDE) database and associated tools: status in 2013. Nucleic Acids Res. 2013;41:D1063–D1069.

21. Perez-Riverol Y, Csordas A, Bai J, Bernal-Llinares M, Hewapathirana S, Kundu DJ, Inuganti A, Griss J, Mayer G, Eisenacher M, Pérez E, Uszkoreit J, Pfeuffer J, Sachsenberg T, Yilmaz S, Tiwary S, Cox J, Audain E, Walzer M, Jarnuczak AF, Ternent T, Brazma A, Vizcaíno JA. The PRIDE database and related tools and resources in 2019: improving support for quantification data.. Nucleic Acids Res. 2019 Jan 8;47(D1):D442-D450. doi: 10.1093/nar/gky1106. PubMed ID:30395289.

22. Perez-Riverol Y, Xu QW, Wang R, Uszkoreit J, Griss J, Sanchez A, Reisinger F, Csordas A, Ternent T, del Toro N, Dianes JA, Eisenacher M, Hermjakob H, Vizcaíno JA. PRIDE Inspector Toolsuite: moving towards a universal visualization tool for

proteomics data standard formats and quality assessment of ProteomeXchange datasets.. Mol Cell Proteomics 2016 Jan; 15(1):305-17. PubMed ID: 26545397.

23. Samaras, P., Schmidt, T., Frejno, M., Gessulat, S., Reinecke, M., Jarzab, A., Zecha, J., Mergner, J., Giansanti, P., Ehrlich, H.C. and Aiche, S., 2020. ProteomicsDB: a multi-omics and multi-organism resource for life science research. Nucleic acids research, 48(D1), pp.D1153-D1163.

24. Choi, M., Carver, J., Chiva, C., Tzouros, M., Huang, T., Tsai, T.H., Pullman, B., Bernhardt, O.M., Hüttenhain, R., Teo, G.C. and Perez-Riverol, Y., 2020. MassIVE. quant: a community resource of quantitative mass spectrometry–based proteomics datasets. Nature methods, 17(10), pp.981-984.

25. Deutsch E.W. The PeptideAtlas project. Methods Mol Biol. 2010;604:285–296.

26. Kusebauch, U., Deutsch, E.W., Campbell, D.S., Sun, Z., Farrah, T. and Moritz, R.L., 2014. Using PeptideAtlas, SRMAtlas, and PASSEL: comprehensive resources for discovery and targeted proteomics. Current protocols in bioinformatics, 46(1), pp.13-25.

# Overview of Proteomics Data Analysis

**Sudhir Srivastava, Sneha Murmu, Dwijesh Chandra Mishra,**

**U. B. Angadi and K. K. Chaturvedi**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

## Introduction

Proteins are important large biomolecules or macromolecules performing a wide variety of functions. The word "proteome" is defined as the entire set of proteins translated and/ or modified within a living organism. The word "proteome" was coined by Marc Wilkins in 1994 in a symposium on "2D Electrophoresis: from protein maps to genomes" held in Siena in Italy while he was a Ph.D. student at Macquarie University. An organism's genome is more or less constant whereas proteome is not constant. Proteomes differs from cell to cell and from time to time. That's why proteomics is more complicated when compared to genomics.

Proteomics more generally refers to large-scale liquid chromatography (LC) coupled with mass spectrometry (MS) [LC-MS] based discovery studies designed to address both quantitative and qualitative aspects of the proteome research (Figure 1).

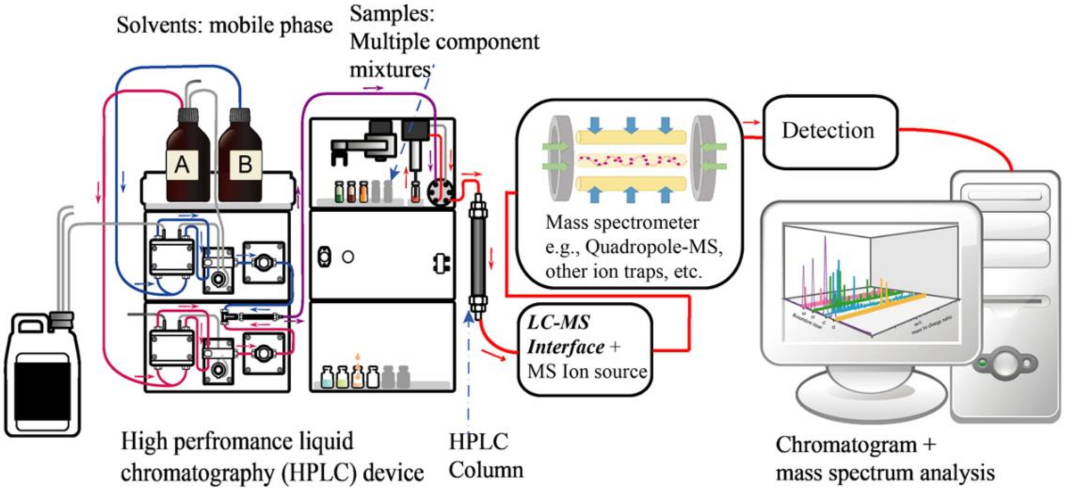


Figure 1. Liquid chromatography coupled with mass spectrometry [LC-MS]

**Source:** https://upload.wikimedia.org/wikipedia/en/f/f9/Liquid_chromatography_tandem_Mass_spectrometry_diagram.png

Now proteomics has emerged as a powerful tool across various fields such as biomedicine mainly applied to diseases, agriculture, and animal sciences. It is important for studying different aspects of plant functions such as identification of candidate proteins involved in the defensive response of plants to biotic and abiotic stresses, effect of global climate changes on crop production, etc. In animal sciences, proteomics studies play important role in studying physiology, immunology, reproduction and lactational biology. The practical application of proteomics includes expression proteomics, structural proteomics, biomarker discovery, interaction proteomics, protein networks, etc.

### Basics Steps of Proteomics Data Analysis

The proteomic abundance (expression) data are usually generated using high throughput technologies usually involving MS. LC-MS is used in proteomics as a method for identification and quantification of peptides and proteins in complex mixtures. There are two basic proteomics approaches, namely bottom-up and top-down. The most common proteomics approach is the bottom-up in which proteins in a sample are enzymatically digested into peptides and subjected to chromatographic separation, ionization and mass analysis. Conversely, top-down proteomics addresses the study of intact proteins and consequently is most often used to address purified or partially purified proteins. There are various steps involved in quantitative proteomics data analysis, viz., peptide and protein identification, protein abundance quantification, data cleaning, data normalization, handling of missing values by using imputation techniques, data visualization and interpretation, statistical analysis of proteomics data, etc.

### Peptide and protein identification

There are two major approaches for determining the sequence of peptides.

(i) Searching against fragmentation spectra databases

(ii) de novo peptide sequencing

Some of the software/ tools for peptide and protein identification are listed below:

| **Category** | **Name** | **Description** |
|---|---|---|
| Searching against fragmentation spectra databases | Andromeda (part of Mascot) | A peptide search engine based on probabilistic scoring |
| | Mascot | Probability-based database searching algorithm |

| | | |
|---|---|---|
| | SEQUEST | Identifies collections of tandem mass spectra to peptide sequences that have been generated from protein sequence databases |
| | X!Tandem/X!!Tandem | Searches tandem mass spectra with peptide sequences in database |
| de novo peptide sequencing | PEAKS | Performs de novo sequencing for each peptide, confidence scores on individual amino acid assignments with manually assisted mode and automated de novo sequencing on an entire LC run processed data |
| | SHERENGA | Performs de novo peptide sequencing via tandem mass spectrometry |
| | PECAN | Library free peptide detection for data-independent acquisition of tandem mass spectrometry data |

### Quantification of feature abundance

The quantification of features (peptides or proteins) may be either label-free or labelled (metabolic, enzymatic, or chemical) to detect differences in feature abundances among different conditions. In label-free quantification, MS ion intensity (peak area) and spectral counting of features are the major approaches. In this article, we have considered MS ion intensity data obtained from label-free bottom-up proteomics experiments.

Software/Tools for label-based quantitative proteomics:

- MaxQuant
- Proteome Discoverer (Thermo Scientific)
- XPRESS

Software/Tools for label-free quantitative proteomics:

- MaxLFQ - Label free quantification module available in MaxQuant
- emPAI - Exponentially modified protein abundance index
- Mascot Distiller (Matrix Science)

### Problem of missing values and heterogeneity in proteomics data

Various approaches exist for proteomics data analysis in which the first step is to summarize the intensities of all features using a quantitative summary followed by logarithmic transformation to approximate it to normal distribution. In spite of availability of various tools/methods, there are various challenges in analyzing proteomics data such as missing value (MV) and data heterogeneity. There are various drawbacks of the methods which can be studied by examining the statistical properties of these methods.

The variations in the biological data or technical approaches to data collection lead to heterogeneity for the samples under study. The data set usually consists of biological replicates only or both biological and technical replicates. Biological variability arises from genetic and environmental factors and it is intrinsic to all organisms. The technical approaches include sample collection and storage, sample preparation, extraction, LC separation and MS detection.

The data set is called balanced when it contains an equal number of subjects/ samples in each group, and the features have no missing observations. However, this is not always the condition. Sometimes the data can be unbalanced having unequal number of subjects, or missing observations, or both. MVs in proteomics data can occur due to biological and/or technical issues. These are of three types of MVs: (i) missing completely at random (MCAR) in which MVs are independent of both unobserved and observed data; (ii) missing at random (MAR) if conditional on the observed data, the MVs are independent of the missing measurements; and (iii) missing not at random (MNAR) when data is neither MCAR nor MAR. The data with missing observations can be analyzed either by excluding the features having missing observations, by using statistical methods that can handle unbalanced data, or by using imputation methods. If the features having missing observations are excluded, then there is loss of information from the experiment. Therefore, the use of methods that can handle MVs, such as imputation methods, are generally preferred. However, the use of imputation methods may lead to wrong interpretation and these methods are questionable in statistical terms.

### Statistical analysis of proteomics abundance data

Differential abundance analysis is carried out to detect significant features in two or more conditions such as normal versus different disease conditions. However, data normalization is necessary before performing further analysis. There are various transformation and/ or normalization methods such as logarithmic transformation, quantile normalization, variance stabilizing normalization, median scaling normalization, etc. In case of missing values, the user has to impute the data using imputation techniques such as singular value decomposition, *k*-nearest neighbor, maximum likelihood estimation, etc. The statistical approaches/ tests such as t-test, moderated t-test, ANOVA, linear mixed model, etc. can be used for detecting significant features. A general workflow of label-free quantitative proteomics data is given below:

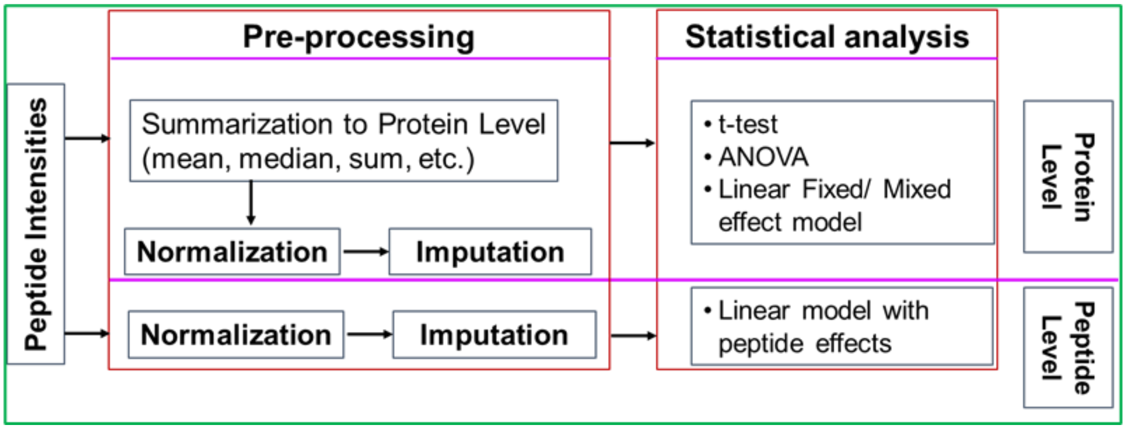


Figure 2. A general workflow of label-free quantitative proteomics data

Various methods of normalizing proteomics expression data are given below:

- Variance stabilizing normalization (VSN)
- Quantile normalization (quantile)
- Median normalization (median)
- EigenMS normalization (EigenMS)
- Local regression normalization (LoessF, LoessCyc)

Various imputation methods can be categorized into the following:

(i) Imputation by a single value:

- Half of global minimum intensity among peptides - the minimal observed intensity value among all peptides
- Half of minimal intensity of individual peptide
- Random tail imputation

(ii) Local-similarity-based imputation methods:

- *K*-nearest neighbors (KNN)

- Local least-squares (LLS) imputation
- Regularized expectation maximization (REM) algorithm

(iii) Global-structure-based imputation methods

- Probabilistic principal component analysis (PPCA)
- Bayesian principal component analysis (BPCA) algorithm

There are various tools and packages available for proteomics abundance data analysis such as DanteR, MSstats, RepExplore, PANDA-view, MSqRob, PANDA, DAPAR, ProStaR etc. Some of the important tools are discussed below:

(i) DanteR: Taverner *et al.* (2012) developed DanteR, a graphical R package that features extensive statistical and diagnostic functions for quantitative proteomics data analysis, including normalization, imputation, hypothesis testing, interactive visualization and peptide-to-protein rollup.

(ii) MSstats: Choi *et al.* (2014) developed an R package "MSstats" for statistical relative quantification of proteins and peptides in MS based proteomics. It (version 2.0) supports label-free and label-based experimental workflows and data-dependent, targeted and data-independent spectral acquisition. It performs differentially abundance/ expression analysis of features (peptides or proteins) based on linear mixed models.

(iii) RepExplore: Glaab and Schneider (2015) developed a web server "RepExplore" to analyse the proteomics and metabolomics data with technical and biological replicates. The analysis is based on previously published statistical methods, which have been applied successfully to biomedical omics.

(iv) PANDA-view: Chang *et al.* (2018) developed an easy-to-use tool "PANDA-view" for both statistical analysis and visualization of quantitative proteomics data and other -omics data. There are various kinds of analysis methods such as normalization, MV imputation, statistical tests, clustering and principal component analysis, an interactive volcano plot.

(v) MSqRob: Goeminne *et al.* (2018) provided a tutorial on analysis of quantitative proteomics data. The tutorial discussed the key statistical concepts to design proteomics experiments and analyse label-free MS based quantitative proteomics data using their free and open-source R package MSqRob.

(vi) PANDA: Chang *et al.* (2019) developed a comprehensive and flexible tool named PANDA for proteomics data quantification. The tool supports both label-free and labeled quantifications and it is compatible with existing peptide identification tools and pipelines with considerable flexibility.

(vii) DAPAR & ProStaR: Wieczorek et al. (2017) developed software tools, DAPAR and ProStaR that can perform the statistical analysis of label-free XIC-based quantitative discovery proteomics experiments. DAPAR is an R package that contains various functions such as filtering, normalization, imputation of missing values, aggregation of peptide intensities, differential abundance analysis of proteins, etc. ProStaR is a user-friendly graphical interface that allows access to the DAPAR functionalities through a web browser.

## Conclusion

In this article, we have given the basic introduction of proteomics, various steps of proteomics data analysis, problem of MVs and heterogeneity in proteomics data and different methods for analysis of proteomics data. This article will be useful for the researchers working in the field of proteomics and bioinformatics. Furthermore, the methods for proteomics data analysis can further be used for analyzing the expression data obtained from similar experiments (e.g., microarray and metabolomics data).

## References

Anderson NL, Anderson NG (1998). Proteome and proteomics: new technologies, new concepts, and new words. *Electrophoresis,* **19(11)**, 1853-61.

Ceciliani F, Eckersall D, Burchmore R, Lecchi C. (2014). Proteomics in veterinary medicine: applications and trends in disease pathogenesis and diagnostics. *Vet Pathol*., **51(2)**:351-62. doi: 10.1177/0300985813502819.

Chang C, et al. (2018). PANDA-view: An easy-to-use tool for statistical analysis and visualization of quantitative proteomics data. *Bioinformatics*.

Choi M, et al. (2014). MSstats: an R package for statistical analysis of quantitative mass spectrometry-based proteomic experiments. *Bioinformatics*, **30(17)**. 2524-6.

Glaab E, Schneider R (2015). RepExplore: addressing technical replicate variance in proteomics and metabolomics data analysis. *Bioinformatics*, **31(13)**, 2235-7.

Goeminne LJE, Gevaert K, and Clement L (2018). Experimental design and data-analysis in label-free quantitative LC/MS proteomics: A tutorial with MSqRob. *J Proteomics*, **171**, 23-36.

Karpievitch YV, Dabney AR, and Smith RD (2012). Normalization and missing value imputation for label-free LC-MS analysis. *BMC Bioinformatics*, **13 Suppl 16**, S5.

Rubin DB (1976). Inference and missing data. *Biometrika*, **63(3)**, 581–92.

Taverner T., et al. (2012). DanteR: an extensible R-based tool for quantitative analysis of -omics data. *Bioinformatics*, **28(18)**, 2404–2406. doi:10.1093/bioinformatics/bts449.

Wasinger, VC, Cordwell, SJ, Cerpa-Poljak, A, Yan, JX, Gooley, AA, Wilkins, MR, Duncan, MW, Harris, R, Williams, KL, Humphery-Smith, I (1995). Progress with gene-product mapping of the Mollicutes: *Mycoplasma genitalium*. *Electrophoresis*, **16 (1)**, 1090-1094. doi:10.1002/elps.11501601185

Wieczorek, S., Combes, F., Lazar, C., Giai Gianetto, Q., Gatto, L., Dorffer, A., Hesse, A.-M., Couté, Y., Ferro, M., Bruley, C., & Burger, T. (2017). DAPAR & ProStaR: software to perform statistical analyses in quantitative discovery proteomics. *Bioinformatics* (Oxford, England), **33(1)**, 135-136. https://doi.org/10.1093/bioinformatics/btw580

https://en.wikipedia.org/wiki/Proteomics

https://en.wikipedia.org/wiki/List_of_mass_spectrometry_software

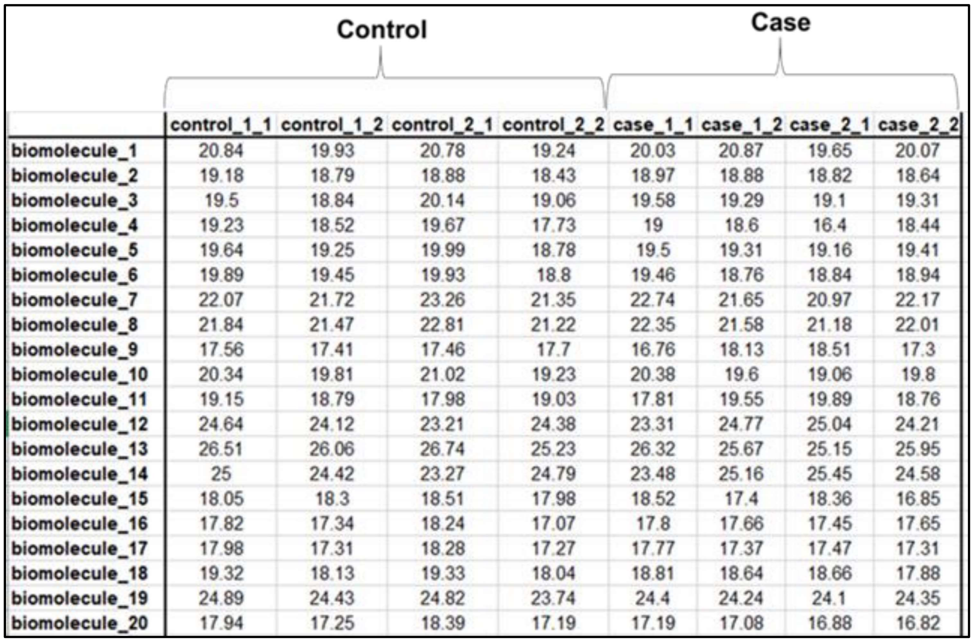
# Working with Proteomics Data Analysis

**Sudhir Srivastava, Sneha Murmu and K. K. Chaturvedi**
**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

In this article, proteomics abundance data analysis has been demonstrated by using an online web tool "RepExplore" and a shiny app "ProStaR".

**Differential abundance analysis using RepExplore**

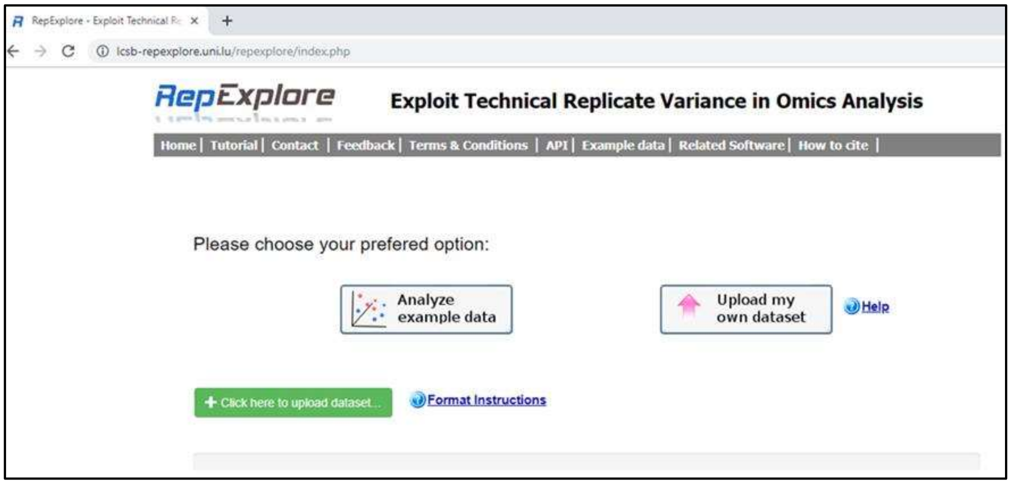
In this article, we are dealing with the bottom-up approach in which peak area values have been used in label-free quantification of proteins. An example of proteomics abundance data analysis using "RepExplore" has been illustrated below. A portion of an example dataset for a case-control study is shown in Figure 1. The dataset has two biological replicates each having two technical replicates in each group (case and control).



| | Control | | | | Case | | | |
|---|---|---|---|---|---|---|---|---|
| | control_1_1 | control_1_2 | control_2_1 | control_2_2 | case_1_1 | case_1_2 | case_2_1 | case_2_2 |
| biomolecule_1 | 20.84 | 19.93 | 20.78 | 19.24 | 20.03 | 20.87 | 19.65 | 20.07 |
| biomolecule_2 | 19.18 | 18.79 | 18.88 | 18.43 | 18.97 | 18.88 | 18.82 | 18.64 |
| biomolecule_3 | 19.5 | 18.84 | 20.14 | 19.06 | 19.58 | 19.29 | 19.1 | 19.31 |
| biomolecule_4 | 19.23 | 18.52 | 19.67 | 17.73 | 19 | 18.6 | 16.4 | 18.44 |
| biomolecule_5 | 19.64 | 19.25 | 19.99 | 18.78 | 19.5 | 19.31 | 19.16 | 19.41 |
| biomolecule_6 | 19.89 | 19.45 | 19.93 | 18.8 | 19.46 | 18.76 | 18.84 | 18.94 |
| biomolecule_7 | 22.07 | 21.72 | 23.26 | 21.35 | 22.74 | 21.65 | 20.97 | 22.17 |
| biomolecule_8 | 21.84 | 21.47 | 22.81 | 21.22 | 22.35 | 21.58 | 21.18 | 22.01 |
| biomolecule_9 | 17.56 | 17.41 | 17.46 | 17.7 | 16.76 | 18.13 | 18.51 | 17.3 |
| biomolecule_10 | 20.34 | 19.81 | 21.02 | 19.23 | 20.38 | 19.6 | 19.06 | 19.8 |
| biomolecule_11 | 19.15 | 18.79 | 17.98 | 19.03 | 17.81 | 19.55 | 19.89 | 18.76 |
| biomolecule_12 | 24.64 | 24.12 | 23.21 | 24.38 | 23.31 | 24.77 | 25.04 | 24.21 |
| biomolecule_13 | 26.51 | 26.06 | 26.74 | 25.23 | 26.32 | 25.67 | 25.15 | 25.95 |
| biomolecule_14 | 25 | 24.42 | 23.27 | 24.79 | 23.48 | 25.16 | 25.45 | 24.58 |
| biomolecule_15 | 18.05 | 18.3 | 18.51 | 17.98 | 18.52 | 17.4 | 18.36 | 16.85 |
| biomolecule_16 | 17.82 | 17.34 | 18.24 | 17.07 | 17.8 | 17.66 | 17.45 | 17.65 |
| biomolecule_17 | 17.98 | 17.31 | 18.28 | 17.27 | 17.77 | 17.37 | 17.47 | 17.31 |
| biomolecule_18 | 19.32 | 18.13 | 19.33 | 18.04 | 18.81 | 18.64 | 18.66 | 17.88 |
| biomolecule_19 | 24.89 | 24.43 | 24.82 | 23.74 | 24.4 | 24.24 | 24.1 | 24.35 |
| biomolecule_20 | 17.94 | 17.25 | 18.39 | 17.19 | 17.19 | 17.08 | 16.88 | 16.82 |

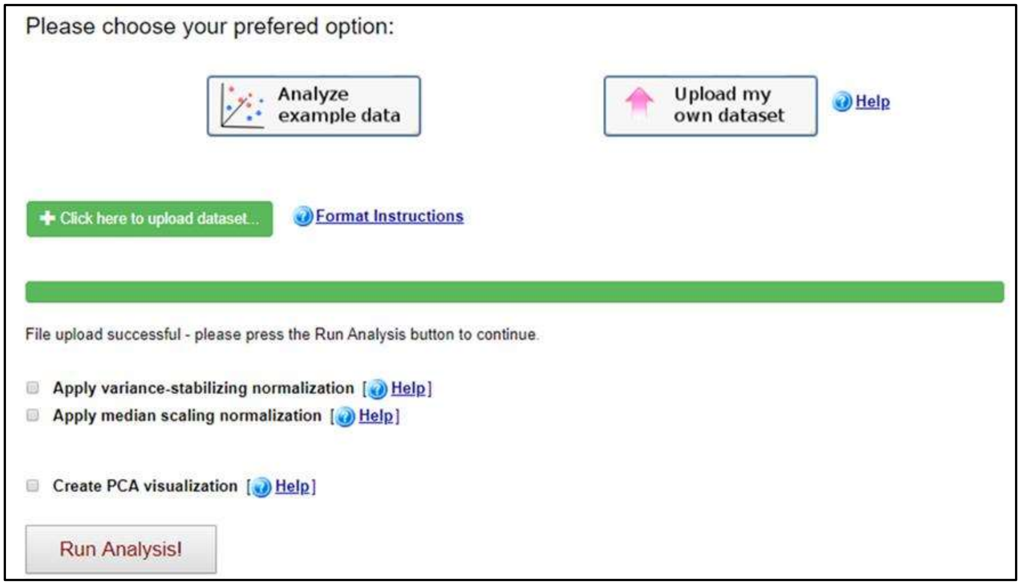
Figure 1. A portion of test dataset for a case-control study

The user has to upload the abundance data as given in Figure 2. The user has to choose various options after uploading the data (Figure 3). Then, the user has to click on "Run Analysis!" button.


RepExplore - Exploit Technical R
lcsb-repexplore.uni.lu/repexplore/index.php
RepExplore
Exploit Technical Replicate Variance in Omics Analysis
Home | Tutorial | Contact | Feedback | Terms & Conditions | API | Example data | Related Software | How to cite |
Analyze example data
Upload my own dataset
Help
Click here to upload dataset...
Format Instructions

Please choose your prefered option:

Figure 2. Upload the data



Please choose your prefered option:

File upload successful - please press the Run Analysis button to continue.

□ Apply variance-stabilizing normalization [Help]

□ Apply median scaling normalization [Help]

□ Create PCA visualization [Help]

Figure 3. Selecting the options

Then, the user will get menus of various results as shown below in Figure 4.

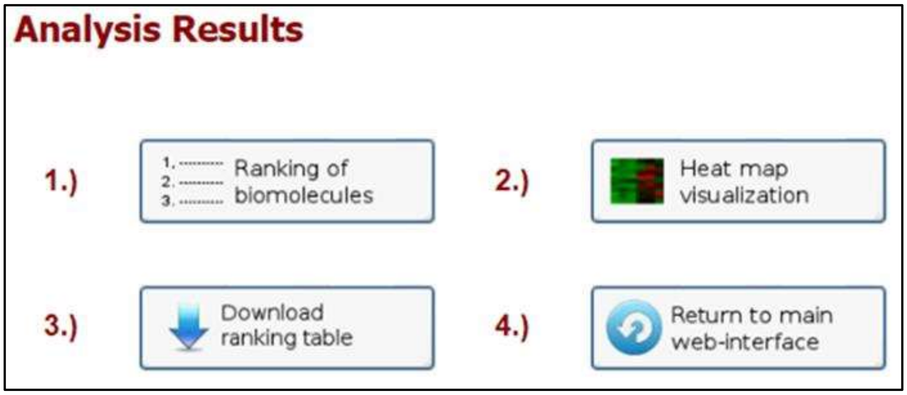
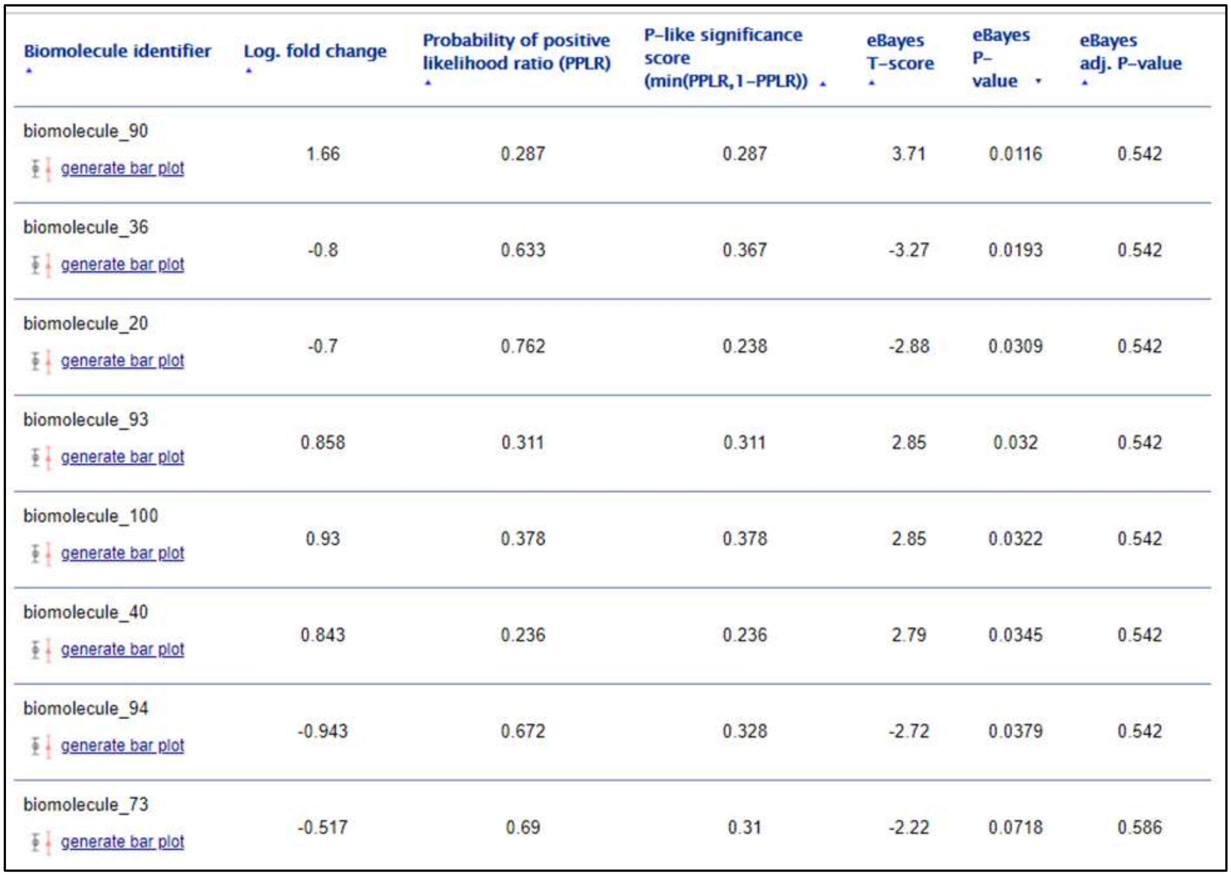


Figure 4. The menus of various results obtained

The ranking table of differential abundant/ expressed features is given in Figure 5.



| Biomolecule identifier | Log. fold change | Probability of positive likelihood ratio (PPLR) | P-like significance score (min(PPLR,1-PPLR)) | eBayes T-score | eBayes P-value | eBayes adj. P-value |
|---|---|---|---|---|---|---|
| biomolecule_90<br>generate bar plot | 1.66 | 0.287 | 0.287 | 3.71 | 0.0116 | 0.542 |
| biomolecule_36<br>generate bar plot | -0.8 | 0.633 | 0.367 | -3.27 | 0.0193 | 0.542 |
| biomolecule_20<br>generate bar plot | -0.7 | 0.762 | 0.238 | -2.88 | 0.0309 | 0.542 |
| biomolecule_93<br>generate bar plot | 0.858 | 0.311 | 0.311 | 2.85 | 0.032 | 0.542 |
| biomolecule_100<br>generate bar plot | 0.93 | 0.378 | 0.378 | 2.85 | 0.0322 | 0.542 |
| biomolecule_40<br>generate bar plot | 0.843 | 0.236 | 0.236 | 2.79 | 0.0345 | 0.542 |
| biomolecule_94<br>generate bar plot | -0.943 | 0.672 | 0.328 | -2.72 | 0.0379 | 0.542 |
| biomolecule_73<br>generate bar plot | -0.517 | 0.69 | 0.31 | -2.22 | 0.0718 | 0.586 |

Figure 5. The ranking table of differential abundant/expressed features

The user can generate the bar plot of any feature by clicking “generate_bar_plot” button for which an example is shown below.

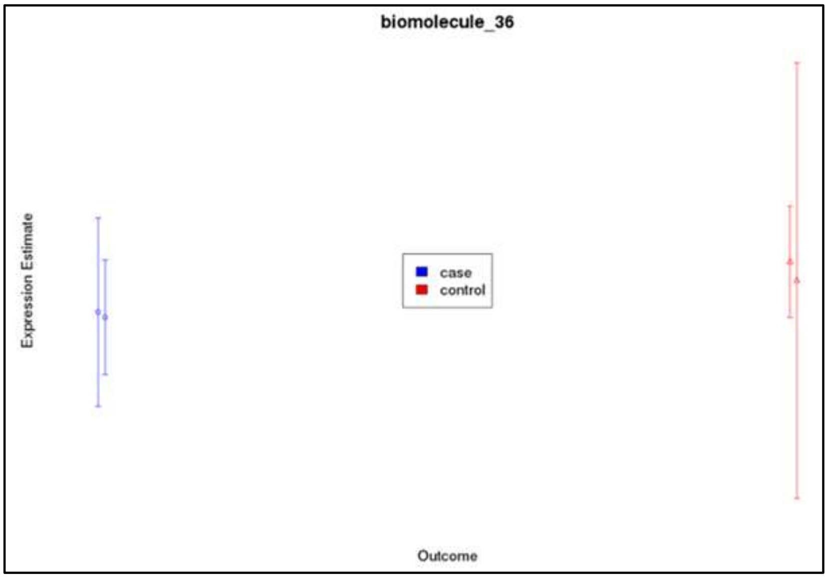


Figure 6. An example of bar plot of a feature

### Differential Abundance Analysis using ProStaR

Prostar (Proteomics statistical analysis with R) has been used to demonstrate the analysis of label-free quantitative proteomics data. Various steps involved are given below:

### Software and package installation

Download and install the latest version of R.

After installation open R console.

To install Bioconductor package manager run the following commands:

```
if  (!requireNamespace(“BiocManager”, quietly = TRUE))
     install.package(“BiocManager”)
BiocManager::install(versiob=’3.14’)
```

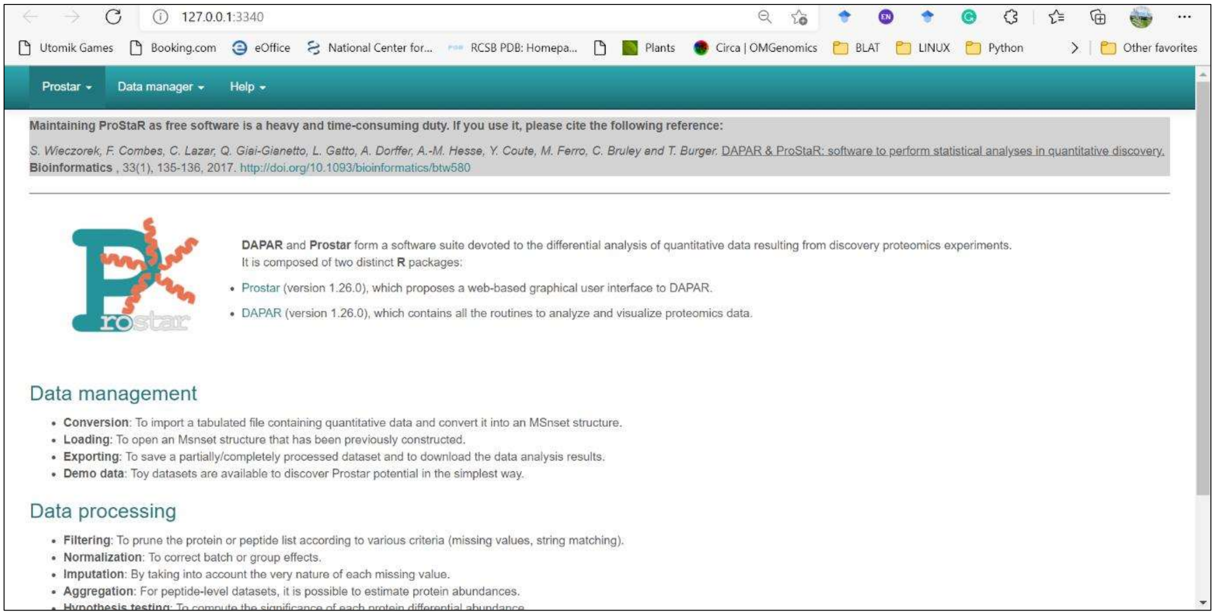
To install Prostar run the following command:

```
BiocManager::install(“Prostar”)
```

To launch Prostar run the following command:

```
library(Prostar)
Prostar()
```

The homepage of the web application in the browser as shown in Figure 7 will be opened after executing the above R codes.



Maintaining ProStaR as free software is a heavy and time-consuming duty. If you use it, please cite the following reference:

*S. Wieczorek, F. Combes, C. Lazar, Q. Giai-Gianetto, L. Gatto, A. Dorffer, A.-M. Hesse, Y. Coute, M. Ferro, C. Bruley and T. Burger.* DAPAR & ProStaR: software to perform statistical analyses in quantitative discovery. **Bioinformatics** , 33(1), 135-136, 2017. http://doi.org/10.1093/bioinformatics/btw580

**DAPAR** and **Prostar** form a software suite devoted to the differential analysis of quantitative data resulting from discovery proteomics experiments.
It is composed of two distinct **R** packages:

- Prostar (version 1.26.0), which proposes a web-based graphical user interface to DAPAR.
- DAPAR (version 1.26.0), which contains all the routines to analyze and visualize proteomics data.

Data management

- **Conversion**: To import a tabulated file containing quantitative data and convert it into an MSnset structure.
- **Loading**: To open an Msnset structure that has been previously constructed.
- **Exporting**: To save a partially/completely processed dataset and to download the data analysis results.
- **Demo data**: Toy datasets are available to discover Prostar potential in the simplest way.

Data processing

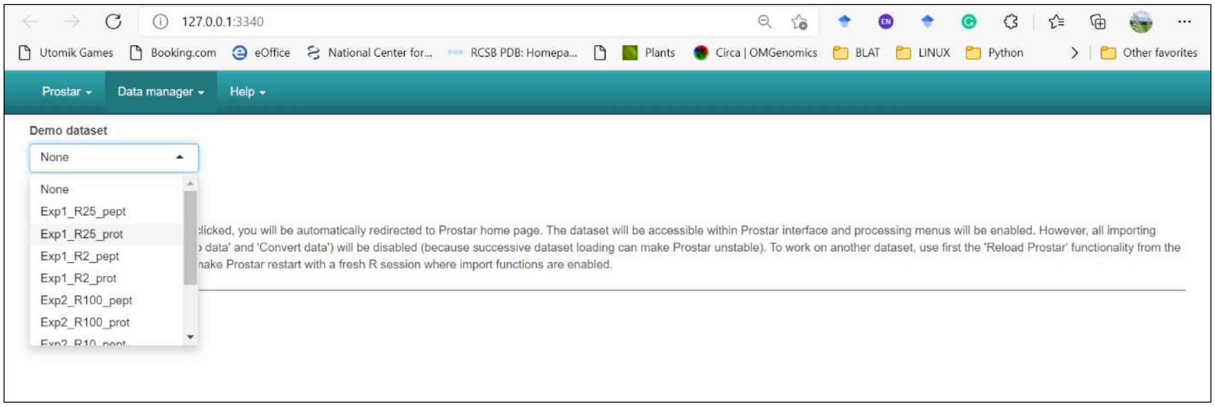
- **Filtering**: To prune the protein or peptide list according to various criteria (missing values, string matching).
- **Normalization**: To correct batch or group effects.
- **Imputation**: By taking into account the very nature of each missing value.
- **Aggregation**: For peptide-level datasets, it is possible to estimate protein abundances.
- **Hypothesis testing**: To compute the significance of each protein differential abundance

Figure 7. Homepage of Prostar web application.

### Data loading

- Click on “Demo data” in the dropdown menu of “Data manager”.
- Select the data named Exp1_R25_prot provided in the package (shown in Figure 8).

- Click on “Load demo dataset”.



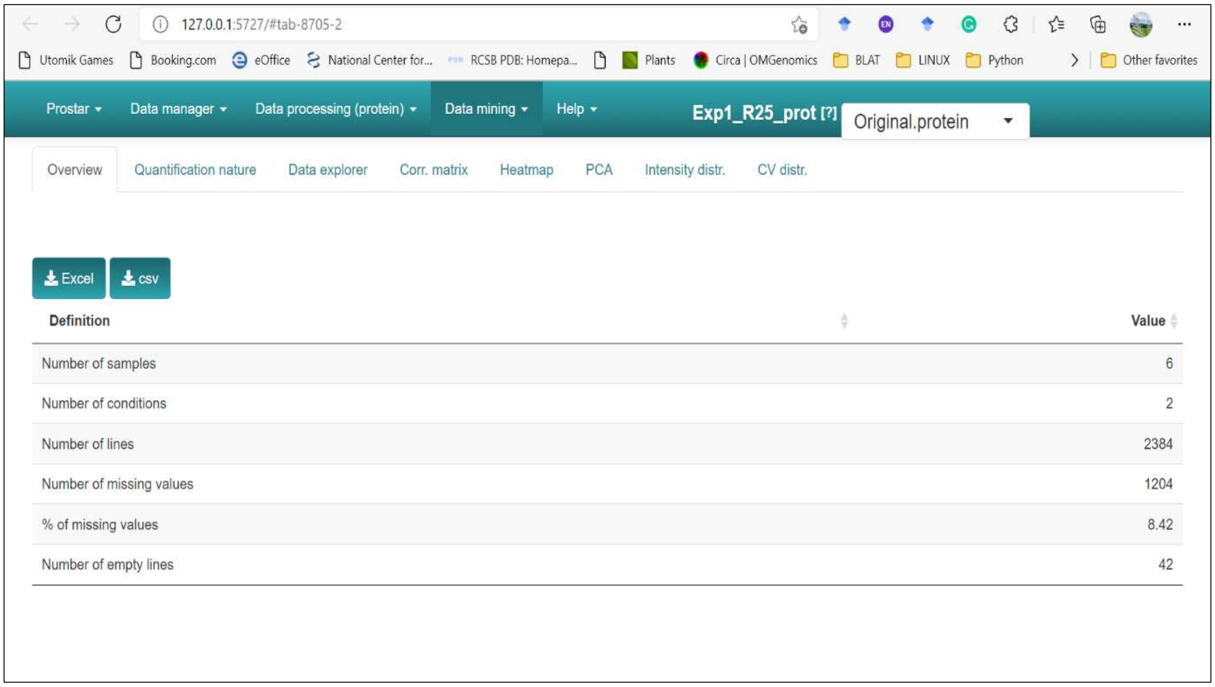
licked, you will be automatically redirected to Prostar home page. The dataset will be accessible within Prostar interface and processing menus will be enabled. However, all importing
data' and 'Convert data') will be disabled (because successive dataset loading can make Prostar unstable). To work on another dataset, use first the 'Reload Prostar' functionality from the
nake Prostar restart with a fresh R session where import functions are enabled.

Figure 8. Select the dataset.

### Descriptive statistics

1. Click on “Descriptive statistics” in the “Data mining” menu to access several tabs generating various plots.

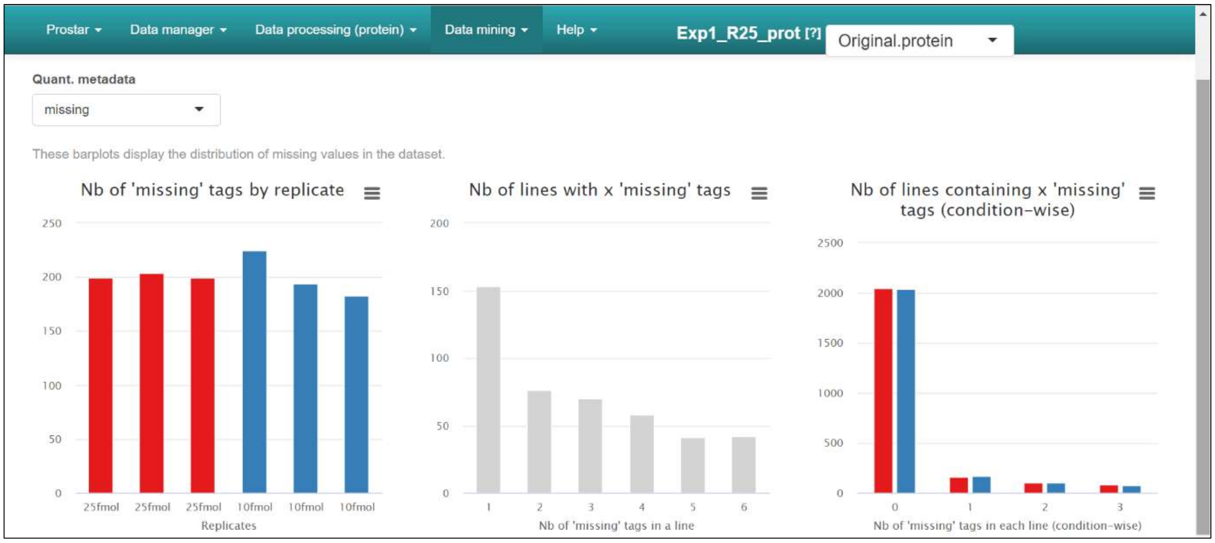
2. The “Overview” tab provides the quantitative data summary as shown in Figure 9.



| Definition | Value |
|---|---|
| Number of samples | 6 |
| Number of conditions | 2 |
| Number of lines | 2384 |
| Number of missing values | 1204 |
| % of missing values | 8.42 |
| Number of empty lines | 42 |

Figure 9. Brief summary of the quantitative data size.

3. “Missing value” tab depicts the distribution of missing values (MVs) condition and sample-wise as shown in Figure 10.



Prostar
Data manager
Data processing (protein)
Data mining
Help
Exp1_R25_prot [?]
Original.protein

Quant. metadata

missing

These barplots display the distribution of missing values in the dataset.

Nb of 'missing' tags by replicate
250
200
150
100
50
0
25fmol
25fmol
25fmol
10fmol
10fmol
10fmol
Replicates
Nb of lines with x 'missing' tags
200
150
100
50
0
1
2
3
4
5
6
Nb of 'missing' tags in a line
Nb of lines containing x 'missing' tags (condition-wise)
2500
2000
1500
1000
500
0
0
1
2
3
Nb of 'missing' tags in each line (condition-wise)

Figure 10. Information on missing value. The left-hand side barplot represents the number of MVs in each sample. The second barplot (in the middle) displays the distribution of MVs. The last barplot represents the same information as the previous one condition-wise.

4. Click on “Data explorer” tab to view the content of the dataset. It is made of three tables.

- ✓ "Quantitative data" contains quantitative values (Figure 11). The missing values are represented by empty cells.
- ✓ "Protein metadata" contains all the column dataset that are not the quantitative data (Figure 12).
- ✓ "Experimental design", summarize the experimental design (Figure 13).

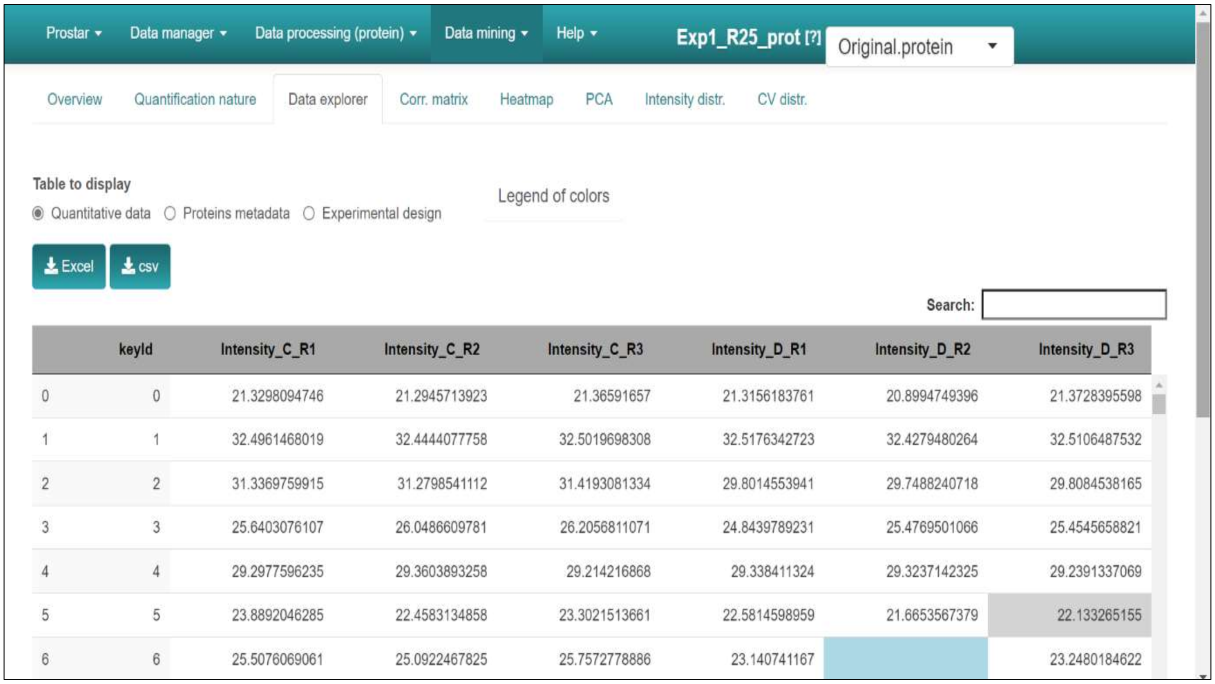


Table to display

● Quantitative data ○ Proteins metadata ○ Experimental design

Search:

| | keyId | Intensity_C_R1 | Intensity_C_R2 | Intensity_C_R3 | Intensity_D_R1 | Intensity_D_R2 | Intensity_D_R3 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 21.3298094746 | 21.2945713923 | 21.36591657 | 21.3156183761 | 20.8994749396 | 21.3728395598 |
| 1 | 1 | 32.4961468019 | 32.4444077758 | 32.5019698308 | 32.5176342723 | 32.4279480264 | 32.5106487532 |
| 2 | 2 | 31.3369759915 | 31.2798541112 | 31.4193081334 | 29.8014553941 | 29.7488240718 | 29.8084538165 |
| 3 | 3 | 25.6403076107 | 26.0486609781 | 26.2056811071 | 24.8439789231 | 25.4769501066 | 25.4545658821 |
| 4 | 4 | 29.2977596235 | 29.3603893258 | 29.214216868 | 29.338411324 | 29.3237142325 | 29.2391337069 |
| 5 | 5 | 23.8892046285 | 22.4583134858 | 23.3021513661 | 22.5814598959 | 21.6653567379 | 22.133265155 |
| 6 | 6 | 25.5076069061 | 25.0922467825 | 25.7572778886 | 23.140741167 | | 23.2480184622 |

Figure 11. Quantitative data

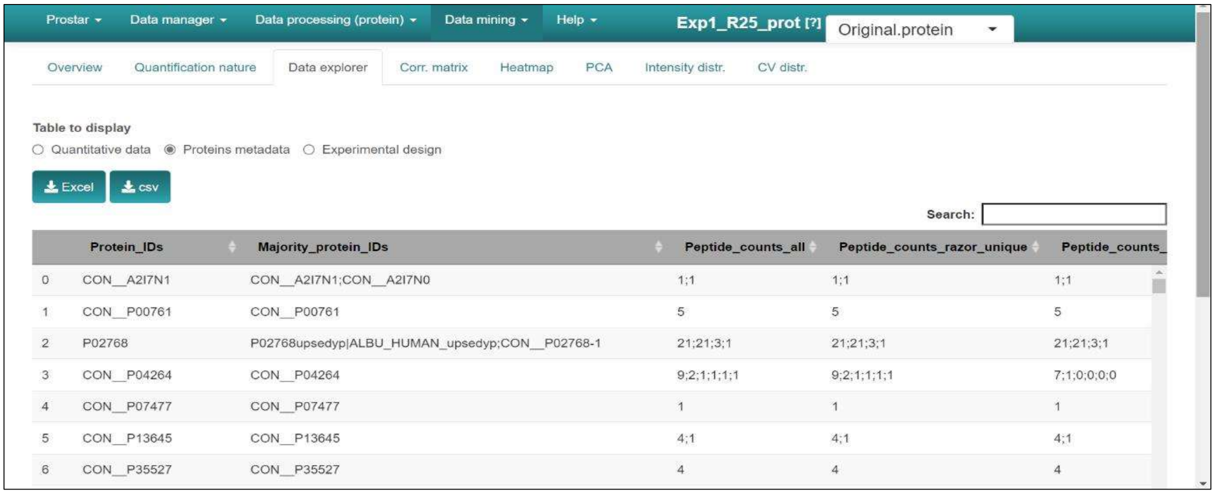


Table to display

○ Quantitative data ● Proteins metadata ○ Experimental design

|  | Protein_IDs | Majority_protein_IDs | Peptide_counts_all | Peptide_counts_razor_unique | Peptide_counts_ |
|---|---|---|---|---|---|
| 0 | CON__A2I7N1 | CON__A2I7N1;CON__A2I7N0 | 1;1 | 1;1 | 1;1 |
| 1 | CON__P00761 | CON__P00761 | 5 | 5 | 5 |
| 2 | P02768 | P02768upsedyp\|ALBU_HUMAN_upsedyp;CON__P02768-1 | 21;21;3;1 | 21;21;3;1 | 21;21;3;1 |
| 3 | CON__P04264 | CON__P04264 | 9;2;1;1;1;1 | 9;2;1;1;1;1 | 7;1;0;0;0;0 |
| 4 | CON__P07477 | CON__P07477 | 1 | 1 | 1 |
| 5 | CON__P13645 | CON__P13645 | 4;1 | 4;1 | 4;1 |
| 6 | CON__P35527 | CON__P35527 | 4 | 4 | 4 |

Figure 12. Proteins metadata

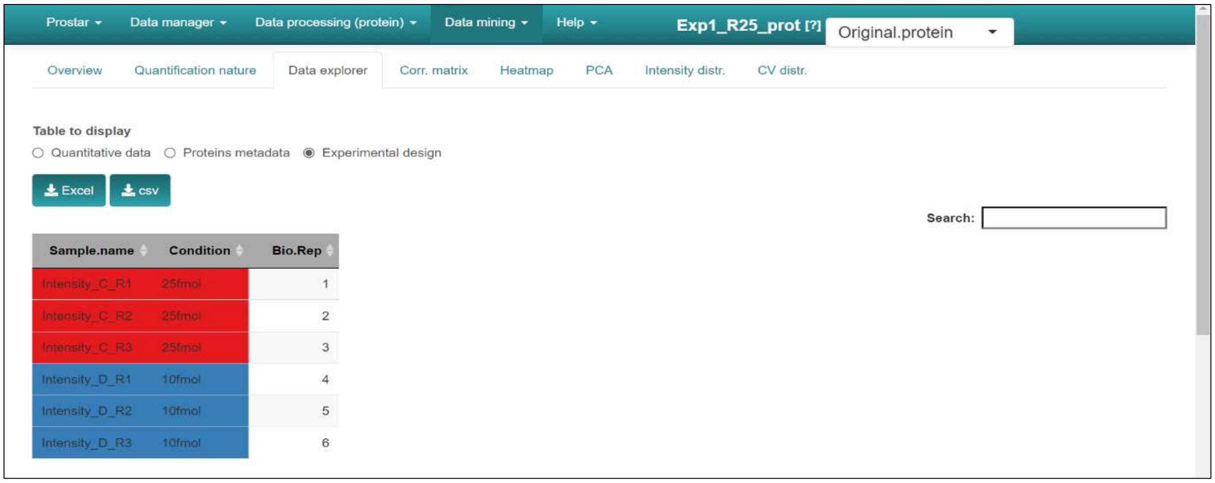


Table to display

○ Quantitative data ○ Proteins metadata ● Experimental design

| Sample.name | Condition | Bio.Rep |
|---|---|---|
| Intensity_C_R1 | 25fmol | 1 |
| Intensity_C_R2 | 25fmol | 2 |
| Intensity_C_R3 | 25fmol | 3 |
| Intensity_D_R1 | 10fmol | 4 |
| Intensity_D_R2 | 10fmol | 5 |
| Intensity_D_R3 | 10fmol | 6 |

Figure 13. Experimental design

5. Click on fourth tab “Correlation. matrix”, to visualize to what extent the replicate samples correlate or not as shown in Figure 14.

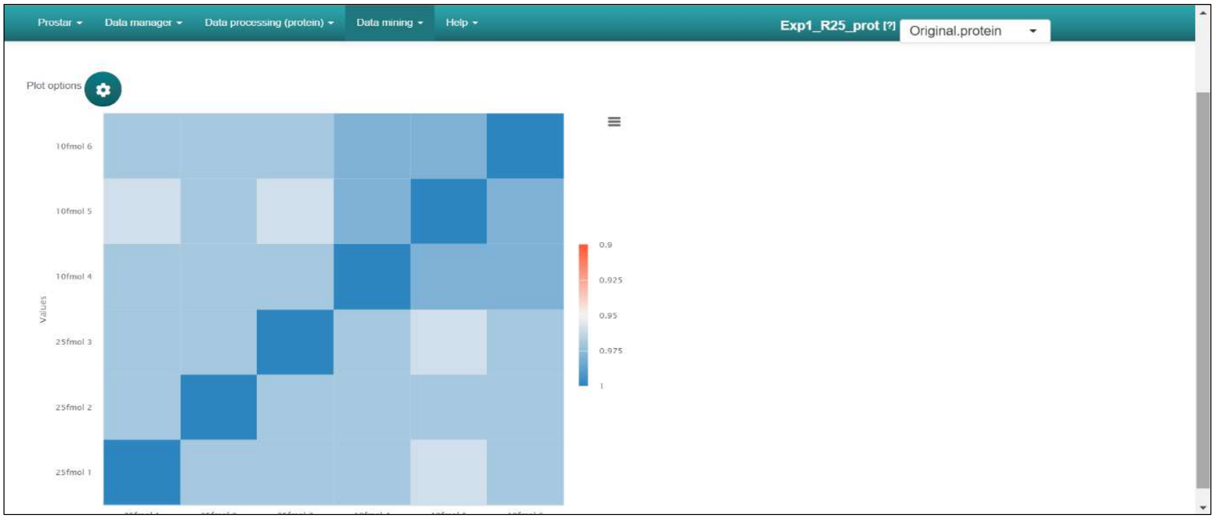


Figure 14. Correlation matrix

6. The fifth tab depicts the heatmap and associated dendrogram as shown in Figure 15. The dendrogram shows a hierarchical classification of the samples, so as to check that samples are related according to the experimental design.

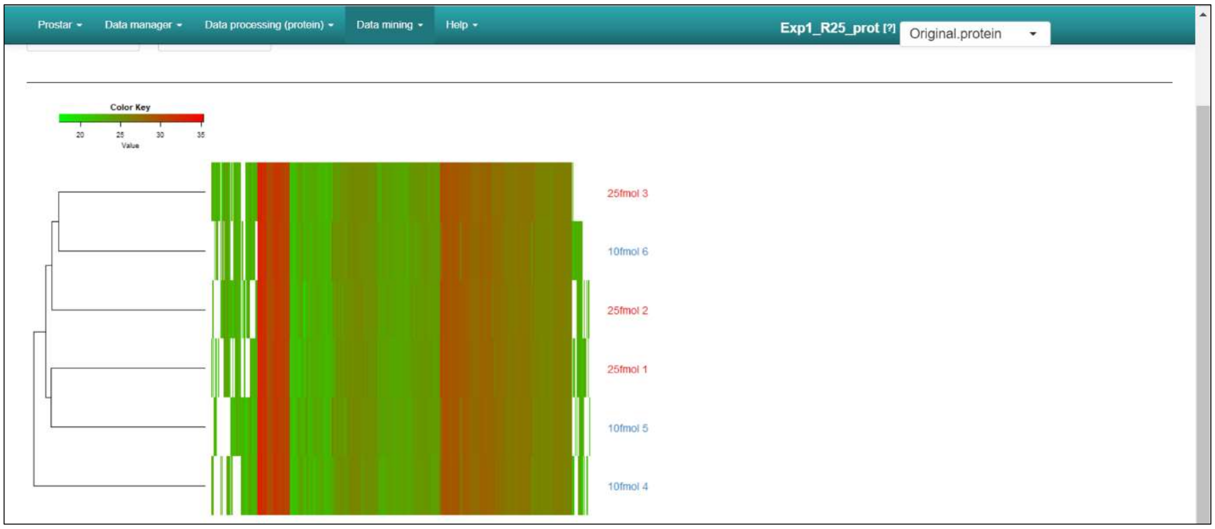


Figure 15. Heatmap: Red represents high intensities and green is for low intensities. White colour represents missing values.

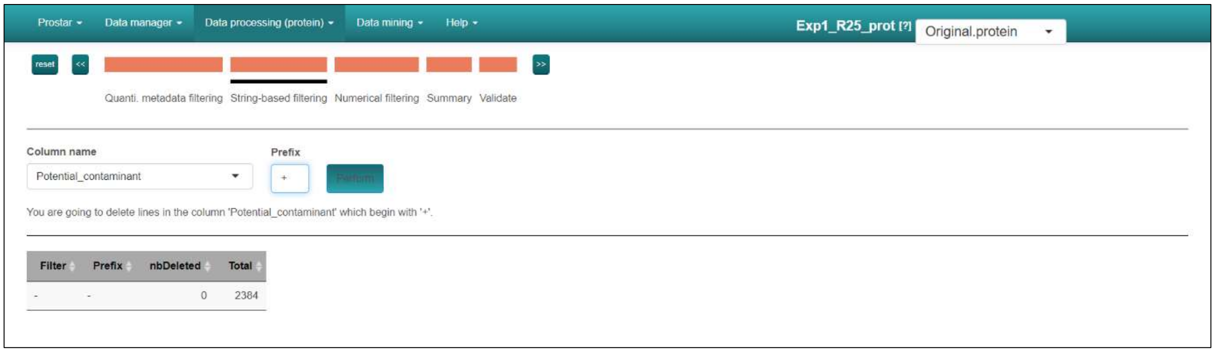
### Filtering

This aim is to filter out proteins according to their number of missing values, as well as according to some information stored in the protein metadata.

1. Click on “Filter data” in the “Data processing” menu.

2. Click on "Missing values” to select among the various options which proteins should be filtered out or not. In this case we do not filter out the missing values as later will be imputed later in the stage.

4. Click on "String based filtering", to filter out proteins according to information stored in the metadata. Among the columns constituting the protein metadata listed in the drop-down menu, select the one containing the information of interest (“Contaminant” and “Reverse”). Then, specify in each case the prefix chain of characters that identifies the proteins to filter. In this case it is plus sign (+) as shown in Figure 16.



Column name: Potential_contaminant

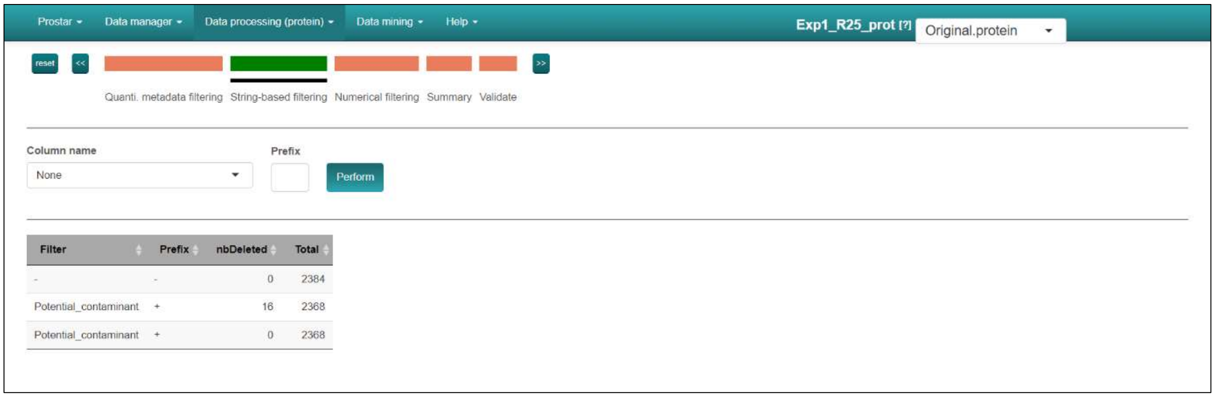
Prefix: +

You are going to delete lines in the column 'Potential_contaminant' which begin with '+'.

| Filter | Prefix | nbDeleted | Total |
|---|---|---|---|
| - | - | 0 | 2384 |

Figure 16. Remove the contaminants

6. Click on "Perform " to remove the corresponding proteins. A new line appears in the table listing all the filters that have been applied as shown in Figure 17.



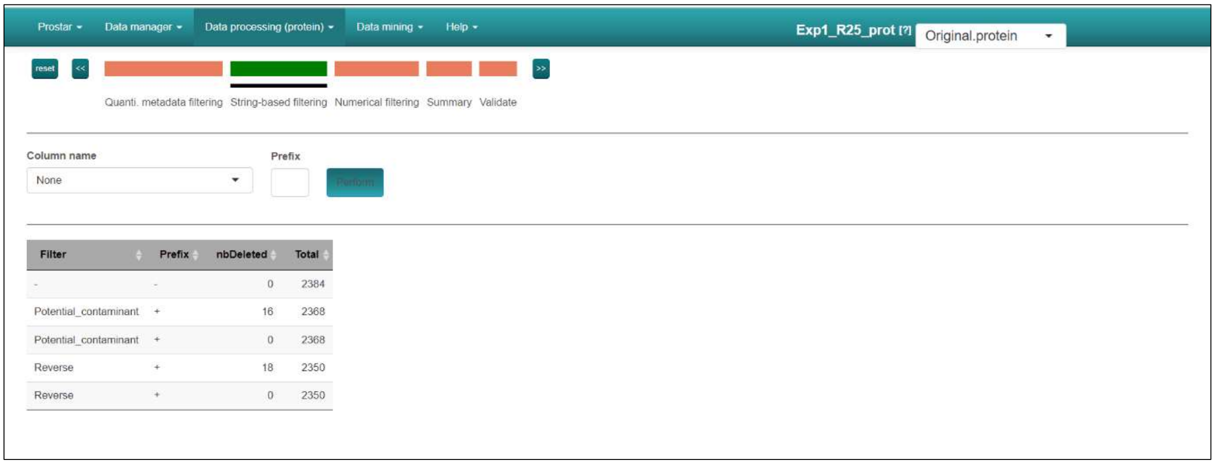
Column name: None

Prefix:

| Filter | Prefix | nbDeleted | Total |
|---|---|---|---|
| - | - | 0 | 2384 |
| Potential_contaminant | + | 16 | 2368 |
| Potential_contaminant | + | 0 | 2368 |

Figure 17. The table shows that 16 potential contaminants have been removed

7. If another filter must be applied, go back to Step 4, as shown in Figure 18.



Column name: None

Prefix:

| Filter | Prefix | nbDeleted | Total |
|---|---|---|---|
| - | - | 0 | 2384 |
| Potential_contaminant | + | 16 | 2368 |
| Potential_contaminant | + | 0 | 2368 |
| Reverse | + | 18 | 2350 |
| Reverse | + | 0 | 2350 |

Figure 18. Apply another filter on *Reverse* column.

8. Once all the filters have been applied, click on “Validate" tab to check the set of filtered out proteins.

9. Click on "Save filtered dataset".

10. The filtered dataset now appears as “Filtered.protein” below “Original.protein” on the upper right corner of the homepage as shown in Figure 19.

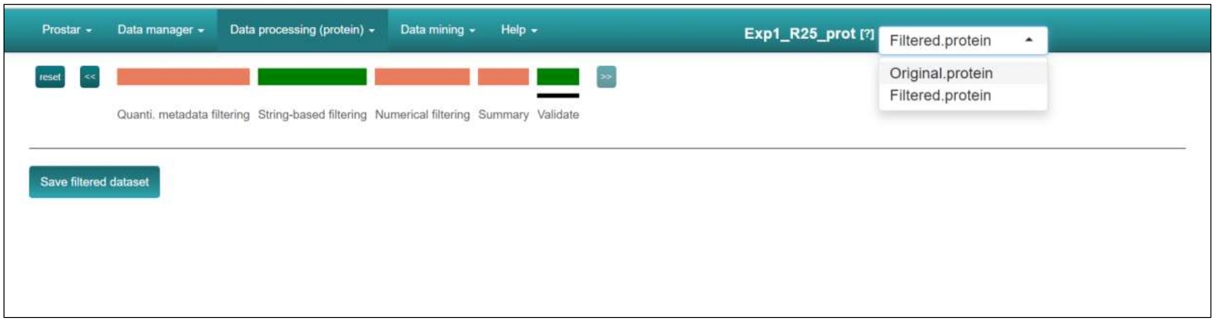


Figure 19. Save the filtered dataset

## Normalization

The next processing step proposed by ProStaR is data normalization. The objective is to reduce the biases introduced at any preliminary stage, for example, batch effects.

1. Prostar provides several methods of normalization as briefly described below. In this example, we have chosen quantile method for normalizing the dataset.

a. None: No normalization is applied

b. Global quantile alignment

c. Column sums

d. Quantile Centering

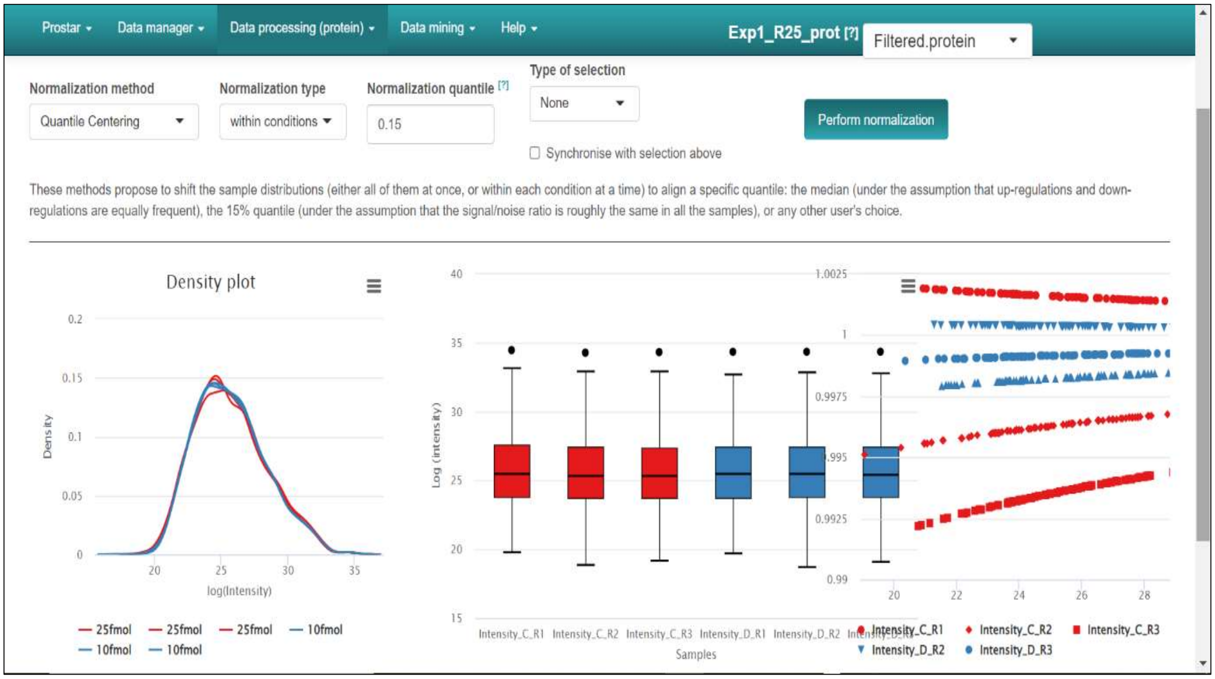
e. Mean Centering

f. Variance Stabilizing Normalization.

g. Loess normalization

2. Select "within conditions" as the "normalization type". This will normalize each condition independently of the others as shown in Figure 20.

3. Click on "Perform normalization". Three types of plots are generated as shown in Figure 20.



Normalization method: Quantile Centering

Normalization type: within conditions

Normalization quantile [?]: 0.15

Type of selection: None

☐ Synchronise with selection above

These methods propose to shift the sample distributions (either all of them at once, or within each condition at a time) to align a specific quantile: the median (under the assumption that up-regulations and down-regulations are equally frequent), the 15% quantile (under the assumption that the signal/noise ratio is roughly the same in all the samples), or any other user's choice.

Figure 20. Normalization-within conditions

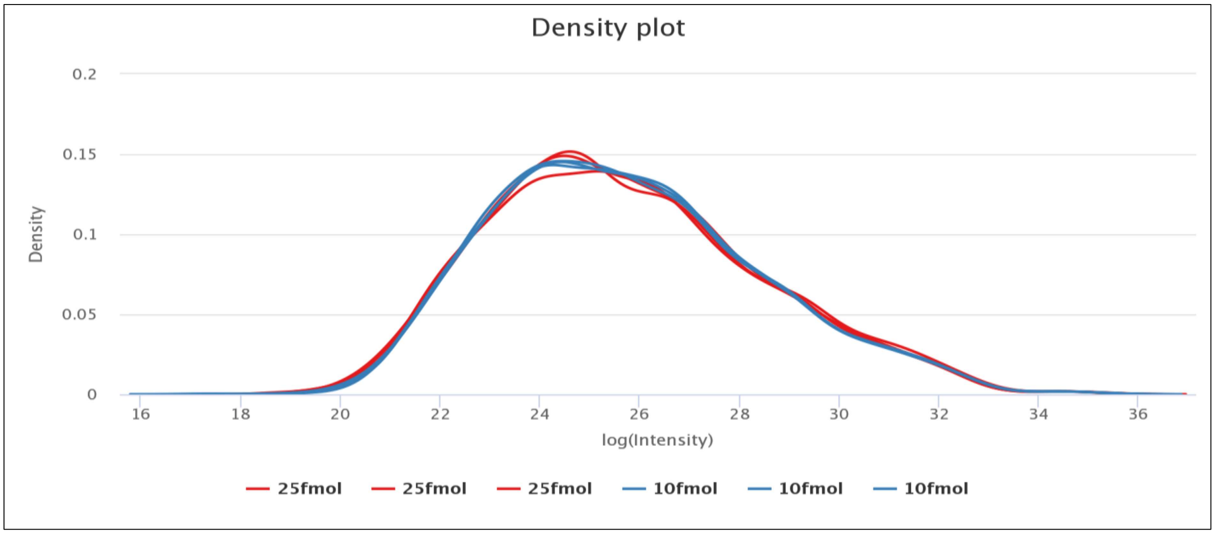


Figure 20. a) Density plot

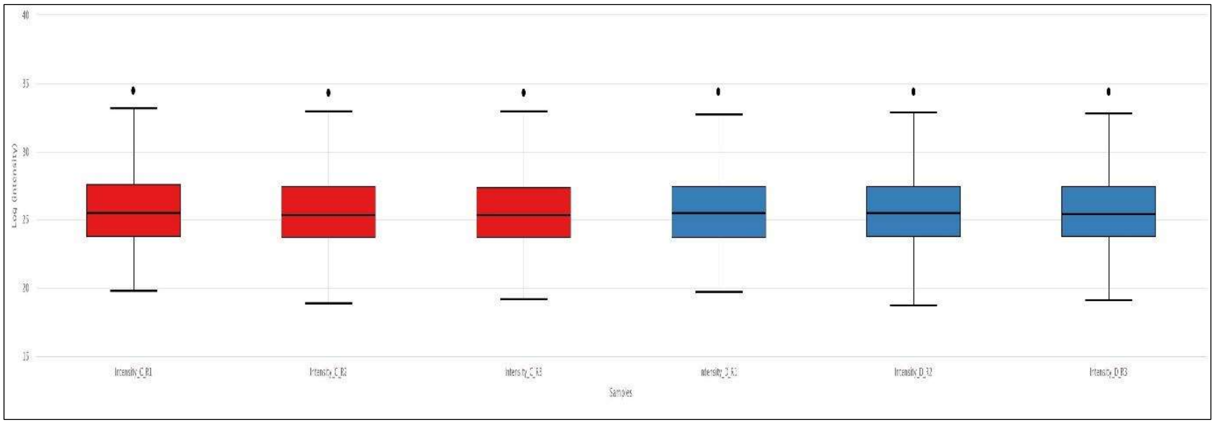


Figure 20. b) Box plot

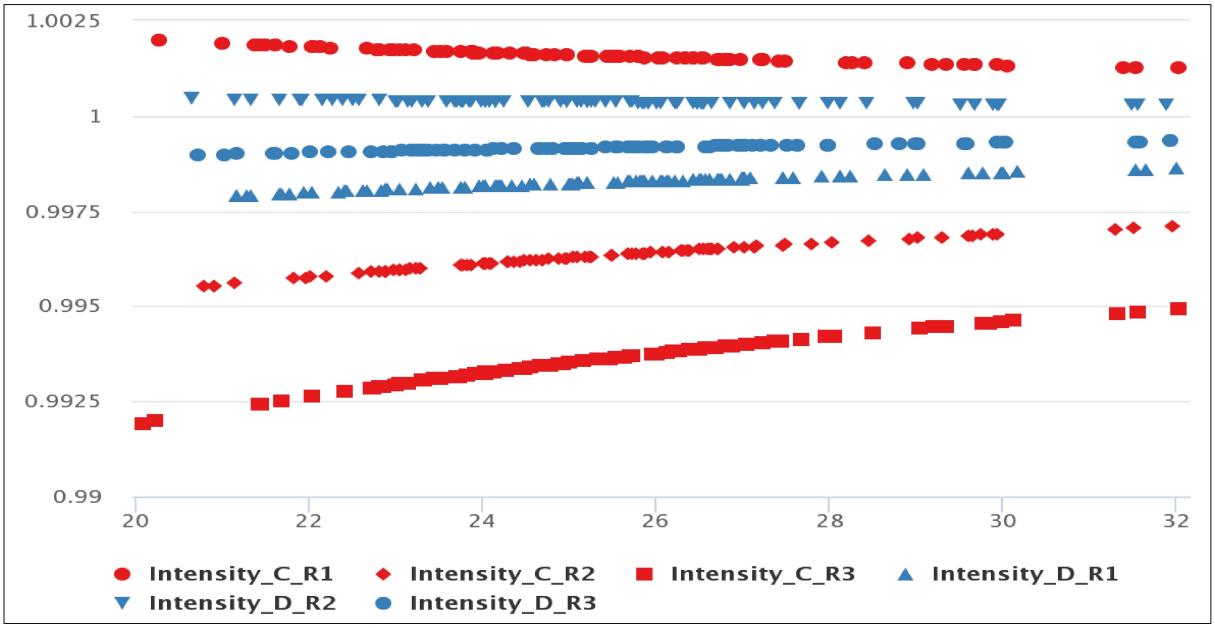


Figure 20. c) It shows the extent of change the dataset has to undergo in order to be normalized

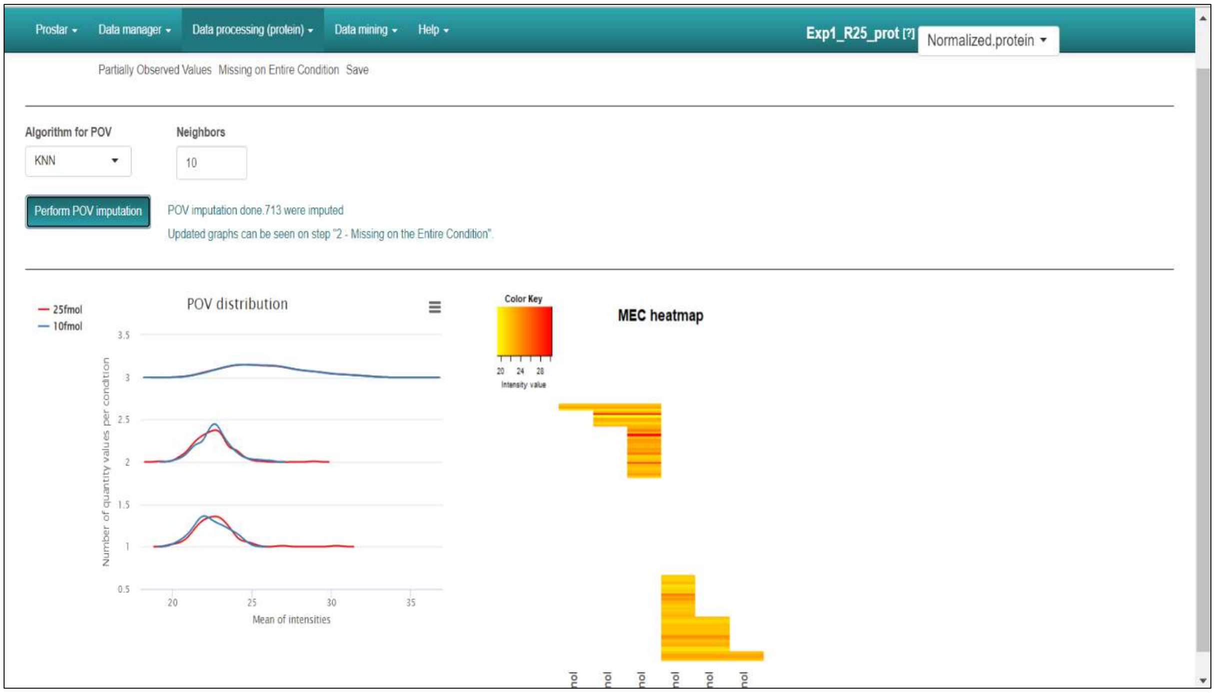
6. Click on “Save normalization”.

7. Normalized Protein” appears in the dataset version drop-down menu as the new version.

**Imputation**

ProStaR allows us to have separate processing for two different types of MVs, in protein-level datasets: POV (Partially Observed Value) and MEC (Missing in the Entire Condition). All the missing values for a given protein in a given condition are considered POV if and only if there is at least one observed value for this protein in this condition. And if all the intensity values are missing for this protein in this condition, the missing values are considered MEC.

1. On the first tab, select the algorithm to impute POV missing values. Here, we have selected the KNN and “neighbours” parameter as 10 which is also the default value but other methods are also of interest in specific situations as shown in Figure 21.



Algorithm for POV: KNN

Neighbors: 10

POV imputation done.713 were imputed

Updated graphs can be seen on step "2 - Missing on the Entire Condition".

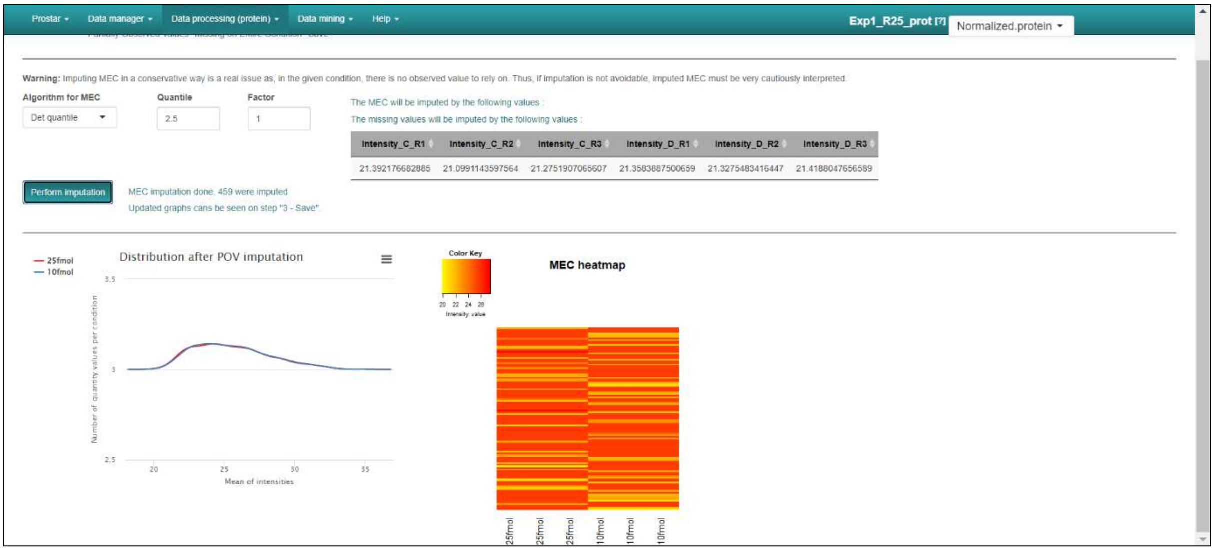
Figure 21. POV imputation

2. Tune the parameters of the chosen imputation method and the corresponding change in the plot will be visible as we change the algorithm and its associated parameters.

3. Click on “Perform Imputation”. It will enable the next tab, on which the result of the imputation is shown. In this example, 713 missing values were imputed.

4. Next, select appropriate method for MOV imputation. In this example we selected detQuantile algorithm.

6. Click on “Perform Imputation”. The result showed that 459 values were imputed as shown in Figure 22.



**Warning:** Imputing MEC in a conservative way is a real issue as, in the given condition, there is no observed value to rely on. Thus, if imputation is not avoidable, imputed MEC must be very cautiously interpreted.

Algorithm for MEC: Det quantile

Quantile: 2.5

Factor: 1

The MEC will be imputed by the following values :

The missing values will be imputed by the following values :

| Intensity_C_R1 | Intensity_C_R2 | Intensity_C_R3 | Intensity_D_R1 | Intensity_D_R2 | Intensity_D_R3 |
|---|---|---|---|---|---|
| 21.392176682885 | 21.0991143597564 | 21.2751907065607 | 21.3583887500659 | 21.3275483416447 | 21.4188047656589 |

MEC imputation done. 459 were imputed

Updated graphs cans be seen on step "3 - Save".

Figure 22. MOV imputation

8. The combined result showed that no more missing values are left to be imputed. Click on “Save imputation”.

9. “Imputed - Protein” appears in the dataset version dropdown menu as the updated version.

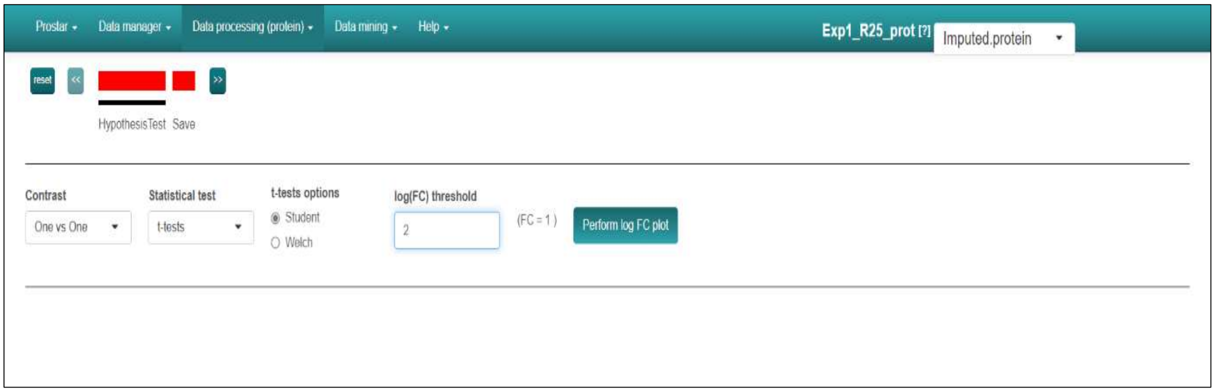
### Hypothesis testing

Once all the missing values have been imputed, the next step is to perform hypothesis testing in order to test whether each protein is significantly differentially abundant between the conditions. To do so, click on “Hypothesis testing” in the “Data processing” menu.

1. Choose the test contrasts. In this example since there are only two conditions, we selected one vs one.

2. Then, choose the type of statistical test, between limma or t-test (either Welch or Student).

3. Tune the log(FC) threshold value. In this example, we selected it to be 2 (FC=4) as shown in Figure 23.



Contrast: One vs One

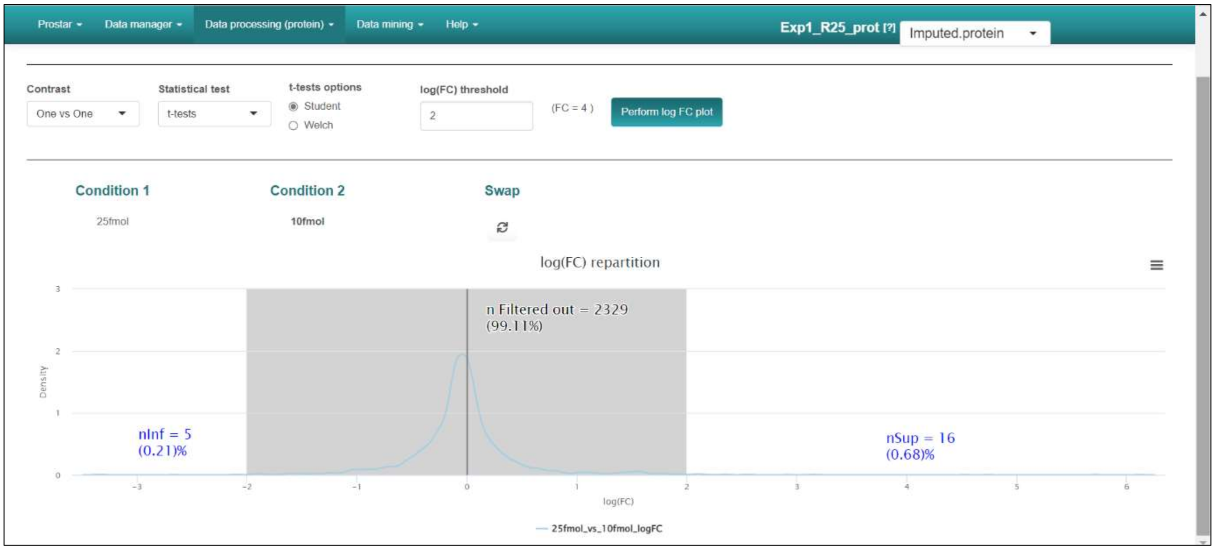
Statistical test: t-tests

t-tests options: ● Student ○ Welch

log(FC) threshold: 2 (FC = 1)

Figure 23. Parameters for hypothesis testing

4. Run the tests. This will generate density plot representing fold-change (FC) as shown in Figure 24. Save the dataset to preserve the results (i.e. all the computed p-values).



Imputed.protein

Contrast: One vs One

Statistical test: t-tests

t-tests options: ● Student ○ Welch

log(FC) threshold: 2 (FC = 4 )

Condition 1: 25fmol

Condition 2: 10fmol

Swap

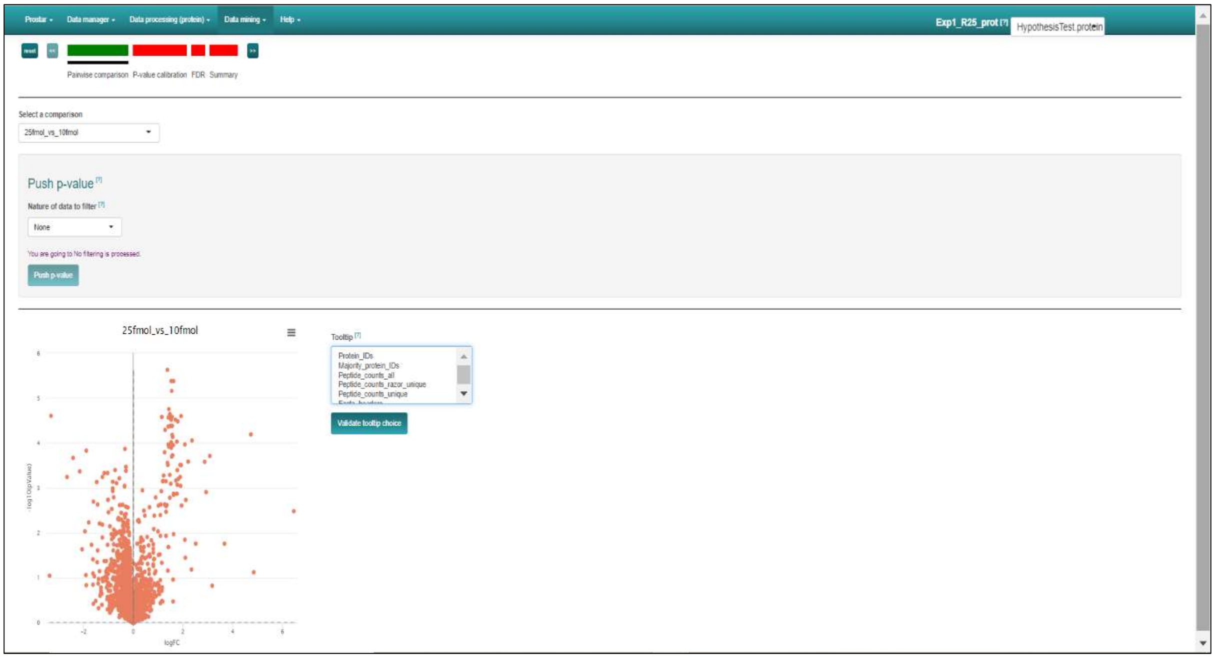
Figure 24. Density plot

5. “Hypothesized protein” will now appear as the new version of dataset. Then, this new dataset, contains the p-values and FC cut-off for the desired contrasts, which now can be explored in the “Differential analysis” tabs available in the “Data mining” menu.

## Differential Analysis

Click on “Differential analysis” in the “Data mining” menu to analyze the results of all statistical tests.

1. Select a pairwise comparison of interest from the dropdown menu. In this example it is “25fmol vs 10fmol”. The corresponding volcano plot is displayed as shown in Figure 25.



HypothesisTest.protein

Select a comparison: 25fmol_vs_10fmol

Push p-value [?]

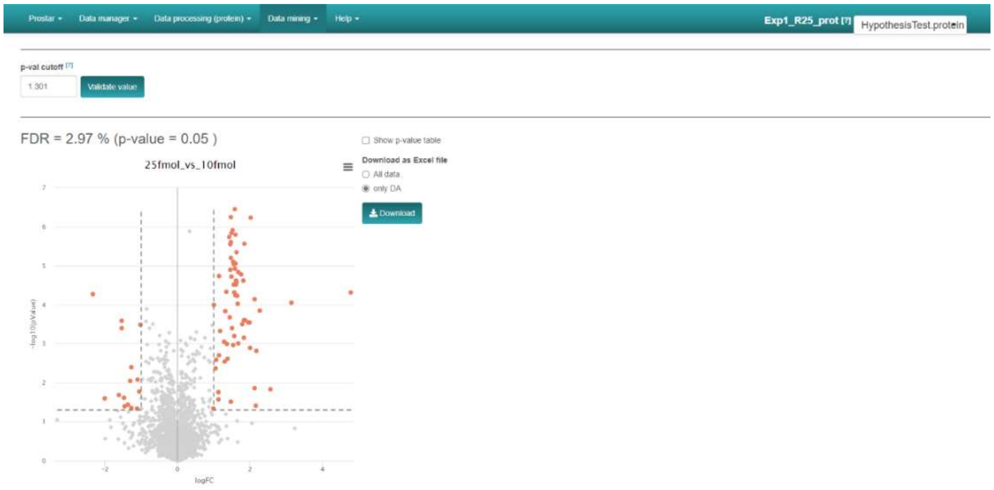
Nature of data to filter [?]: None

You are going to No filtering is processed.

Tooltip [?]: Protein_IDs, Majority_protein_IDs, Peptide_counts_all, Peptide_counts_razor_unique, Peptide_counts_unique, Fasta_headers

Figure 25. Volcano plot

2. Click on the next tab for adjusting the FDR threshold.
3. Save the result which is shown in Figure 26.



p-val cutoff [?]

1.301

FDR = 2.97 % (p-value = 0.05 )

□ Show p-value table

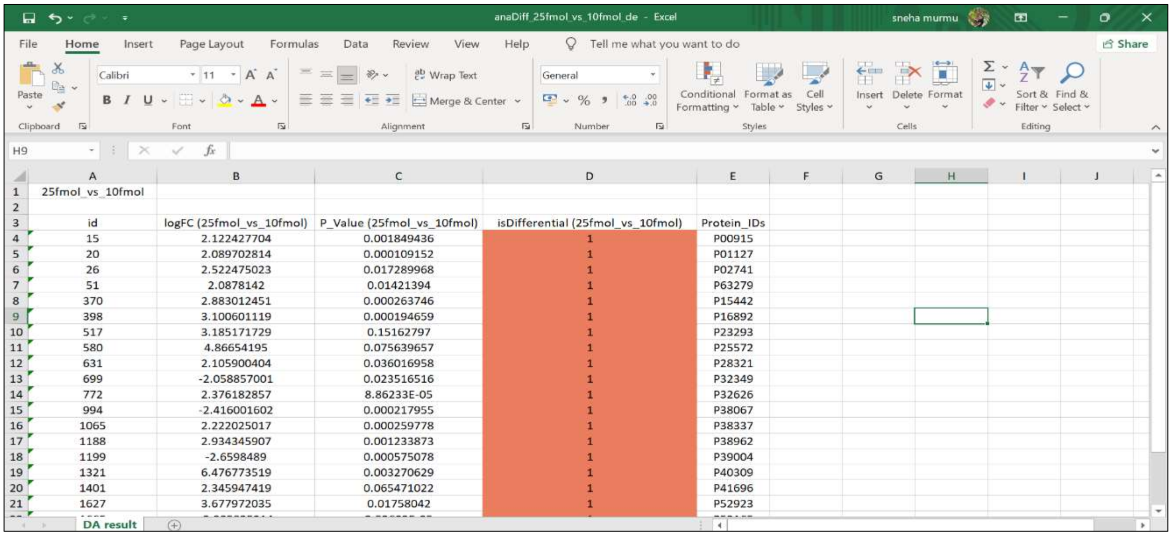
Download as Excel file

○ All data

● only DA

Figure 26. Result highlighting the proteins which have FC equal to 4 after adjusting FDR.

4. The list of differentially expressed proteins can be downloaded in excel format as shown in Figure 27.



| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | 25fmol_vs_10fmol | | | | |
| 2 | | | | | |
| 3 | id | logFC (25fmol_vs_10fmol) | P_Value (25fmol_vs_10fmol) | isDifferential (25fmol_vs_10fmol) | Protein_IDs |
| 4 | 15 | 2.122427704 | 0.001849436 | 1 | P00915 |
| 5 | 20 | 2.089702814 | 0.000109152 | 1 | P01127 |
| 6 | 26 | 2.522475023 | 0.017289968 | 1 | P02741 |
| 7 | 51 | 2.0878142 | 0.01421394 | 1 | P63279 |
| 8 | 370 | 2.883012451 | 0.000263746 | 1 | P15442 |
| 9 | 398 | 3.100601119 | 0.000194659 | 1 | P16892 |
| 10 | 517 | 3.185171729 | 0.15162797 | 1 | P23293 |
| 11 | 580 | 4.86654195 | 0.075639657 | 1 | P25572 |
| 12 | 631 | 2.105900404 | 0.036016958 | 1 | P28321 |
| 13 | 699 | -2.058857001 | 0.023516516 | 1 | P32349 |
| 14 | 772 | 2.376182857 | 8.86233E-05 | 1 | P32626 |
| 15 | 994 | -2.416001602 | 0.000217955 | 1 | P38067 |
| 16 | 1065 | 2.222025017 | 0.000259778 | 1 | P38337 |
| 17 | 1188 | 2.934345907 | 0.001233873 | 1 | P38962 |
| 18 | 1199 | -2.6598489 | 0.000575078 | 1 | P39004 |
| 19 | 1321 | 6.476773519 | 0.003270629 | 1 | P40309 |
| 20 | 1401 | 2.345947419 | 0.065471022 | 1 | P41696 |
| 21 | 1627 | 3.677972035 | 0.01758042 | 1 | P52923 |

Figure 27. List of differentially expressed protein.

## References

Glaab E, Schneider R (2015). RepExplore: addressing technical replicate variance in proteomics and metabolomics data analysis. *Bioinformatics*, **31(13)**, 2235-7.

Wieczorek, S., Combes, F., Lazar, C., Giai Gianetto, Q., Gatto, L., Dorffer, A., Hesse, A.-M., Couté, Y., Ferro, M., Bruley, C., & Burger, T. (2017). DAPAR & ProStaR: software to perform statistical analyses in quantitative discovery proteomics. *Bioinformatics* (Oxford, England), **33(1)**, 135-136. https://doi.org/10.1093/bioinformatics/btw580

Wieczorek, S., Combes, F., Borges, H., & Burger, T. (2019). Protein-level statistical analysis of quantitative label-free proteomics data with ProStaR. *Methods Mol Biol*., 1959:225-246. doi: 10.1007/978-1-4939-9164-8_15.

# Overview of Post-Translational Modifications

**Monendra Grover**

**ICAR-Indian Agricultural Statistics Research Institute, New Delhi**

Posttranslational modifications (PTMs) of proteins greatly expand proteome diversity, increase functionality, and allow for rapid responses, all at relatively low costs for the cell. PTMs play key roles in plants through their impact on signaling, gene expression, protein stability and interactions, and enzyme kinetics. Following a brief discussion of the experimental and bioinformatics challenges of PTM identification, localization, and quantification (occupancy), a concise overview is provided of the major PTMs and their (potential) functional consequences in plants, with emphasis on plant metabolism. Classic examples that illustrate the regulation of plant metabolic enzymes and pathways by PTMs and their cross talk are summarized. Recent large-scale proteomics studies mapped many PTMs to a wide range of metabolic functions. Unraveling of the PTM code, i.e. a predictive understanding of the (combinatorial) consequences of PTMs, is needed to convert this growing wealth of data into an understanding of plant metabolic regulation.

The primary amino acid sequence of proteins is defined by the translated mRNA, often followed by N- or C-terminal cleavages for preprocessing, maturation, and/or activation. Proteins can undergo further reversible or irreversible posttranslational modifications (PTMs) of specific amino acid residues. Proteins are directly responsible for the production of plant metabolites because they act as enzymes or as regulators of enzymes. Ultimately, most proteins in a plant cell can affect plant metabolism (e.g. through effects on plant gene expression, cell fate and development, structural support, transport, etc.). Many metabolic enzymes and their regulators undergo a variety of PTMs, possibly resulting in changes in oligomeric state, stabilization/degradation, and (de)activation (Huber and Hardin, 2004), and PTMs can facilitate the optimization of metabolic flux. However, the direct in vivo consequence of a PTM on a metabolic enzyme or pathway is frequently not very clear, in part because it requires measurements of input and output of the reactions, including flux through the enzyme or pathway.

PTMs can occur spontaneously (nonenzymatically) due to the physical-chemical properties of reactive amino acids and the cellular environment (e.g. pH, oxygen, reactive oxygen species [ROS], and metabolites) or through the action of specific modifying enzymes

PTMs are also determined by neighboring residues and their exposure to the surface. The 20 amino acids differ strongly in their general chemical reactivity and their observed PTMs . In particular, Cys and Lys can each carry many types of PTMs, whereas the N-terminal residue of proteins is also prone to multiple PTMs, ranging from N-terminal cleavage to N-terminal lipid modifications (acylation), acetylation, and ubiquitination . In addition to these PTMs that occur in vivo and presumably have physiological significance, several PTMs are often generated during sample preparation due to exposure to organic solvents (e.g. formic acid leading to the formylation of Ser, Thr, and N termini), (thio) urea (N-terminal or Lys carbamylation), reducing agents and oxygen, unpolymerized acrylamide (Cys propionamide), and low or high pH (cyclization of N-terminal Gln or Glu into pyro-Glu;). A large-scale proteomics study of Arabidopsis (Arabidopsis thaliana) leaf extracts did address the frequency of PTMs that do not require specific affinity enrichment based on a data set of 1.5 million tandem mass spectrometry (MS/MS) spectra acquired at 100,000 resolution on an LTQ-Orbitrap instrument followed by error-tolerant searches and systematic validation by liquid chromatography retention time . This revealed, for example, that modification of Met and N-terminal Gln into oxidized Met and pyro-Glu, respectively, showed by far the highest modification frequencies, followed by N-terminal formylation, most likely induced during sample analysis, as well as deamidation of Asn/Gln (approximately 1.2% of all observed Asn/Gln). Several of these nonenzymatic PTMs (in particular deamidation, oxidation, and formylation) can also occur in vivo and, therefore, cannot be simply dismissed as artifacts but need to be considered as potential regulators.

Since many PTMs are reversible, specific residues can also alternate between PTMs (e.g. dependent on cellular conditions, protein configuration [folding], or protein-protein interactions), and one PTM can influence the generation of other PTMs. This can result in an explosion of possible proteoforms and in cross talk between PTMs occurring on the same protein. Cross talk between PTMs on the same protein can coordinately determine the activity, function, and/or interactions of a protein. Finally, cross talk also exists between PTMs located on interacting proteins. Time-resolved and quantitative determination of combinatorial PTMs is challenging, and understanding of the biological outcomes is only in its infancy. Prominent examples of PTM cross talk are Lys ubiquitination and acetylation or Lys ubiquitination and phosphorylation . Phosphorylation can also directly promote substrate proteolysis by caspase (peptidase) during apoptosis. Recent biochemical and proteomics studies suggested that most if not all enzymes of the Calvin-Benson cycle undergo redox regulation through selective redox PTMs, including reversible disulfide bond formation, glutathionylation, and nitrosylation, with an interplay between these PTMs dependent on (sub)cellular conditions . Moreover, the regulators carrying out these PTMs (e.g. thioredoxins, glutaredoxins, etc.) themselves can also undergo some of these PTMs, making for a complex network of PTMs

The identification, localization, and quantification of different combinations of PTMs on the same protein can sometimes be better solved by so-called top-down or middle-down proteomics, as opposed to the more common bottom-up proteomics (. or chemical cleavage) prior to MS analysis. In contrast, in top-down proteomics, proteins are not digested into smaller fragments but rather injected and analyzed by a specialized mass spectrometer in its entirety. In middle-down proteomics, the intact proteins are cleaved into just a few fragments by limited proteolysis prior to injection into the mass spectrometer. Top-down and middle-down proteomics are not high throughput and are best carried out on either purified proteins or protein mixtures of low complexity. Classic examples of studies using top-down, middle-down, but also bottom-up proteomics on proteins with different PTMs involve histones) and the p53 tumor suppression protein.

## References

Agetsuma M, Furumoto T, Yanagisawa S, Izui K (2005) The ubiquitin-proteasome pathway is involved in rapid degradation of phosphoenolpyruvate carboxylase kinase for C4 photosynthesis. Plant Cell Physiol **46**: 389–398.

Akter S, Huang J, Waszczak C, Jacques S, Gevaert K, Van Breusegem F, Messens J (2015) Cysteines under ROS attack in plants: a proteomics view. J Exp Bot **66**: 2935–2944.

Alban C, Tardif M, Mininno M, Brugière S, Gilgen A, Ma S, Mazzoleni M, Gigarel O, Martin-Laffon J, Ferro M, et al. (2014) Uncovering the protein lysine and arginine methylation network in Arabidopsis chloroplasts. PLoS One **9**: e95512.

Altelaar AF, Munoz J, Heck AJ (2013) Next-generation proteomics: towards an integrative view of proteome dynamics. Nat Rev Genet **14**: 35–48.

Bailey KJ, Gray JE, Walker RP, Leegood RC (2007) Coordinate regulation of phosphoenolpyruvate carboxylase and phospho*enol*pyruvate carboxykinase by light and $CO_2$ during $C_4$ photosynthesis. Plant Physiol **144**: 479–486.

Balmer Y, Vensel WH, Tanaka CK, Hurkman WJ, Gelhaye E, Rouhier N, Jacquot JP, Manieri W, Schürmann P, Droux M, et al. (2004) Thioredoxin links redox to the regulation of fundamental processes of plant mitochondria. Proc Natl Acad Sci USA **101**: 2642–2647.

Balsera M, Uberegui E, Schürmann P, Buchanan BB (2014) Evolutionary development of redox regulation in chloroplasts. Antioxid Redox Signal **21**: 1327–1355.

Banerjee A, Sharkey TD (2014) Methylerythritol 4-phosphate (MEP) pathway metabolic regulation. Nat Prod Rep **31**: 1043–1055.

Barberon M, Zelazny E, Robert S, Conéjéro G, Curie C, Friml J, Vert G (2011) Monoubiquitin-dependent endocytosis of the iron-regulated transporter 1 (IRT1) transporter controls iron uptake in plants. Proc Natl Acad Sci USA **108**: E450–E458.

Bartel B, Citovsky V (2012) Focus on ubiquitin in plant biology. Plant Physiol **160**: 1.

Bartsch O, Mikkat S, Hagemann M, Bauwe H (2010) An autoinhibitory domain confers redox regulation to maize glycerate kinase. Plant Physiol **153**: 832–840.

Berr A, Shafiq S, Shen WH (2011) Histone modifications in transcriptional activation during plant development. Biochim Biophys Acta **1809**: 567–576.

Bigeard J, Rayapuram N, Pflieger D, Hirt H (2014) Phosphorylation-dependent regulation of plant chromatin and chromatin-associated proteins. Proteomics **14**: 2127–2140.

Biggar KK, Li SS (2015) Non-histone protein methylation as a regulator of cellular signalling and function. Nat Rev Mol Cell Biol **16**: 5–17.

Bonissone S, Gupta N, Romine M, Bradshaw RA, Pevzner PA (2013) N-terminal protein processing: a comparative proteogenomic analysis. Mol Cell Proteomics **12**: 14–28.

Borner GH, Lilley KS, Stevens TJ, Dupree P (2003) Identification of glycosylphosphatidylinositol-anchored proteins in Arabidopsis: a proteomic and genomic analysis. Plant Physiol **132**: 568–577.

Boyle PC, Martin GB (2015) Greasy tactics in the plant-pathogen molecular arms race. J Exp Bot **66**: 1607–1616.

Bracha-Drori K, Shichrur K, Lubetzky TC, Yalovsky S (2008) Functional analysis of Arabidopsis postprenylation CaaX processing enzymes and their function in subcellular protein targeting. Plant Physiol **148**: 119–131.

Brzezowski P, Richter AS, Grimm B (2015) Regulation and function of tetrapyrrole biosynthesis in plants and algae. Biochim Biophys Acta **1847**: 968–985.

Carlson SM, Gozani O (2014) Emerging technologies to map the protein methylome. J Mol Biol **426**: 3350–3362.

Catherman AD, Skinner OS, Kelleher NL (2014) Top down proteomics: facts and perspectives. Biochem Biophys Res Commun **445**: 683–693.

Cavazzini D, Meschi F, Corsini R, Bolchi A, Rossi GL, Einsle O, Ottonello S (2013) Autoproteolytic activation of a symbiosis-regulated truffle phospholipase A2. J Biol Chem **288**: 1533–1547.

Černý M, Skalák J, Cerna H, Brzobohatý B (2013) Advances in purification and separation of posttranslationally modified proteins. J Proteomics **92**: 2–27.

Chalkley RJ, Bandeira N, Chambers MC, Clauser KR, Cottrell JS, Deutsch EW, Kapp EA, Lam HH, McDonald WH, Neubert TA, et al. (2014) Proteome informatics research group (iPRG)_2012: a study on detecting modified peptides in a complex mixture. Mol Cell Proteomics **13**: 360–371.

Chalkley RJ, Clauser KR (2012) Modification site localization scoring: strategies and performance. Mol Cell Proteomics **11**: 3–14.

Champion A, Kreis M, Mockaitis K, Picaud A, Henry Y (2004) Arabidopsis kinome: after the casting. Funct Integr Genomics **4**: 163–187.

Chastain CJ, Failing CJ, Manandhar L, Zimmerman MA, Lakner MM, Nguyen TH (2011) Functional evolution of C(4) pyruvate, orthophosphate dikinase. J Exp Bot **62**: 3083–3091.

Chen YB, Lu TC, Wang HX, Shen J, Bu TT, Chao Q, Gao ZF, Zhu XG, Wang YF, Wang BC (2014) Posttranslational modification of maize chloroplast pyruvate orthophosphate dikinase reveals the precise regulatory mechanism of its enzymatic activity. Plant Physiol **165**: 534–549.

Choudhary C, Weinert BT, Nishida Y, Verdin E, Mann M (2014) The growing landscape of lysine acetylation links metabolism and cell signalling. Nat Rev Mol Cell Biol **15**: 536–550.

Christian JO, Braginets R, Schulze WX, Walther D (2012) Characterization and prediction of protein phosphorylation hotspots in Arabidopsis thaliana. Front Plant Sci **3**: 207.

Cieśla J, Frączyk T, Rode W (2011) Phosphorylation of basic amino acid residues in proteins: important but easily missed. Acta Biochim Pol **58**: 137–148.

Cox J, Mann M (2011) Quantitative, high-resolution proteomics for data-driven systems biology. Annu Rev Biochem **80**: 273–299.

Czyzewicz N, Yue K, Beeckman T, De Smet I (2013) Message in a bottle: small signalling peptide outputs during growth and development. J Exp Bot **64**: 5281–5296.

Daloso DM, Müller K, Obata T, Florian A, Tohge T, Bottcher A, Riondet C, Bariat L, Carrari F, Nunes-Nesi A, et al. (2015) Thioredoxin, a master regulator of the tricarboxylic acid cycle in plant mitochondria. Proc Natl Acad Sci USA **112**: E1392–E1400.

de Boer AH, van Kleeff PJ, Gao J (2013) Plant 14-3-3 proteins as spiders in a web of phosphorylation. Protoplasma **250**: 425–440.

DeHart CJ, Chahal JS, Flint SJ, Perlman DH (2014) Extensive post-translational modification of active and inactivated forms of endogenous p53. Mol Cell Proteomics **13**: 1–17.

Denison FC, Paul AL, Zupanska AK, Ferl RJ (2011) 14-3-3 proteins in plant physiology. Semin Cell Dev Biol **22**: 720–727.

Dietz KJ, Hell R (2015) Thiol switches in redox regulation of chloroplasts: balancing redox state, metabolism and oxidative stress. Biol Chem **396**: 483–494.

Dinh TV, Bienvenut WV, Linster E, Feldman-Salit A, Jung VA, Meinnel T, Hell R, Giglione C, Wirtz M (2015) Molecular identification and functional characterization of the first Nα-acetyltransferase in plastids by global acetylome profiling. Proteomics **15**: 2426–2435.

di Pietro M, Vialaret J, Li GW, Hem S, Prado K, Rossignol M, Maurel C, Santoni V (2013) Coordinated post-translational responses of aquaporins to abiotic and nutritional stimuli in Arabidopsis roots. Mol Cell Proteomics **12**: 3886–3897.

Dix MM, Simon GM, Wang C, Okerberg E, Patricelli MP, Cravatt BF (2012) Functional interplay between caspase cleavage and phosphorylation sculpts the apoptotic proteome. Cell **150**: 426–440.

Dong L, Ermolova NV, Chollet R (2001) Partial purification and biochemical characterization of a heteromeric protein phosphatase 2A holoenzyme from maize (Zea mays L.) leaves that dephosphorylates C4 phosophoenolpyruvate carboxylase. Planta **213**: 379–389.

Duncan KA, Huber SC (2007) Sucrose synthase oligomerization and F-actin association are regulated by sucrose concentration and phosphorylation. Plant Cell Physiol **48**: 1612–1623.

Elortza F, Mohammed S, Bunkenborg J, Foster LJ, Nühse TS, Brodbeck U, Peck SC, Jensen ON (2006) Modification-specific proteomics of plasma membrane proteins: identification and characterization of glycosylphosphatidylinositol-anchored proteins released upon phospholipase D treatment. J Proteome Res **5**: 935–943.

Elrouby N, Coupland G (2010) Proteome-wide screens for small ubiquitin-like modifier (SUMO) substrates identify Arabidopsis proteins implicated in diverse biological processes. Proc Natl Acad Sci USA **107**: 17415–17420.

Engineer CB, Ghassemian M, Anderson JC, Peck SC, Hu H, Schroeder JI (2014) Carbonic anhydrases, EPF2 and a novel protease mediate $CO_2$ control of stomatal development. Nature **513**: 246–250.

Fedorova M, Bollineni RC, Hoffmann R (2014) Protein carbonylation as a major hallmark of oxidative damage: update of analytical strategies. Mass Spectrom Rev **33**: 79–97.

Fedosejevs ET, Ying S, Park J, Anderson EM, Mullen RT, She YM, Plaxton WC (2014) Biochemical and molecular characterization of RcSUS1, a cytosolic sucrose synthase phosphorylated in vivo at serine 11 in developing castor oil seeds. J Biol Chem **289**: 33412–33424.

Ferrández-Ayela A, Micol-Ponce R, Sánchez-García AB, Alonso-Peral MM, Micol JL, Ponce MR (2013) Mutation of an Arabidopsis NatB N-alpha-terminal acetylation complex component causes pleiotropic developmental defects. PLoS One **8**: e80697.

Finkemeier I, Laxa M, Miguet L, Howden AJ, Sweetlove LJ (2011) Proteins of diverse function and subcellular location are lysine acetylated in Arabidopsis. Plant Physiol **155**: 1779–1790.

Gao ZP, Chen GX, Yang ZN (2012) Regulatory role of Arabidopsis pTAC14 in chloroplast development and plastid gene expression. Plant Signal Behav **7**: 1354–1356.

Geigenberger P. (2011) Regulation of starch biosynthesis in response to a fluctuating environment. Plant Physiol **155**: 1566–1577.

Geigenberger P, Kolbe A, Tiessen A (2005) Redox regulation of carbon storage and partitioning in response to light and sugars. J Exp Bot **56**: 1469–1479.

Giglione C, Fieulaine S, Meinnel T (2015) N-terminal protein modifications: bringing back into play the ribosome. Biochimie **114**: 134–146.

Graciet E, Lebreton S, Gontero B (2004) Emergence of new regulatory mechanisms in the Benson-Calvin pathway via protein-protein interactions: a glyceraldehyde-3-phosphate dehydrogenase/CP12/phosphoribulokinase complex. J Exp Bot **55**: 1245–1254.

Grimaud F, Rogniaux H, James MG, Myers AM, Planchot V (2008) Proteome and phosphoproteome analysis of starch granule-associated proteins from normal maize and mutants affected in starch biosynthesis. J Exp Bot **59**: 3395–3406.

Guerra DD, Callis J (2012) Ubiquitin on the move: the ubiquitin modification system plays diverse roles in the regulation of endoplasmic reticulum- and plasma membrane-localized proteins. Plant Physiol **160**: 56–64.

Haag F, Buck F (2015) Identification and analysis of ADP-ribosylated proteins. Curr Top Microbiol Immunol **384**: 33–50.

Hang R, Liu C, Ahmad A, Zhang Y, Lu F, Cao X (2014) Arabidopsis protein arginine methyltransferase 3 is required for ribosome biogenesis by affecting precursor ribosomal RNA processing. Proc Natl Acad Sci USA **111**: 16190–16195.

Havelund JF, Thelen JJ, Møller IM (2013) Biochemistry, proteomics, and phosphoproteomics of plant mitochondria from non-photosynthetic cells. Front Plant Sci **4**: 51.

Hemsley PA. (2014) Progress in understanding the mechanisms and functional importance of protein-membrane interactions in plants. New Phytol **204**: 741–743.

Hemsley PA. (2015) The importance of lipid modified proteins in plants. New Phytol **205**: 476–489.

Hemsley PA, Weimar T, Lilley K, Dupree P, Grierson C (2013a) Palmitoylation in plants: new insights through proteomics. Plant Signal Behav **8**: 8.

Hemsley PA, Weimar T, Lilley KS, Dupree P, Grierson CS (2013b) A proteomic approach identifies many novel palmitoylated proteins in Arabidopsis. New Phytol **197**: 805–814.

Heyl A, Brault M, Frugier F, Kuderova A, Lindner AC, Motyka V, Rashotte AM, Schwartzenberg KV, Vankova R, Schaller GE (2013) Nomenclature for members of the two-component signaling pathway of plants. Plant Physiol **161**: 1063–1065.

Hodges M, Jossier M, Boex-Fontvieille E, Tcherkez G (2013) Protein phosphorylation and photorespiration. Plant Biol (Stuttg) **15**: 694–706.

# Genomics Approaches to Investigate Plant Structure and Function: Case Studies with Photosynthesis and Environmental Signaling

**Aashish Ranjan**

**National Institute of Plant Genome Research, New Delhi**

Omics approaches, such as next-generation sequencing in combination with genomics, transcriptomics, and bioinformatics, have facilitated global insights into the genome and transcriptome to address specific biological questions relating to structure and function in different model and non-model plant species. The lecture will involve a detailed presentation on usage of usage of integrated transcriptomics and genomics approaches to understand the genetic insights of plant development and physiology of both non-model as well as model organisms.

Transcriptomics approach was used to decipher the genetic basis of plant parasitism of an obligate stem plant parasite *Cuscuta pentagona* (Dodder). Parasitic plants, one of the most destructive agricultural pests, penetrate and establish vascular connections through specialized organs called haustoria to steal nutrients and water from host plants. Dodder transcriptome was *de novo* assembled using RNAseq reads from multiple tissues and stages. Subsequent gene expression analysis and dissection of transcriptional dynamics across the stages identified key genes and gene categories, such as plant defense and transporter genes, involved in the process of plant parasitism (Ranjan et al., 2014). Similarly, transcriptomics deciphered the molecular and genetic basis of patterning in one of the largest unicellular coenocytic alga, *Caulerpa taxifolia*, with distinct functional pseudo-organs. The study not only revealed a global, apical-basal pattern of the transcript distribution across the algal body but also demonstrated the contribution of transcript partitioning to morphology in plants (Ranjan et al., 2015). In addition, the genetical genomics approach to investigate the genetic architecture of gene expression in a model plant tomato will also be briefly discussed. Using an introgression population developed from a wild and a domesticated tomato, more than 7000 expression QTL (eQTL) regulating global gene expression patterns in tomato were identified. Moreover, several genetic hotspots regulating gene expression patterns relating to diverse biological processes such as plant development, photosynthesis, and defense were also identified (Ranjan et al., 2016).

The current trends of population growth and the availability of limited agricultural land and resources have raised serious concerns regarding food security. The exponential increase in population and rapid global environmental changes observed in recent years are serious threats to sustainable food production for the planet (Lobell et al, 2012). Reducing agricultural land and environmental changes further compound the requirement for increasing crop yield and productivity. Developing crop varieties in order to achieve greater yields has been a major focus of plant biologists and breeders with a view to ensuring food availability for an increasing world population under changing environmental conditions (Long et al., 2015; Zhu et al., 2010). Innovative genomics approaches could be instrumental in achieving sustainable increases in crop yield and productivity in the wake of climate change. During the talk, the basic concepts of genomics as well as their usage for investigating plant structure and function and in crop improvement programs will be discussed. Moreover, large-scale data analysis to investigate the effects of environmental effects on crop developmental features will be discussed. The utilization of the natural variation in leaf features and photochemical and biochemical traits for increasing crop photosynthetic efficiency will also be discussed.

Developing crop varieties in order to achieve greater yields has been a major focus of plant biologists and breeders with a view to ensure sustainable food availability for an increasing world population under changing environmental conditions. The optimization of plant developmental traits has great potential for a sustainable increase in crop yield, as plant performance is strongly associated with, and dependent on, plant development and growth (Mathan et al., 2016). Increasing photosynthetic efficiency has now been realized as one of the promising strategies for improving crop yield and productivity. Knowing that leaves are the primary site of photosynthesis, optimizing leaf morphological and anatomical features could be instrumental in increasing crop photosynthetic efficiency. We are using genomics and transcriptomics approaches to harness the natural variation in rice photosynthesis to identify the genetic loci, genes, and gene-regulatory networks that could be used for improving photosynthetic efficiency, and thus yield, in crop improvement programs.

The natural genetic variation in leaf photosynthesis, and underlying developmental, biochemical, and genetic basis is an overlooked and untapped resource. The genus *Oryza*, which includes cultivated rice and more than 20 wild relatives, offers tremendous genetic variability to explore photosynthetic differences and underlying biochemical and developmental differences. Photosynthetically efficient wild rice accessions had specific developmental features, such as larger mesophyll cells with more chloroplasts, distribution of chloroplasts along the mesophyll cell wall, larger and closer veins, and a smaller number of mesophyll cells between two consecutive veins (Mathan et al., 2021). The wild species with higher photosynthesis also exhibited striking differences in leaf shape and size, as well as differences in Shoot Apical Meristem (SAM) size and leaf initiation rate. We are, currently, investigating the genetic basis of leaf developmental and biochemical differences that could be attributed to differences in photosynthesis. Leaf morphological traits, such as wider and thicker leaves, and anatomical features, such as mesophyll features and chloroplast surface area contribute to higher photosynthetic efficiency in wild rice accessions. The comparative transcriptomics approach has dissected the genetic basis of rice leaf size regulation. Differential gene expression analysis followed by Principal Component Analysis and a Self-organizing map identified the group of genes that may contribute to leaf size regulation (Jathar et al., 2022). The gene-expressions network analysis then identified the major regulators and downstream signals that control rice leaf size. The signalling module involves Gibberelic Acid, GRF transcription factors, and downstream cell-cycle components. A more comprehensive comparative transcriptomic comparison is being used to identify the genetic regulators of the transition from development to photosynthesis. A detailed biological as well as technical presentation of the usage of integrated transcriptomic analyses to dissect the genetic underpinnings of leaf development and photosynthesis will be discussed.

The usage of genomic approaches, complementary to transcriptomic approaches, strengthens the pursuit of the identification of genes and genetic loci regulating a trait. large-scale field phenotyping exhibited remarkable variation in leaf photosynthesis and related leaf physiological and developmental traits among cultivated Indian rice accessions. While comparative transcriptomics involving wild and cultivated rice identified genetic regulators of rice leaf size and transition from development to photosynthesis, GWAS with cultivated landraces identified the genetic loci regulating the desirable leaf developmental and physiological features. The GWAS results were analyzed to identify the relevant haploblocks and haplotypes contributing to the leaf photosynthesis and developmental differences across the rice accessions. These regulators could be targeted for increasing the photosynthetic efficiency of cultivated rice varieties.

In the last part of the lecture, comparative transcriptomic insights to understand the plant responses to changing light and temperature conditions will be discussed. Optimum light and temperature conditions are required to maximize the fitness of the plants. Shade and small rises in temperature are the inevitable threats to the fitness of the plant under changing climatic conditions. While shade- and temperature-induced elongation in

Arabidopsis via Phytochrome Interacting Factors (PIFs), members of bHLH-family transcription factors, is extensively studied, there is a limited understanding of comprehensive tissue-specific gene-regulatory networks involved in light and temperature responses in plants. Moreover, the genetic understanding of signaling and responses to shade and high temperature in crop plants is scarce. Therefore, we aimed not only to identify novel regulators of shade and high temperature signalling in Arabidopsis but also a comparative investigation of signaling and response across Arabidopsis, tomato, and rice. Organ-specific comparative transcriptome profiling revealed a more pronounced impact of high temperature on gene expression dynamics than the shade in all three species. Transcription, development, cell cycle, and hormonal responses were the major conserved biological pathways affected by shade and high temperature in all three species. Orthology overlap of shade- and high-temperature-regulated genes were used to identify conserved molecular networks and regulators for environmental signaling across the species. Detailed analyses of transcription factors suggested the involvement of novel regulators belonging to bZIP, NF-Y, CO-like, MYB, NAC, GATA, and Dof-family in the shade and high-temperature signaling in all the three species along with bHLH, HD-ZIP and TCP family previously reported for these signaling pathways. In summary, the comparative transcriptome analysis for shade and high temperature provides comprehensive information on shade and high temperature signaling across the three plant species and posits these as key transcriptional regulators mediating cell division, phytohormone signaling, cell wall and growth responses across evolutionarily different plant species that could be used to optimize plant growth in a changing environment.

Together, the lecture would underscore the importance of omics approaches and large-scale data analysis for not only establishing the comprehensive gene-regulatory modules and their interactions but also for identifying the key genetic regulators for informed usage in targeted crop improvement programs for increasing yield and productivity under changing climatic conditions.

## References:

- Jathar V, Saini K, Chauhan A, Rani R, Ichihashi Y, Ranjan A (2022). Spatial control of cell division by GA-OsGRF7/8 module in a leaf explaining the leaf length variation between cultivated and wild rice. *New Phytologist* 234(3):867-883.
- Lobell, D. B. and Gourdji, S. M. (2012). The influence of climate change on global crop productivity. *Plant Physiology* 160, 1686-1697.
- Long, S. P., Marshall-Colon, A. and Zhu, X.-G. (2015). Meeting the global food demand of the future by engineering crop photosynthesis and yield potential. *Cell* 161: 56-66.
- Mathan J, Bhattacharya J, Ranjan A (2016). Enhancing crop yield via the optimization of plant developmental features. *Development* 143: 3283-3294.
- Mathan, J., Singh, A., Jathar, V. and Ranjan, A. (2021). High photosynthesis rate in two wild rice species is driven by leaf anatomy mediating high Rubisco activity and electron transport rate. *Journal of Experimental Botany*, 72: 7119-7135.
- Ranjan A, Ichihashi Y, Farhi M, Zumstein K, Townsley BT, David-Schwrtz R, Sinha NR (2014). De novo assembly and characterization of the transcriptome of the parasitic weed *Cuscuta pentagona* identifies genes associated with plant parasitism. *Plant Physiology*. 166: 1186-1199.
- Ranjan A, Townsley BT, Ichihashi Y, Sinha NR, Chitwood DH (2015). An intracellular transcriptomic atlas of the giant coenocyte Caulerpa taxifolia. *PLoS Genetics*. 11(1): e1004900.
- Ranjan A, Budke JM, Rowland SD, Chitwood DH, Kumar R, Carriedo L, Ichihashi Y, Zumstein K, Maloof JN, Sinha NR (2016). eQTL in a Precisely Defined Tomato Introgression Population Reveal Genetic Regulation of Gene Expression Patterns Related to Physiological and Developmental Pathways. *Plant Physiology*. 172: 328-340.
- Zhu, X.-G., Long, S. P. and Ort, D. R. (2010). Improving photosynthetic efficiency for greater yield. *Annual Review of Plant Biology* 61: 235-261.

## @ Disclaimer

The information contained in this reference manual has been taken from various web resources. The information is provided by "ICAR-IASRI" and whilst we endeavour to keep the information up-to-date and correct, we make no representations or warranties of any kind, express or implied, about the completeness, accuracy, reliability, suitability, or availability with respect to the website or the information, products, services, or related graphics contained in the reference manual for any purpose. Any reliance you place on such information is therefore strictly at your own risk.

In no event we will be liable for any loss or damage including without limitation, indirect or consequential loss or damage, or any loss or damage whatsoever arising from loss of data or profits arise out of or in connection with the use of this manual. We have no control over the nature, content and availability of those sites. The inclusion of any links does not necessarily imply a recommendation or endorse the views expressed within them.

## @ Citation

Rajender Parsad, Girish Kumar Jha, Sudhir Srivastava and Neeraj Budhlakoti (2024). Statistical and Computational Advances for Bioinformatics Data Analysis in Agriculture: Practical Aspects, Centre for Advanced Faculty Training, Reference Manual, ICAR-Indian Agricultural Statistics Research Institute, New Delhi.