Mississippi State University
Scholars Junction

Theses and Dissertations | Theses and Dissertations

12-10-2021

Methods for inference and analysis of gene networks from RNA sequencing data

Himangi Srivastava
himprans10@gmail.com

Follow this and additional works at: https://scholarsjunction.msstate.edu/td

 Part of the Biomedical Commons

Recommended Citation

Srivastava, Himangi, "Methods for inference and analysis of gene networks from RNA sequencing data" (2021). *Theses and Dissertations*. 5352.
https://scholarsjunction.msstate.edu/td/5352

This Dissertation - Open Access is brought to you for free and open access by the Theses and Dissertations at Scholars Junction. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of Scholars Junction. For more information, please contact scholcomm@msstate.libanswers.com.

Methods for inference and analysis of gene networks from RNA sequencing data

By

Himangi Srivastava

Approved by:

Mehmet Kurum (Major Professor)
George V. Popescu (Minor Professor)
John Ball
Bo Tang
Jenny Du
Jenny Du (Graduate Coordinator)
Jason M. Keith (Dean, Bagley College of Engineering)

A Dissertation
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Doctorate of Philosophy
in Electrical and Computer Engineering
in the Department of Electrical and Computer Engineering

Mississippi State, Mississippi

December 2021

Copyright by

Himangi Srivastava

2021

Name: Himangi Srivastava

Date of Degree: December 10, 2021

Institution: Mississippi State University

Major Field: Electrical and Computer Engineering

Major Professor: Mehmet Kurum

Title of Study: Methods for inference and analysis of gene networks from RNA sequencing data

Pages of Study: 126

Candidate for Degree of Doctorate of Philosophy

RNA (Ribonuceic Acid) sequencing technology is a powerful technology used to give researchers essential information about the functionality of genes. The transcriptomic study and downstream analysis highlight the functioning of the genes associated with a specific biological process/treatment. In practice, differentially expressed genes associated with a particular treatment or genotype are subjected to downstream analysis to find some critical set of genes. This critical set of genes/ genes pathways infers the effect of the treatment in a cell or tissue. This dissertation describes the multiple stages framework of finding these critical sets of genes using different analysis methodologies and inference algorithms.

RNA sequencing technology helps to find the differentially expressed genes associated with the treatments and genotypes. The preliminary step of RNA-seq analysis consists of extracting the mRNA(messenger RNA) followed by mRNA libraries' preparation and sequencing using the Illumina HiSeq 2000 platform. The later stage analysis starts with mapping the RNA sequencing data (obtained from the previous step) to the genome annotations and counting each annotated

gene's reads to produce the gene expression data. The second step involves using the statistical method such as linear model fit, clustering, and probabilistic graphical modeling to analyze genes and gene networks' role in treatment responses.

In this dissertation, an R software package is developed that compiles all the RNA sequencing steps and the downstream analysis using the R software and Linux environment.

Inference methodology based on loopy belief propagation is conducted on the gene networks to infer the differential expression of the gene in the further step. The loopy belief propagation algorithm uses a computational modeling framework that takes the gene expression data and the transcriptional Factor interacting with the genes. The inference method starts with constructing a gene-Transcriptional Factor network. The construction of the network uses an undirected probabilistic graphical modeling approach. Later the belief message is propagated across all the nodes of the graphs.

The analysis and inference methods explained in the dissertation were applied to the Arabidopsis plant with two different genotypes subjected to two different stress treatments. The results for the analysis and inference methods are reported in the dissertation.

Key words: Gene, RNA sequencing, networks, gene-Transcriptional Factor, Loopy belief Propagation

DEDICATION

Dedicated to my parents for inspiring me to pursue my dream

and

for sharing my joy and disappointments in this endeavor

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my major advisor, Dr. Mehmet Kurum, for his constant guidance, encouragement, and insightful comments towards my dissertation. Besides my advisor, I would like to express my sincere gratitude to Dr. George V. Popescu, my minor professor, for his continuous support of my Ph.D. work, guidance, motivation, and immense research knowledge.

I would extend my gratitude to the rest of my committee members: Dr. Jenny Du, Dr. John Ball, Dr. Bo Tang, for their motivation and insightful observations towards my research.

I would also like to express my gratitude to Dr. Sorina Popescu for helping me with some of the very insightful doubts during my research work. I thank my colleagues Drew Ferrel, Philip Berg, Gizem Dimlioglu for their constant help and discussions on various projects.

Also, I thank my cousin Dr. Saurabh Dayal and his wife, Dr. Nimisha Srivastava, both Mississippi State University alumni, for strengthening me in all the phases of Ph.D. research. Lastly, I would like to thank my family for encouraging and empowering me all my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS, ABBREVIATIONS, AND NOMENCLATURE

BAM Binary alignment mapping reads

BCV Biological Coefficient of variation

DEG Differentially Expressed genes

Flg22 Bacterial Flagellin

DREM Dynamic Regulatory Events Miner

DGECM Differential Gene Expression Sample Comparison Matrix

DGE Differential Gene Expression

DM Design Matrix

GO Gene Ontology

Col-0 Control treatment at 0 hrs

ILK1 Integrin Linked Kinases

ILK1_CT_0_vs_WT_CT_0 A variable representing gene expression comparison between ILK1 under control treatment at 0hrs and WT under control treatment at 0hrs for NaCl/Flg22.

ILK1_NaCl_3_vs_WT_NaCl_3 A variable representing gene expression comparison between ILK1 under NaCl treatment at 3hrs and WT under NaCl treatment at 3hrs.

ILK1_NaCl_6_vs_WT_NaCl_6 A variable representing gene expression comparison between ILK1 under NaCl treatment at 6hrs and WT under NaCl treatment at 6hrs.

ILK1_NaCl_12_vs_WT_NaCl_12 A variable representing gene expression comparison between ILK1 under NaCl treatment at 12hrs and WT under NaCl treatment at 12hrs.

ILK1_flg22_3_vs_WT_flg22_3 A variable representing gene expression comparison between ILK1 under Flg22 treatment at 3hrs and WT under Flg22 treatment at 3hrs.

ILK1_flg22_6_vs_WT_flg22_6 A variable representing gene expression comparison between ILK1 under Flg22 treatment at 6hrs and WT under Flg22 treatment at 6hrs.

ILK1_flg22_12_vs_WT_flg22_12 A variable representing gene expression comparison between ILK1 under Flg22 treatment at 12hrs and WT under Flg22 treatment at 12hrs.

ILK1_Flg22_0_vs_ILK1_CT_0 A variable representing gene expression comparison between ILK1 type under Flg22 treatment at 3hrs and ILK1 under control treatment at 0hrs.

ILK1_Flg22_6_vs_ILK1_CT_0 A variable representing gene expression comparison between ILK1 type under Flg22 treatment at 6hrs and ILK1 under control treatment at 0hrs.

ILK1_Flg22_12_vs_ILK1_CT_0 A variable representing gene expression comparison between ILK1 type under Flg22 treatment at 12hrs and ILK1 under control treatment at 0hrs.

ILK1_NaCl_3_vs_ILK1_CT_0 A variable representing gene expression comparison between ILK1 type under NaCl treatment at 3hrs and ILK1 under control treatment at 0hrs.

ILK1_NaCl_6_vs_ILK1_CT_0 A variable representing gene expression comparison between ILK1 type under NaCl treatment at 6hrs and ILK1 under control treatment at 0hrs.

ILK1_NaCl_12_vs_ILK1_CT_0 A variable representing gene expression comparison between ILK1 type under NaCl treatment at 12hrs and ILK1 under control treatment at 0hrs.

mRNA Messenger RNA

MDS Multi Dimensional Scaling

MD Mean difference

NaCl Sodium Chloride

NGS Next Gene Sequencing

PGM Probabilistic graphical modelling

RNA seq RNA sequencing

SAM Sequencing Alignment mapping read

STAR Spliced Transcripts Alignment to a Reference

TF Transcription Factor

WT Wild Type

WGCNA Weighted Gene Correlation Network Analysis

WT_NaCl_3_vs_WT_CT_0 A variable representing gene expression comparison between WT type under NaCl treatment at 3hrs and WT under control treatment at 0hrs.

WT_NaCl_6_vs_WT_CT_0 A variable representing gene expression comparison between WT type under NaCl treatment at 6hrs and WT under control treatment at 0hrs.

WT_NaCl_12_vs_WT_CT_0 A variable representing gene expression comparison between WT type under NaCl treatment at 12 hrs and WT under control treatment at 0hrs.

WT_Flg22_3_vs_WT_CT_0 A variable representing gene expression comparison between WT type under Flg22 treatment at 3hrs and WT under control treatment at 0hrs.

WT_Flg22_6_vs_WT_CT_0 A variable representing gene expression comparison between WT type under Flg22 treatment at 6hrs and WT under control treatment at 0hrs.

WT_Flg22_12_vs_WT_CT_0 A variable representing gene expression comparison between WT type under Flg22 treatment at 12hrs and WT under control treatment at 0hrs.

CHAPTER I

INTRODUCTION

1.1 Motivation

Pathogen (biotic) stress and chemical (abiotic) stress negatively affect crop productivity. Therefore, there is a need to study the signaling landscape of plants using different technologies. Several computational models based on experimental observations are used to understand the signaling landscape of plants. Computational models give an insight into signaling networks and the properties that signaling networks possess [9].

The research explained in the dissertation presents some of the methodologies used for the analysis and inference in studying the signaling networks of plants. The research also further helps identify some significant genes and pathways in the plants. These genes and pathways are essential to extract because they help examine the responses of plants to various environmental stresses that further affect crop productivity. In this dissertation, the effect on plants is studied because of pathogen treatment and salinity treatment. The chemical reaction caused due to pathogen treatment has an adverse effect on the plant. The salinity is studied to investigate the effect of overirrigation and saline water irrigation on plant biological processes.

This analysis framework consists of a computational model that included Ribonucleic acid (RNA) sequencing technology, network analysis, and probabilistic graphical modeling technique to perform data mining, data analysis, and modeling the inference. This dissertation presents

a network model approach for computational analysis to infer essential functions of the plant's signaling pathway.

The motivation of the dissertation is to study the gene expression data and characterize the signaling network associated with biotic and abiotic stresses in Arabidopsis thaliana. figure 1.1 on page 15 shows an image of Arabidopsis Thaliana. Gene expression data for Arabidopsis thaliana associated with bacterial flagellin (biotic) and NaCl (abiotic stresses) was mined using different bioinformatics tools.

The second fundamental aim of this dissertation is to design an advanced bioinformatics computational pipeline using RNA sequencing technology and machine learning algorithms. This pipeline could be used to infer useful information about the biological functionality of genes expressed in our data. The methodology explained in the dissertation can help study the signaling landscape in other plants and could be used to identify crucial genes and pathways.

1.2 Background and Literature Review

"Signaling network in a cellular organism is a function of molecules interacting and stimulating surrounding proteins, lipids, and ions, resulting in cytoskeletal reorganization, modulation of differentiation, and induction of gene expression" [65]. Previous research has gained insight into several biochemical processes inferred by studying the Arabidopsis thaliana's signaling network. The short life cycle, small plant size, and efficient reproduction through self-pollination are reasons to use Arabidopsis for genetic analysis [38].

In the research [17], the authors have used an information theory algorithm to investigate the activation of the gene as an interference method to study signal transduction in the Arabidopsis

roots. The research performed a similar analysis on Arabidopsis Integrin Linked Kinases (ILK-1) mutant to gain further insight into its effect. There have been several techniques used in previous work to identify the function of signaling networks in plants. This section reviews most of the techniques used in the network model approach to infer the gene signaling network. In one of the previous work [53], the transcriptomic response of ILK-1 type mutant of Arabidopsis mutant has been studied to learn the signaling response that is mediated by ILK1-1 type mutant in Arabidopsis.

The research carried out in this dissertation investigated the signaling network response on a time-series transcriptome data of Arabidopsis thaliana wild-type (WT) and ILK-1 mutant. The idea is to study the effects of treatments within the plant having the normal phenotype(Wild Type) and on special ILK1 phenotype. The gene expression data used in the research was obtained using RNA sequencing technology. Only recently, with the advent and increase of Next Gene Sequencing (NGS) technology, the potential of using RNA sequencing technology for various other research goals has increased [62]. The NGS is a collection of advanced technologies used for the mass profiling of Deoxyribonucleic acid (DNA) and RNA sequences [31]. This technology includes various software platforms utilized on different operating systems (mainly Linux) for quality check, processing, and analysis of sequenced reads. Furthermore, RNA sequencing technology is also used for many different aspects of RNA biology, including single-cell gene expression analysis [63]. The overall methodology and steps required for RNA sequencing technology is presented in literature [14]. A significant requirement in RNA sequencing technology is good experimental design. The experimental design involves choosing the library type, sequencing depth, and replicating appropriately for the sample under the study. Then it requires setting up a sufficient execution of the sequencing experiment [14]. The first step after experimental setup is checking the quality of

reads using software FastQC [7]. This software gives the quality of raw reads. Reads quality are accessed using Phred score. Phred Score is estimated score which is related to the log of quality of errors. Reads with good quality have a Phred score between 20 and 30 [56]. In the research provided in this dissertation, the reads whose Phred score ≥ 30 were used for further analysis.

The next step to RNA sequencing technology is the mapping of the reads. Several read alignment tools are used in the RNA sequencing step to find quantified counts for genes across different samples. Some of them are STAR (Spliced Transcripts Alignment to a Reference), HISAT (hierarchical indexing for spliced alignment of transcripts), and Kallisto (a program for quantifying abundances of transcripts from bulk and single-cell RNA-Seq data). To get deeper insight into the alignment software, comparison of different tools used for RNA sequencing were studied in the previous work [55]. The article [51] discusses the use of software such as HISAT, StringTie (a fast and highly efficient assembler of RNA-Seq alignments into potential transcripts), and Ballgown (a software package designed to facilitate flexible differential expression analysis of RNA-seq data).To study the transcript level analysis on gene expression data, the author in article [68] discussed TopHat (a spliced read mapper for RNA-Seq) [35] for the differential gene, and transcript expression analysis of RNA-seq data. HISAT is a fast spliced aligner with low memory requirements, making it efficient for analysis of RNA sequencing data [34].

In chapter 4, two types of alignment methods: transcript-level analysis and genotypic level analysis, have been presented. In the subsequent chapter, genotypic expression analysis was carried out using the STAR aligner [22]. In the article [21], the functionality of the STAR aligner, and how to set parameters to run STAR aligner on RNA seq data has been discussed. STAR works on identifying the novel splice junction and aligning reads to the transcriptome. Thus it is

dependent on the gene annotations for the placement of the aligned reads. The reason to use STAR aligner is that it is time-saving and very sensitive to aligning reads to a reference genome than the other alignment programs. The article [21] also provides a comparative analysis of STAR's performance with all the other aligners.

The next step in RNA sequencing technology is to quantify the gene expression data. This step is performed using software that can count the number of reads mapped(aligned) to each gene. Several software tools carry out this task; some of them are HTSeq-count [6], featureCounts [43], Cufflinks [69], etc. These read quantification tools help the user produce a count matrix. This matrix is used for further downstream analysis to discover the differentially expressed genes. The next step is to find the differentially expressed gene out of the count's matrix obtained, using the gene quantification step.

The reason to identify differentially expressed genes is to find and observe the genes that show statistically significant changes in expression level between two experimental conditions [8]. The article [76] highlights a comparative analysis of different analysis tools such as Cuffdiff 2 (finds significant changes in transcript expression, splicing, and promoter use.) [67]. To analyse the counts data, DEseq (an R package to analyse count data from high-throughput sequencing assays) [5] and EdgeR (a Bioconductor package for differential expression analysis of digital gene expression data) have also been discussed [44] [58] that could be used to find differentially expressed genes (DEGs). All three of the analysis tools are based on modeling the count's data. These analysis tools use negative binomial distribution and filter out the lowly expressed gene using the relative log expression method. The relative log expression method is used to scale the counts data into the log scale. Cuffdiff 2 [67] is part of the cufflink tool; it robustly identifies differentially

expressed transcripts and genes and reveals differential splicing and promoter-preference changes [67]. DEseq and EdgeR are the Bioconductor [25] packages developed on R and could be loaded on the R using the Bioconductor. DEseq and EdgeR [58] model the gene expression data based on the negative binomial distribution, with variance and mean linked by local regression [5]. In this research R software package EdgeR [44] [58] was used to carry out the differential gene expression analysis. The discussion on the differential gene expression analysis using this method is explained in the later chapters. This dissertation also discusses the development of R software package NetSeekR using the above mentioned tools in the later chapters.

The next step was to study the interaction pattern found in DEGs at different obtained time points. The interaction pattern analysis was carried out by constructing the gene correlation network and identifying gene modules/ clusters in the DEGs. The relationship between genes is better described by the scale-free networks [54], so the Weighted Gene Correlation Network Analysis (WGCNA) [41] software package was used. The idea was to identify key genes related to specific biological processes using the Gene ontology process later. WGCNA has already been used in several pieces of research, in one of the researches where it is applied to Proteomic and Metabolomic Data Analysis [50]. WGCNA is also used as a tool to analyze metabolomic profiling for tomato plant [18]. WGCNA could measure the robust association between the genes, and it also accounts for the indirect association. WGCNA could also be used for other plant's gene expression data as well. Apart from using other popular correlation methodology used for calculating correlation, WGCNA uses biweight mid correlation [70] to calculate a robust correlation between DEGs.

The hierarchical clustering method in WGCNA is deterministic, as each gene will be assigned to only one cluster [74]. It consists of a set of function which works iteratively to assign a gene to a cluster. Topological analysis on gene correlation network is conducted using several packages available on R and Python. These packages are also used to create and analyze the co-expressed network from gene expression data; some of them are igraph [15],tidygraph [49]. In addition, these packages have been used to analyze various community structures in social and biological networks [26].

The following process identified transcription factors (TF)-gene interaction for further genomic analysis to infer valid biological processes related to the genes. Transcription factor gene regulatory network is networks obtained from the technique, which involves finding the TF that activates the genes based on the number of the binding site for each TF on the gene. The research in the dissertation uses the database, International System for Agricultural Science and Technology (AGRIS), and software tool Dynamic Regulatory Events Miner (DREM) [27] was used to obtain the TF gene regulatory network.

After finding DEGs from differential gene expression analysis, clusters of DEGs from correlation analysis, and some crucial genes from network analysis, the final task was to find the functional annotation of DEGs and clusters of DEGs. This step is carried out using the gene ontology analysis step. Gene ontology helps find out if the selected gene is associated with specific biological processes or molecular functions. There are several public database-related Gene ontology analyses databases and tools such as Panther (protein analysis through evolutionary relationships) DB (database) [45], DAVID (the database for annotation, visualization and integrated discovery) [29], GSEA (Gene Set Enrichment Analysis) [64] are used; some of them are based on R packages like

TopGO (Enrichment Analysis for Gene Ontology) [4]. The practical implications and benefits of NetseekR are that it offers an easy framework for downstream data analysis of plants gene expression data with a varied number of bioinformatics tools used which could be used for a comparative study of gene expression data.

Probabilistic graphical modeling is the most famous technique used today in the computational analysis of biological data. Probabilistic graphical models are one technique where both biological and mathematical modeling combine into a graphical architecture with a standard, intuitive formalism [2]. The authors in the work [47], reviews the basics of different probabilistic graphical models for network modeling. In this dissertation, a gene-TF regulatory network was converted into a probabilistic graphical model to infer the differential expression of TF. There have been several articles that have used a similar approach. In the research explained in the work [24], different methods of inferring Cellular Networks using probabilistic graphical models have been explained. In the article [1], the probabilistic modeling using Bayesian network models is used in predicting types of hematological malignancies. This algorithm has earlier been used to solve image denoising and image segmentation in medical imaging. There are various optimization techniques used for energy optimization in probabilistic graphical modeling. One of them is loopy belief propagation. Loopy belief propagation algorithm uses the methodology of passing belief messages in the network which has loops to find the marginal probability of each node.The authors in the research [3] discuss the use of the Loopy belief Propagation algorithm on a directed graph. In the article [46], the authors explain application of the belief propagation algorithm to both the directed and undirected graph. The application of belief propagation and its different derivatives have been discussed in the article [73]. In the research [36] mean-field method and loopy belief

propagation is applied to the pairwise Markov networking model to accomplish the task of image processing.

1.3 Contributions

In chapter 2, data mining of differentially expressed genes at different time points and treatments is discussed. The data generated from the data mining step was then further used for downstream analysis (explained in Chapter 3) using correlation analysis, clustering, and network topology analysis to find the critical set of genes responsible for a particular biological process flg22(Bacterial Flagellin) treatment. The RNA seq data was divided for two different treatments and was checked for low expression and data artifacts of the genes across all the samples. This data preprocessing was performed using the Log-CPM filtering technique and TMM(Trimmed Mean value) based and Voom-based normalization using EdgeR and Limma packages. The Log-CPM filtering actually converts the numeric counts of each reads into the form of log of counts per million. Later the genes with the value lower than a set threshold of Log-CPM is filtered out. TMM is based on normalizing Log-CPM counts with respect to mean of a reference sample and Voom is a variance based normalization technique. The differentially expressed were investigated for many different comparison analyses. The comparison analysis involved genotypic and time series analysis across all the samples and NaCl and Flg22 treatments. The differentially expressed genes obtained across different time points and different genotypes were further investigated for their biological function using the second step in the downstream analysis mentioned in Chapter 3.

Chapter 3 follows the downstream analysis of the DEGs obtained from chapter 2. This chapter discussed constructing a gene correlation network from DEGs and identifying gene modules/ clus-

ters in the DEGs obtained from chapter 2. This analysis methodology helps us to find the essential genes related to specific biological processes.The analysis methodology reported in chapter 3 was used in reporting the effect of ILK1 expression (activated by the bacterial elicitor flg22) on the cell wall integrity and immunity. The majority of 30 DEGs linked to cell wall biosynthesis and modification, and the 20 DEGs encoding anchored membrane proteins, 18 immune response DEGs, and 13 microtubule-associated DEGs had defective regulation in the ilk1-1 line.

Chapter 4 explains developing a novel data analysis pipeline for mining and analyzing different types of gene expression data. The network analysis R package developed in Chapter 4 includes the capacity to analyze the time series RNASeq data, perform correlation and regulatory network inferences, and use network analysis methods to summarize the results of a comparative genomics study. The pipeline compiles all the analysis methodology explained in chapter 2 and chapter 3 and other methodologies and bioinformatics tools as an R software package. This software package was called NetseekR. NetSeekR can be executed using arguments from an argument .tsv file, or a .tsv file containing a set of argument files and could also be executed individually by entering the parameters circumventing argument passing from argument files entirely.

Chapter 5 explains an inference algorithm for the TF-gene regulatory network using probabilistic graphical modeling to find differential expression of genes for flg22 treatment. Most of the software tools available for bioinformatics data analysis to find the differential expression of genes assume that the gene expression data has a negative binomial distribution. The algorithm proposed in this chapter, on the other hand, uses a probabilistic approach of interaction between genes to find the differential expression of genes. The inference method used an undirected graphical modeling technique to construct a factor graph. After modeling the factor graph, an inference algorithm

based on a belief propagation algorithm was used to find the differential expression of gene and TF. This algorithm could be used to find the expression of a set of the genes that act as TF without considering the entire gene expression data distribution.

The different methodologies of analysis and inference explained in the thesis could be used for performing analysis and inference on several other time series RNA sequencing data.

1.4 Overview

This dissertation focuses on performing analysis and inference of gene networks using gene expression time-series data. It gives a detailed overview and methodology of various tools used for RNA sequencing technology to obtain the data and further analyze it. The first two chapters describe the steps to analyze gene expression data using different bioinformatics software tools. The third chapter of the dissertation contains a description of the implementation of a network bioinformatics software pipeline. Finally, the fourth chapter contains several machine learning algorithms used to draw gene network inference on the gene expression data analyzed in the first chapters. The framework of the dissertation is given in the figure 1.2 on page 15.

The analysis of time series RNASeq data starts with the data mining of gene expression data. Gene expression data were obtained using the RNA sequencing technology following an experimental design performed in a collaborating plant biochemistry laboratory. There are multiple workflows required in identifying and analyzing the gene (genomic loci) expression data. The first workflow is an experimental method called sequencing of the genes, which starts from the extraction of RNA followed by library preparation of the samples and finally producing the sequence reads. The second workflow uses bioinformatics methods implemented in R to find differentially

expressed genes. This workflow starts with the processing of reads sequence (quality check, alignment, trimming, etc.), mapping reads to coding regions, followed by summarizing individual expression of genes on each sample, normalizing the gene expression data, and finally identifying the differentially expressed gene. After obtaining the DEG genes, the next task is to find the gene expression correlations and perform network analysis of the DEG gene expression data. This analysis employs network inference modeling, systems dynamics, stochastic modeling, and simulation methods for time-series observations of gene expression dynamics. Finally, we draw inference on gene networks using probabilistic graphical modeling and loopy belief propagation message passing algorithm.

The experimental data we used in our work comes from an Arabidopsis thaliana study to identify factors that modulate its responses to various biotic and abiotic stresses. The RNASeq data was sequenced using the Illumina HiSeq 2000 platform to produce an average of 15 M pair-end reads per sample for each of the 96 samples in our dataset. These samples were collected for two genotypes (WT or Col0 and ILK1-silenced mutant) on which two different stress treatments (NaCl and Flg22) were carried out at four different time points (0 hrs., 3 hrs., 6 hrs. and 12 hrs.) in three replicates per time point.

The work was divided into four stages to accomplish the goal of my research. Several bioinformatics tools and software packages were used to perform the data mining, analysis, and inference on the data. Each stage of my work is briefly described briefly here. The rest of the chapters give a detailed description of the entire process step by step.

The first objective of this study is to identify the DEG genes obtained from the gene expression data and find out their biological significance. In this stage of the work, mapping the reads from

RNA seq data was done using the software STAR aligner [21]. The mapping of reads and then processed the alignment using the software Samtools and finally software FeatureCounts was used to find the mRNA counts for each gene obtained previously by mapping short reverse-transcribed RNA fragments to a reference genome. Finally, R packages called Limma [57] and EdgeR [44] [58] were used to perform the differential analysis on the RNA- seq data to obtain and identify DEG associated with stress treatments.

In the second stage of work, correlation patterns amongst the genes were analyzed by constructing gene co-expression networks using the R software package WGCNA [41]. Gene prediction tool, DREM [61] was also used to identify transcription factors that control gene expression dynamics in time series data. Finally, this network analysis was integrated with the multiple bioinformatics tools and studied the resulting gene network's topology. After obtaining the correlation network of the DEGs, the correlation networks were integrated with existing public gene networks and further inferred and analyzed the dynamics of regulatory and signaling of A. thaliana gene networks in our study.

The last part of the research is dedicated to identifying and inferring gene networks using belief propagation algorithms. The algorithms for gene network inference uses input a gene expression dataset (as described above) and can have many applications. Here undirected graphical modeling was used to implement the method to draw inference on gene networks. Markov random field and belief propagation algorithm were implemented to perform gene network inference using gene expression time-series data as observations and TF- gene regulation data for the network information. The concept of a belief message-passing algorithm called the sum-product algorithm was used to find the marginal probabilities of individual genes in the network to finally obtained

the normalized belief messages for all the genes in the network. Based on the final belief messages the differential expression for the genes was set.

All the methodologies for analyzing the time series data and drawing inference from the time series RNA seq data are explained in the subsequent chapters.

The analysis methodologies and the network analysis pipeline developed in the dissertation can be used for mining and analyzing any kind of genomics data. The work in the dissertation discusses the data mining and analysis performed on Arabidopsis. From the data mining and analysis of DEGs (Differentially Expressed Genes), several genes responsible for several biological processes were obtained. The analysis methodologies and tools used in the dissertation could also be used for other plant's gene expression data in order to retrieve useful information about the biological processes related to a kind of stress treatment. The probabilistic graphical modeling gives a different approach to carrying out the analysis than the one primarily used. This methodology of inferring useful information from gene expression data could also be used for any kind of genetic analysis involving RNA sequencing data.

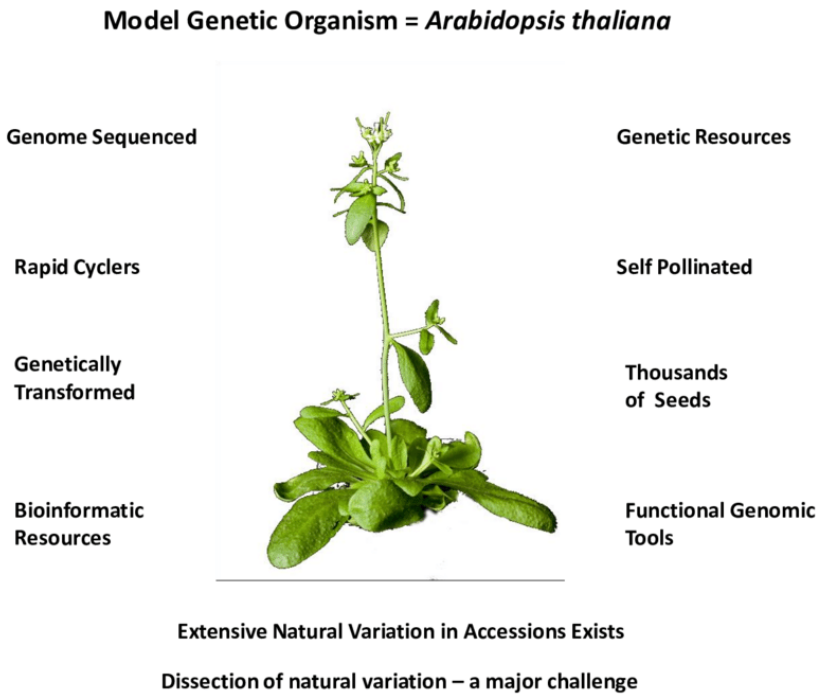

Figure 1.1

Arabidopsis Thaliana from [33]

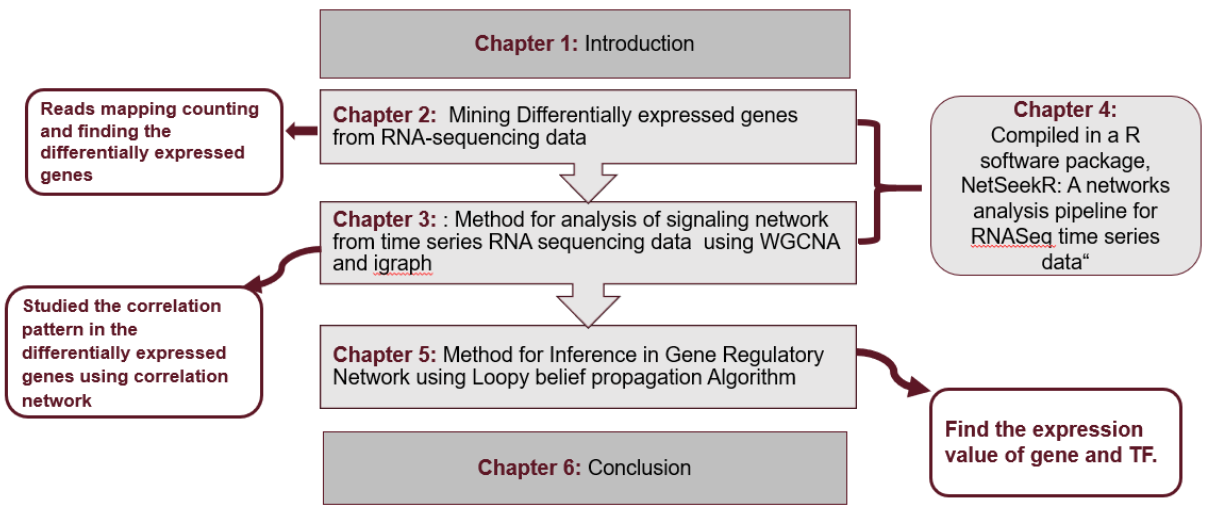

Figure 1.2

Dissertation Framework

CHAPTER II

MINING DIFFERENTIALLY EXPRESSED GENES USING RNA SEQUENCING TECHNOLOGY

2.1 Introduction

This chapter studies the global transcriptional responses of *A. thaliana* mediated by the biotic and abiotic stresses using NGS technology. The idea is to study the transcriptomic data which helps measure the mRNA transcripts expressed by the genome of the cell. In this research, Arabidopsis plants' transcriptomic response of ILK1-silenced and wild-type mutant was analyzed.

Sequenced mRNA were obtained from an experimental design performed in the controlled environment. The experimental design was carried out to study two stress conditions across four different time points. The data obtained from this previous experimental study contains an average of 15 M pair-end reads per sample, for 48 different samples sequenced using the Illumina HiSeq 2000 platform. The samples consist of two types of mutants (Wild Type and ILK1) on which two different stress treatment (NaCl and flg22) treatments were carried out at four different time points (0 hrs., 3 hrs., 6 hrs. and 12 hrs.). The two stress treatment experiments used in the study were: a) abiotic stress-induced through NaCl and b) biotic stress-induced through the bacterial pathogen associates molecular pattern flg22, a bacterial flagellin peptide.The significance of the experimental design was to study the effect of pathogen and salinity on the plant at different time points.

This chapter preprocesses the reads generated for the Arabidopsis plant using NGS technology to find the counts of the genes present in each sample. It then analyses the data under different stress conditions mentioned earlier across different time points to analyze the plant's changes and their impact on the different kinds of arabidopsis plant mutants at different time frames.

The analysis goal in this study is to identify and compare the differentially expressed genes at different time points when two different genotypes (WT and ILK1-silenced) are subjected to different stress conditions (biotic and abiotic stress). This chapter presents a computational workflow to detect the differentially expressed genes from RNA-sequencing data obtained using Illumina HiSeq 2000 sequencer.

2.2 Methodology

This section gives a detailed overview of the pipeline that followed for the analysis.

2.2.1 Alignment of Reads

Starting with the raw reads, the first task is to map millions of sequencing reads to the corresponding A. thaliana gene loci; counts of mapped reads will then be used to estimate transcript abundance present in the specific sample. The challenge involved in this task is to identify the correct spliced junction as there is the possibility of the error involving the spliced alignment of exon-exon spanning reads. Splice-aware aligners can estimate the abundance of multiple different transcripts of the same gene. STAR aligner software was used to carry out the reads mapping task. This alignment software has a high sensitivity , very good precision on the reads that were mapped to the reference genome. Another advantage of STAR aligner is it is faster than other alignment programs; its balanced performance represents an advantage over other alignment software. STAR

The STAR alignment process consists of choosing the appropriate reference genome to map our reads. STAR precision for placement of the aligned reads is dependent on the quality of gene annotations. Here Araport 11 was used, which represents a significant improvement on A thaliana annotations [13].

The input in the STAR aligner consists of the reference genome and the genome annotation file. Initially, Arabidopsis genome and the annotation files were downloaded from Phytozome [40] to carry out the task of alignment. The resultant trimmed FASTAQ files were then mapped to the reference genome using STAR. The individual samples of the experimental data were then mapped to the Arabidopsis genome. To perform STAR alignment, first genome indices were generated using the gene annotation files [21]. The resultant genome index file comprises the genome sequence, suffix array, information about the genes, e.g., the strand, chromosome names, and length and splice junction coordinates. All these resultant files obtained after generating the genome indices act as an input to the next step. The next step in STAR is to align reads from FASTAQ files after genome indices files are generated. The results of our read alignment using STAR are shown later in the chapter. The next task after the alignment is the processing of the aligned reads obtained from STAR.

2.2.2 Processing Aligned Reads

After mapping the FASTAQ file to the reference genome, STAR stores the alignment file's result in a SAM (Sequencing Alignment mapping read) or BAM (Binary alignment mapping reads) format. SAM format explains the sequencing's alignment in the form of a nucleotide alignment format. The SAM format needs to be converted to BAM format (Binary alignment/Map format). To

accomplish this task, another software package called Samtools [42] was used. Samtools software package allows us to read the SAM files and perform filtering on them. Samtools converts SAM files to BAM files and then performs sorting on them. It also tries to remove duplicates, improper and multi-paired reads. The SAM files were converted into BAM file format and performed the sorting of reads using Samtools.

Next, the read summarizing program, FeatureCounts, was used, which counts the reads from the RNA sequencing experiment completed in the previous task. Feature Counts is part of the Subread software package [43]. Using this software, the number of reads mapped to each gene for each sample were counted.

2.2.3 Differential Gene Expression Analysis

In our next step, EdgeR was used to analyze the read counts obtained from the FeatureCounts. The edgeR software package is an R software package used for differential expression analysis of RNA sequencing data [58]. This software implements various statistical methodologies for carrying out operations on our gene counts data, such as filtering, normalization, multidimensional plotting, clustering, etc. This R's software packages use the negative binomial distribution for data modeling, generalized linear models, and empirical Bayes estimation for differential gene expression analysis. The gene expression data requires negative binomial distribution as the mean and variance in the negative binomial distribution don't vary too much as in Poisson's distribution. Some of the functions from the Limma package [57] were also used along with edgeR to carry out the analysis.

The data was into two different batches. The first batch of data contained NaCl samples treated and three wild-types (WT) control samples at 0 hrs. and three samples of ILK1 control at 0 hrs. The second batch of data contained samples that were Flg22 treated along with three samples of Wild type (WT) control at 0 hrs. and three samples of ILK1 control at 0 hrs. We analyzed these datasets separately. Since the data analysis was carried out in two batches NaCl and Flg22, different groups in each batch were also based on the type, treatment, and number of hours.

In order to carry out the analysis, there was need to design the analysis in different groups using design matrix. The design matrix records treatment conditions were applied to each sample, and it also defines how the experimental effects are parametrized in the linear models. The design matrix was formed using the following given groups in table 2.1 on page 32 and table 2.2 on page 33 below. table 2.1 on page 32 represents the groups for batch 1, while table 2.2 on page 33 represents the group for batch 2. In the tables mentioned above, the groups' column specifies the groups the samples were grouped.

The differential gene expression analysis was carried out in the following steps:

1. Filtering of the data: In the data, there were approximately 24532 genes in batch 1 and batch 2, after finding the logCPM and CPM of all our counts, the gene that had less than 0.5 counts per million of the gene counts were filtered out. The reason these genes were filtered out was that these genes acted as the outlier for further analysis. Further, these genes lowly expressed genes do not have any significance while finding differentially expressed genes. After filtering the lowly expressed gene, 20854 genes were left in batch 1, and 19234 genes were left in batch 2. These lowly expressed genes do not help provide any statistical evidence that could be used to make a relevant judgment [12].

2. Normalization of the dataset: An essential task for our data's downstream analysis is to remove any data artifacts or batch effects in our dataset. Since the counts data varies across different samples, so there was a need to normalize the data to a standard scale in order to fit the linear model on the gene expression data. The data normalization in gene expression data is done using scaling the library size of each sample of the gene expression data on a single numerical scale. Normalization was performed using the calc_normfactors function in Edge

R. Trimming the M-value (TMM) method was used to normalize the data. In addition, the article [59] describes various other methods that could perform normalization on the data.

3. Forming the DGE List object of the remaining counts: Next, we create a DGElist object to hold the read counts remaining after the filtering. The DGElist is a list used in the EdgeR software package used for gene expression data manipulation.

   Since the data analysis is carried out in two batches NaCl and Flg22, different groups are present in every batch based on the type, treatment, and number of hours. The DGElist contains the read counts and the associated metadata, including sample names, gene names, and normalization factors once they are computed.

4. Exploratory data analysis: The data is studied using unsupervised clustering of gene expression to study the samples' relationship. Multidimensional scaling plots were also used to perform the analysis of the inter-sample relationships.

5. Setting up the Design matrix and voom normalization of the dataset: Next, the data model is set up by taking the DGElist object and the groups and formed the design matrix. The data is normalized again using the voom normalization technique. Voom is used on log-transformed data because it uses mean-variance trend derived weighths from normalized logCPM value for the dataset. It also removes some of the lowly expressed genes that were not filtered in the previous filtering step. Voom plots were also used to analyze the mean-variance trend of biological replicates before and after normalization.

6. Fitting the model: The next step is to fit the model. This step is carried out using the Limma function lmfit and contrast.fit for contrast matrix. Empirical Bayes moderation is applied to some of our contrast matrix, which is later implied across all the genes in the contrast matrix.

7. Examining the Differential expression genes: Upregulated and Down-regulated genes obtained after fitting the model were summarised in the form of a table. The adjusted *p-value* is set at 5% by default. However, a stricter significance was needed to define the proper analysis and decisions performed in the next step.

8. Setting the threshold log-FC for stricter decisions: In this research, the log-FC (Log-Fold-change) was needed to be above 1. The treat method was used by using the Treat function provided by the Limma package to carry out this step. The treat method calculates the *p-value* from the empirical Bayes moderator t- statistics with the desired threshold log-FC value requirement.

9. Differentially expressed data: Upregulated (+1) and downregulated genes (-1) were found in each sample and were stored in a CSV file for each comparison. Mean difference plot was also plotted to study the upregulated and downregulated genes.

2.3 Results and Discussions

The results of each of the pipelines, as mentioned above, and procedures are discussed in this section. The first step in RNA sequencing was obtained from the Illumina HiSeq 2000 platform, which generated 394 and 492 million reads for NaCl and flg22 treatments, respectively. These reads are 50 base pair long, pair-ended reads. These reads were obtained from three different biological replicates of Wild Type-Control, and ILK1 knockouts plants for both stress responses (NaCl and Flg22) carried out at four different time points, which resulted in an average of 15 M pair end reads per sample for each of the 48 samples in our dataset. The 48 samples' sequencing resulted in 96 FASTQ files because the sequencing was done using paired-end sequencing. These reads were subjected to sequencing data quality checks by running those using FastQC. The base pairs below the quality score of 28 were cut. Finally, the remaining low-quality reads were trimmed using Trimmomatic to remove the adaptors and low-quality reads, increasing the quality score ($\geq$30). The last reads obtained were of about 36 base pairs. The resulting high-quality, pair-ended data we obtained after quality check and trimming for each sample was about 15 million.

2.3.1 Alignment and mapping reads

The resultant trimmed FASTAQ files containing reads were then mapped against the Araport11 Arabidopsis reference genome using STAR. The samples were mapped to the Arabidopsis genome. After generating indices, we then performed read alignment using STAR. The results obtained using STAR aligner are below.

table 2.3 on page 34 and table 2.4 on page 35 represent the reads alignment results for NaCl and Flg22 treated samples respectively. The first column in both the tables represents the library

id and the total number of initial reads found in that particular sample. The next column represents the reads that were uniquely mapped to the reference Arabidopsis genome.

Looking at the data,one can say there was on an average of 95% uniquely mapping of reads in our data was obtained. This signifies that the reads were mapped to the reference genome effectively.

After mapping the FASTAQ file to the reference genome, SAM alignment file was obtained as the output. the file that were in SAM format was converted to BAM format to perform the next sorting task, and then sorting of the BAM File was done to perform the downstream analysis in the latter task. After processing reads, read summarization program FeatureCounts was used. Counts of reads mapped to each gene for every 48 samples from the RNA sequencing experiment were obtained. The output obtained from feature counts consisted of two files: in the form of the text file containing the actual read counts per gene along with gene ID, genomic coordinates of the gene's information, including strand and length and a summary file that gives an overview of the genes that could be assigned to the gene and some of the reason some of them could not be assigned to the gene. Some data cleaning needed to be performed before we can use the data for further analysis. This data cleaning was done using FeatureCounts "cut" operation.Cleaned text files were obtained that could be directly loaded into R software as an R data frame and carry out the genes' downstream analysis.

2.3.2 Differential expression analysis

Getting started with the analysis, counts text files that FeatureCounts generated was loaded into the R data frame and then combined into a matrix. The sample matrix for the analysis is shown below in table 2.5 on page 35.

table 2.5 on page 35 contains the example of the gene expression data for samples 21, 22, 23, 24, 25, 26,27 and 28 (last two digits of sample's id). The corresponding gene is represented as gene ids. Our actual data frame matrix consists of 24532 rows, which denote the gene names, and the columns indicate the sample library names by their respective numbers. The entries in each cell of the table contains the entries for gene counts. The entire matrix's sample-level information was collected to further acquire the data into a specific matrix for the following downstream analysis. The data according to the two batches were segregated. Batch 1 contained all the NaCl treatment for both the types (wild and the ILK1 type) collected at all different time points and three control type treatment at 0 hr. for Wild type and three control type treatment 0 hr. for ILK1 type. Similarly, batch 2 contained all the Flg22 treatment for both the types (wild and the ILK1 type) collected at all different time points and three control type treatment at 0 hr. for Wild type and three control type treatments 0 hr. for ILK1 type. Next, the essential libraries for the analysis were loaded.

2.3.2.1 Filtering and Normalization of the count data

In the data, there were approximately 24532 genes in batch 1 and batch 2; after finding the logCPM and CPM of all our counts, the gene that had less than 0.5 counts per million of the gene counts were filtered out. After Filtering the lowly expressed gene, we were left with 20854 genes in batch 1 and 19234 genes in batch 2. The DGElist object of this remaining data after the filtering

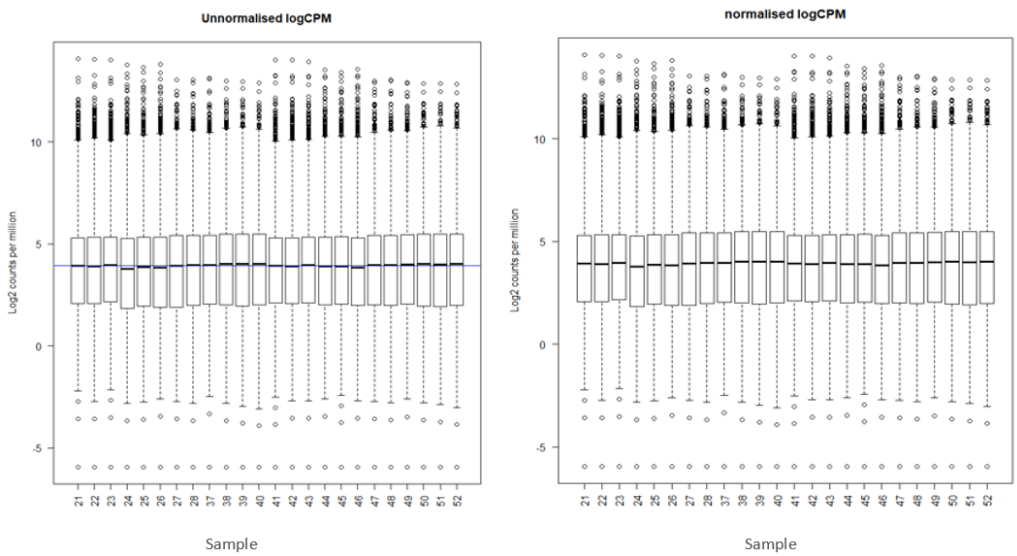

Figure 2.1

Library plot of batch 1 data before and after "TMM" normalization

was formed. Normalization was performed using *calc_normfactors* function in Edge R. We use normalization by trimmed mean of M-value (TMM) method to normalize the data. Normalization factors were checked for all the samples used in the analysis. Different normalization methods are discussed in this review article [19]. The effect of TMM normalization on the dataset was mild as the normalization factor for each sample was close to 1 for each sample. Figure 2.1 and figure 2.2 on the next page depict the change in the library size of the data before and after the normalization in batch 1 and batch 2 respectively. The x axis in the plots represents the logCPM value of counts of genes in each samples and the y axis represents the samples id.

2.3.2.2 Extrapolatory Data Analysis

Next, the exploratory data analysis was performed using the MDS plot. These plots are plotted using Limma plots. MDS plots were plotted using the biological coefficient of variance distances

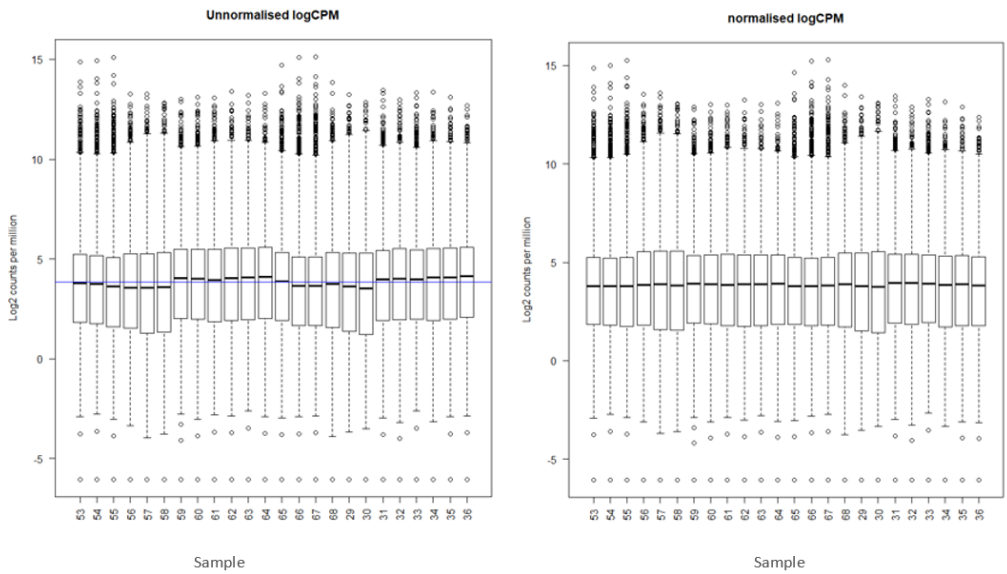

Figure 2.2

Library plot of batch 2 data before and after "TMM" normalization

method as our dimensions. This method calculates the distance between the samples based on their biological coefficient of variation.

Looking at the MDS plot in figure 2.3 on page 36 for batch 1 for all the samples, one can say that there is a larger BCV (biological coefficient of variation) transcriptional difference along with the axis BCV distance one between the samples consisting of both wild type and ILK1 type collected at zero hours in the control treatment and the samples consisting both wild type and ILK1 at different time points treated with NaCl.

The almost similar effect can also be seen for Batch 2 in figure 2.4 on page 37, where there is a larger BCV (biological coefficient of variation) transcriptional difference along the axis BCV distance 1 between the samples consisting of both wild type and ILK1 type collected at zero hours in control treatment and the samples consisting both wild type and ILK1 at different time points

treated with flg22. Looking at both the plots for both the batches to study bcv transcriptional difference along with axis BCV distance 2, one can find that the transcriptional difference is dependent on the hours of the particular type is treated as we can see in batch 1 MDS plot, the wild type and ILK1 type treated for 3 hours are clustered together separating the other cluster of wild type and ILK1 type treated for 6 hours and the cluster containing wild type and ILK1 type treated for 12 hours. A similar effect can again be seen in the batch 2 MDS plot. These clusters show the difference as they are separated by the distance along axis 2, which is BCV distance 2.

2.3.2.3 Estimating the dispersion

The analysis calculated the common and trend-wise dispersion of the data stored in the DGElist object. The idea to calculate the estimated dispersion first is to find the dispersion value and determine if the gene counts we are analyzing will fit the negative binomial model developed out of the gene counts data. figure 2.5 on page 38 below shows the estimated common dispersion and trend-wise dispersion plots for batch 1 and batch 2, respectively. The common dispersion is plotted with the red line, while the Trend wise Dispersion is plotted with the blue line.

2.3.2.4 Voom plot

Further normalization of the data was performed using Voom. figure 2.6 on page 39 below shows the Voom transformed log-CPM plot. Voom provides a significant normalization across all the samples in our dataset to ensure the uniform distribution of the dataset. This normalized data proves to be an essential step for the next downstream test and analysis to obtain differentially expressed genes.

Voom normalized counts of the genes were used in the subsequent downstream analysis of DEG genes in the next chapter for hierarchical clustering and correlation analysis. The idea is to study patterns in the DEG genes and find the pathway analysis and gene ontology analysis of selected genes.

2.3.2.5 Examining Differentially Expressed Genes

For a quick look at the data, the analysis's upregulated and downregulated genes were summarized in the form of a table. The number of upregulated genes were summarized as "Up" and the number downregulated genes were summarized as "Down". The number of genes that were not significant were summarized as "Not Sign" for different comparison sets. These results were obtained when the comparison analysis was carried on different comparison sets and set the required log-FC value threshold. Linear model firring was performed for the comparison set of interest using linear modeling in limma, which is carried out using *contrast.fit* function. Next, the empirical Bayes moderation is carried out. To examine the DE genes, treat function was used to set the Log FC value and then used *decide* function where we set the *p-value* using the *p-value* adjustment method parameter set as "BH." The resultant differentially expressed genes are extracted using the results from *decideTests*. The tables below list the down-regulated genes, the genes that were not regulated, and upregulated genes, respectively, for each of the comparisons listed in the first column for batch 1 and batch 2, respectively. table 2.6 on page 36 and table 2.8 on page 39 are the tables for the summary of DEG genes obtained for batch 1. table 2.7 on page 37 and table 2.9 on page 40 are the tables for the summary of DEG genes obtained for batch 2.

MD plot of our expression data results were also plotted to perform further analysis on our results

figure 2.7 on page 41 shows the type comparison analysis of expression data results by comparing the types ILK1 vs. WT at different time points, 3 hrs., 6hrs., 12 hrs., and 0 hrs. respectively under the NaCl stress treatment for batch 1. Similarly, figure 2.8 on page 42 shows the type comparison analysis of expression data results by comparing the types ILK1 vs. WT at different time

points, 3 hrs., 6hrs., 12 hrs, and 0 hrs. respectively under the Flg22 stress treatment for batch 2. These plots in figure 2.7 on page 41 and figure 2.8 on page 42 were obtained at the threshold value of Log-FC set to 0.5 by performing *treat* function on *lmfit* model instead performing empirical Bayes moderation on our *lmfit* model directly. Later empirical Bayes moderation was used on *lmfit* model with no adjustment method and the *p-value* set to be less than equal to 0.05. figure 2.9 on page 43 shows gene expression comparison between WT type under NaCl treatment across all different time points and WT under control treatment at 0hrs. figure 2.10 on page 44 shows gene expression comparison between ILK1 type under NaCl treatment across all different time points and ILK1 under control treatment at 0hrs. Similarly, figure 2.11 on page 45 shows gene expression comparison between WT type under Flg22 treatment across all different time points and WT under control treatment at 0hrs. figure 2.12 on page 46 shows gene expression comparison between ILK1 type under Flg22 treatment across all different time points and ILK1 under control treatment at 0hrs. These plots were obtained at the threshold value of Log-FC set to 1 by performing the *treat* function on our *lmfit*. We used empirical Bayes moderation on our *lmfit* model.Here also, empirical Bayes moderation was used on *lmfit* model with no adjustment method and the *p-value* set to be less than equal to 0.05.

From the figures one can see that there were significantly larger difference in the number of DEGs that were obtained for different comparison sets. To obtain a significant number of differentially expressed genes, a stricter decision on our comparison fit was needed to be made. For this, we tried changing different parameters. We found that the number of differentially expressed genes showed a drastic change in numbers as the Log-FC threshold value is changed. The decideTests function parameter "adjusted method" also plays a significant role in identifying

and listing the differentially expressed gene. When we set this parameter to any method(provided by the function), it adjusts the results and the decision according to the adjusted *p-value*; however, if we keep it at value "none," the adjustment is no more prominent, and thus the results and the decision is obtained based on the p-value. This chapter summarized the finalized parameter's output that fit the data well and produced the desired results.

2.4 Conclusion

In this overall computational workflow, first reads were mapped to the reference Arabidopsis genome, and the data preprocessing was performed. The differential gene expression analysis of the following RNA sequencing Arabidopsis data shows a fair amount of differential expression genes and the abiotic and biotic stress for both types (WT and ILK1 type) and in all different time frames. The data were also checked for the batch effect and divided into two batches to analyze further. There was no set parameter found to work for all the different comparison tests altogether in the analysis. To obtain the significant level for differentially expressed genes, different parameters were set, and methods were used for different types of comparison set analysis. For example, we set different time series comparison set analysis parameters and different parameters for phenotypic comparison set analysis.

A significant difference was found in the number of differentially expressed genes in comparison set analysis types. A considerable difference in the number of differentially expressed genes was found in both comparison set analysis types. Consequently, in the next chapter, correlation and network analysis were performed on the differentially expressed genes obtained in this project.

Table 2.1

Grouping of samples according to the type, treatment, time for batch NaCl

Sample Names	Type	Treatment	Time	Groups
SL209921	WT	CONTROL	0h	WT CONTROL 0h
SL209922	WT	CONTROL	0h	WT CONTROL 0h
SL209923	WT	CONTROL	0h	WT CONTROL 0h
SL209924	WT	NaCl	3h	WT NaCl 3h
SL209925	WT	NaCl	3h	WT NaCl 3h
SL209926	WT	NaCl	3h	WT NaCl 3h
SL209927	WT	NaCl	6h	WT NaCl 6h
SL209928	WT	NaCl	6h	WT NaCl 6h
SL209937	WT	NaCl	6h	WT NaCl 6h
SL209938	WT	NaCl	12h	WT NaCl 12h
SL209939	WT	NaCl	12h	WT NaCl 12h
SL209940	WT	NaCl	12h	WT NaCl 12h
SL209941	ILK1	CONTROL	0h	ILK1 CONTROL 0h
SL209942	ILK1	CONTROL	0h	ILK1 CONTROL 0h
SL209943	ILK1	CONTROL	0h	ILK1 CONTROL 0h
SL209944	ILK1	NaCl	3h	ILK1 NaCl 3h
SL209945	ILK1	NaCl	3h	ILK1 NaCl 3h
SL209946	ILK1	NaCl	3h	ILK1 NaCl 3h
SL209947	ILK1	NaCl	6h	ILK1 NaCl 6h
SL209948	ILK1	NaCl	6h	ILK1 NaCl 6h
SL209949	ILK1	NaCl	6h	ILK1 NaCl 6h
SL209950	ILK1	NaCl	12h	ILK1 NaCl 12h
SL209951	ILK1	NaCl	12h	ILK1 NaCl 12h
SL209952	ILK1	NaCl	12h	ILK1 NaCl 12h

Table 2.2

Grouping of samples according to the type, treatment, time for batch Flg22

Sample Names	Type	Treatment	Time	Groups
SL209953	WT	CONTROL	0h	WT CONTROL 0h
SL209954	WT	CONTROL	0h	WT CONTROL 0h
SL209955	WT	CONTROL	0h	WT CONTROL 0h
SL209956	WT	Flg22	3h	WT Flg22 3h
SL209957	WT	Flg22	3h	WT Flg22 3h
SL209958	WT	Flg22	3h	WT Flg22 3h
SL209959	WT	Flg22	6h	WT Flg22 6h
SL209960	WT	Flg22	6h	WT Flg22 6h
SL209961	WT	Flg22	6h	WT Flg22 6h
SL209962	WT	Flg22	12h	WT Flg22 12h
SL209963	WT	Flg22	12h	WT Flg22 12h
SL209964	WT	Flg22	12h	WT Flg22 12h
SL209965	ILK1	CONTROL	0h	ILK1 CONTROL 0h
SL209966	ILK1	CONTROL	0h	ILK1 CONTROL 0h
SL209967	ILK1	CONTROL	0h	ILK1 CONTROL 0h
SL209968	ILK1	Flg22	3h	ILK1 Flg22 3h
SL209929	ILK1	Flg22	3h	ILK1 Flg22 3h
SL209930	ILK1	Flg22	3h	ILK1 Flg22 3h
SL209931	ILK1	Flg22	6h	ILK1 Flg22 6h
SL209932	ILK1	Flg22	6h	ILK1 Flg22 6h
SL209933	ILK1	Flg22	6h	ILK1 Flg22 6h
SL209934	ILK1	Flg22	12h	ILK1 Flg22 12h
SL209935	ILK1	Flg22	12h	ILK1 Flg22 12h
SL209936	ILK1	Flg22	12h	ILK1 Flg22 12h

Table 2.3

Mapping output for NaCl treated samples

Input Reads		Mapped reads
SL209921	15486879	95.09%
SL209922	15400488	95.22%
SL209923	14825823	94.95%
SL209924	16919859	93.53%
SL209925	16226945	93.23%
SL209926	14408122	92.73%
SL209927	15851632	94.36%
SL209928	16944518	93.59%
SL209937	12889348	94.41%
SL209938	16961915	94.24%
SL209939	18985000	93.96%
SL209940	21081276	93.50%
SL209941	19491568	95.23%
SL209942	14887305	95.23%
SL209943	14913368	95.48%
SL209944	14182911	94.86%
SL209945	18421895	94.69%
SL209946	15485404	93.68%
SL209947	15904885	93.15%
SL209948	16474980	93.64%
SL209949	14279886	94.83%
SL209950	16781790	92.27%
SL209951	17628332	94.43%
SL209952	19898531	93.57%

Table 2.4

Mapping output for Flg22 treated samples

Input Reads		Mapped reads
SL209953	17688489	94.84%
SL209954	15881217	94.77%
SL209955	19457697	93.73%
SL209956	12527033	94.41%
SL209957	21039873	94.58%
SL209958	18060975	94.69%
SL209959	25609849	88.52%
SL209960	19887661	91.74%
SL209961	16096277	95.69%
SL209962	16862499	95.08%
SL209963	14043704	94.77%
SL209964	17542353	94.43%
SL209965	18362587	95.36%
SL209966	17801017	94.03%
SL209967	17163927	93.78%
SL209968	20319127	92.50%
SL209929	16970583	92.09%
SL209930	14712327	93.91%
SL209931	19065595	92.10%
SL209932	21929347	94.59%
SL209933	14141932	93.59%
SL209934	10810009	94.53%
SL209935	18035483	93.32%
SL209936	17472622	92.94%

Table 2.5

Sample table

gene_id	21	22	23	24	25	26	27	28
AT1G01010	231	251	232	350	342	358	314	421
AT1G01020	181	177	204	275	226	180	194	230
AT1G01030	55	45	57	154	156	123	121	143
AT1G01040	527	772	651	607	679	590	675	738

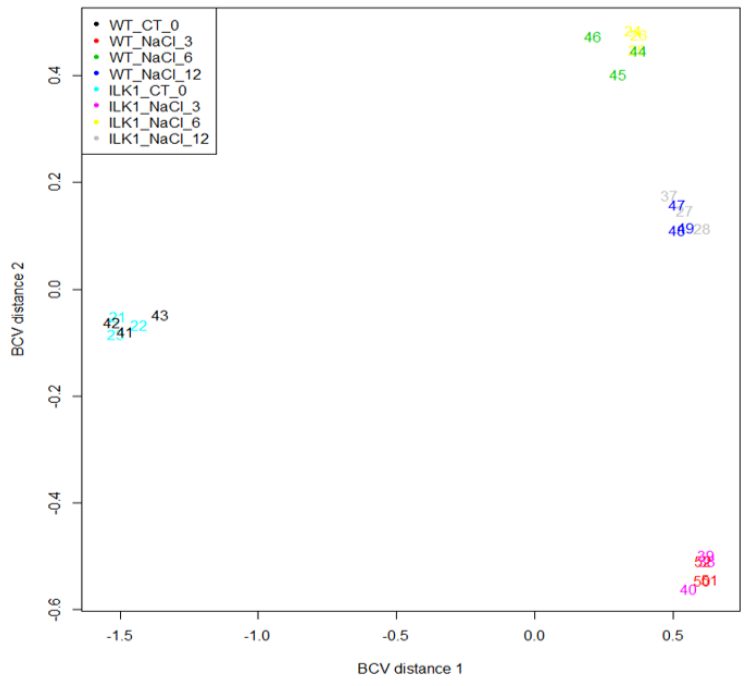

Figure 2.3

Multidimensional plot for batch 1

Table 2.6

Summary of DEG genes obtained using different Log-Fc≥ 0.5 threshold and *p-value*≤0.05 for batch 1

With Log-Fc ≥ 0.5, *p-value* ≤ 0.05, adjusted method = "none"	Down	Not Sign	Up
ILK1_CT_0_vs_WT_CT_0	28	20800	26
ILK1_NaCl_3_vs_WT_NaCl_3	35	20685	134
ILK1_NaCl_6_vs_WT_NaCl_6	35	20775	44
ILK1_NaCl_12_vs_WT_NaCl_12	28	20803	23

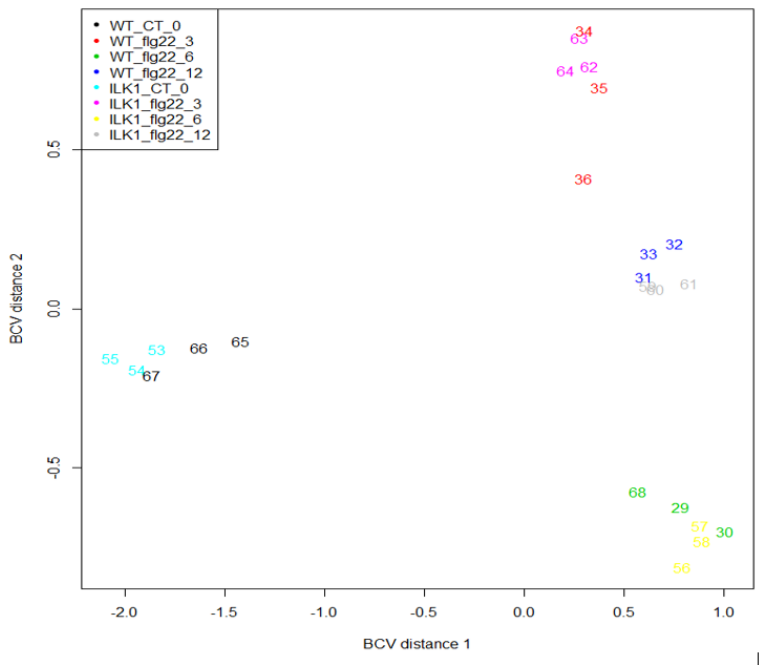

Figure 2.4

Multidimensional plot for batch 2

Table 2.7

Summary of DEG genes obtained using different Log-Fc≥ 0.5 threshold and *p-value*≤0.05 for batch 2

With Log-Fc ≥ 0.5, *p-value* ≤ 0.05, adjusted method = "none"	Down	Not Sign	Up
ILK1_CT_0_vs_WT_CT_0	64	19053	117
ILK1_flg22_3_vs_WT_flg22_3	171	19014	49
ILK1_flg22_6_vs_WT_flg22_6	56	19053	125
ILK1_flg22_12_vs_WT_flg22_12	343	18687	204

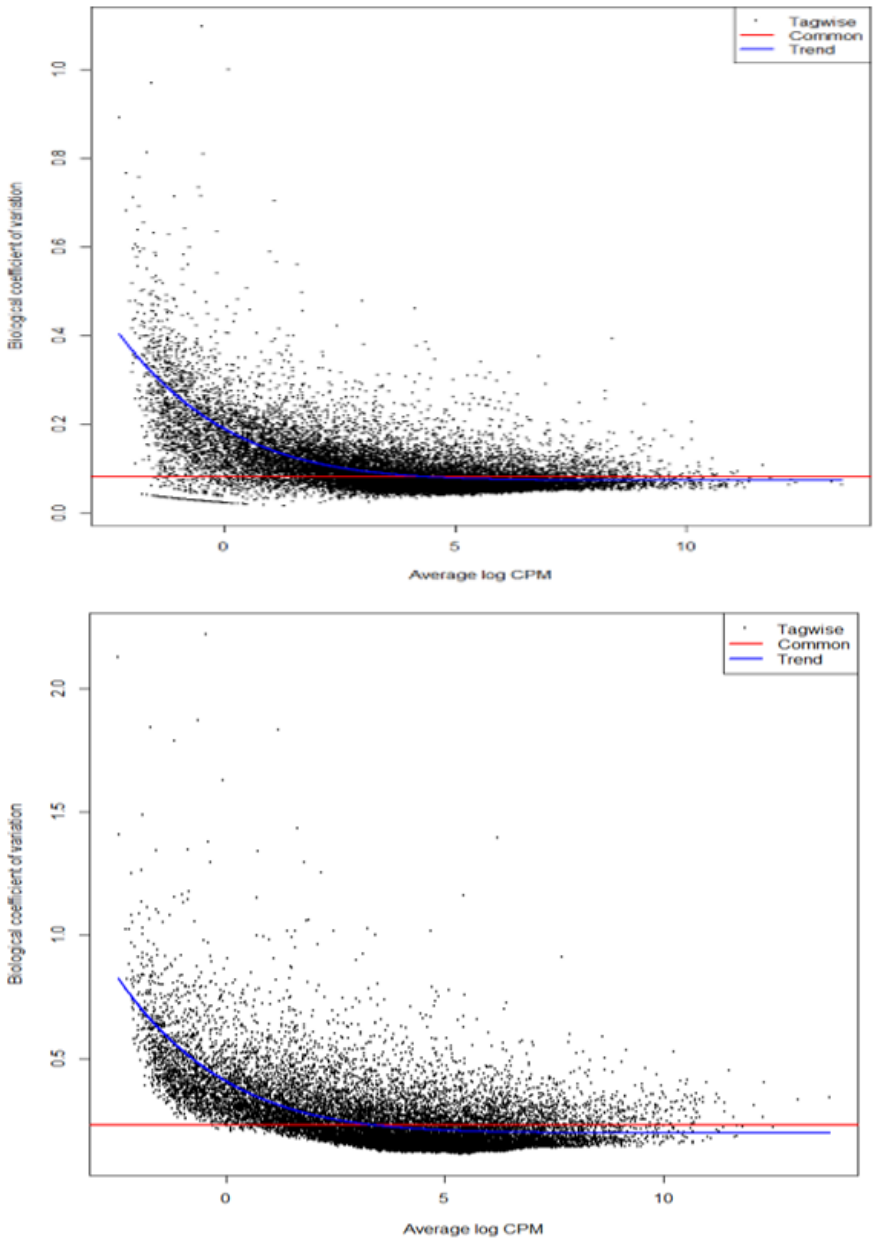

Figure 2.5

Trend wise dispersion of the data from Batch1 and Batch 2 respectively

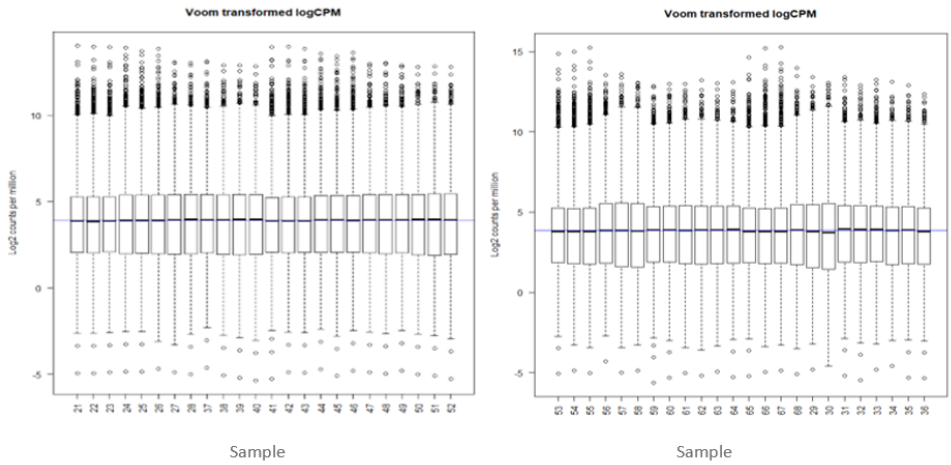

Figure 2.6

Voom normalized logCPM plot for batch 1 and batch 2 respectively

Table 2.8

Summary of DEG genes obtained using different Log-Fc≥ 0.5 threshold and *p-value*≤0.05 for batch 1

With Log-Fc ≥ 1, *p-value* ≤ 0.05, adjusted method = "BH"	Down	Not Sign	Up
WT_NaCl_3_vs_WT_CT_0	1351	17482	2021
WT_NaCl_6_vs_WT_CT_0	1559	17303	1992
WT_NaCl_12_vs_WT_CT_0	1940	16868	2046
ILK1_NaCl_3_vs_ILK1_CT_0	1285	17472	2097
ILK1_NaCl_6_vs_ILK1_CT_0	1511	17237	2106
ILK1_NaCl_12_vs_ILK1_CT_0	1960	16759	2135

Table 2.9

Summary of DEG genes obtained using different Log-Fc≥ 0.5 threshold and *p-value*≤0.05 for batch 2

With Log-Fc ≥ 1, *p-value* ≤ 0.05, adjusted method = "BH"	Down	Not Sign	Up
WT_flg22_3_vs_WT_CT_0	3142	12662	3430
WT_flg22_6_vs_WT_CT_0	2121	14867	2246
WT_flg22_12_vs_WT_CT_0	1022	16766	1446
ILK1_flg22_3_vs_ILK1_CT_0	2920	13393	2921
ILK1_flg22_6_vs_ILK1_CT_0	1558	15609	2067
ILK1_flg22_12_vs_ILK1_CT_0	1541	16364	1329

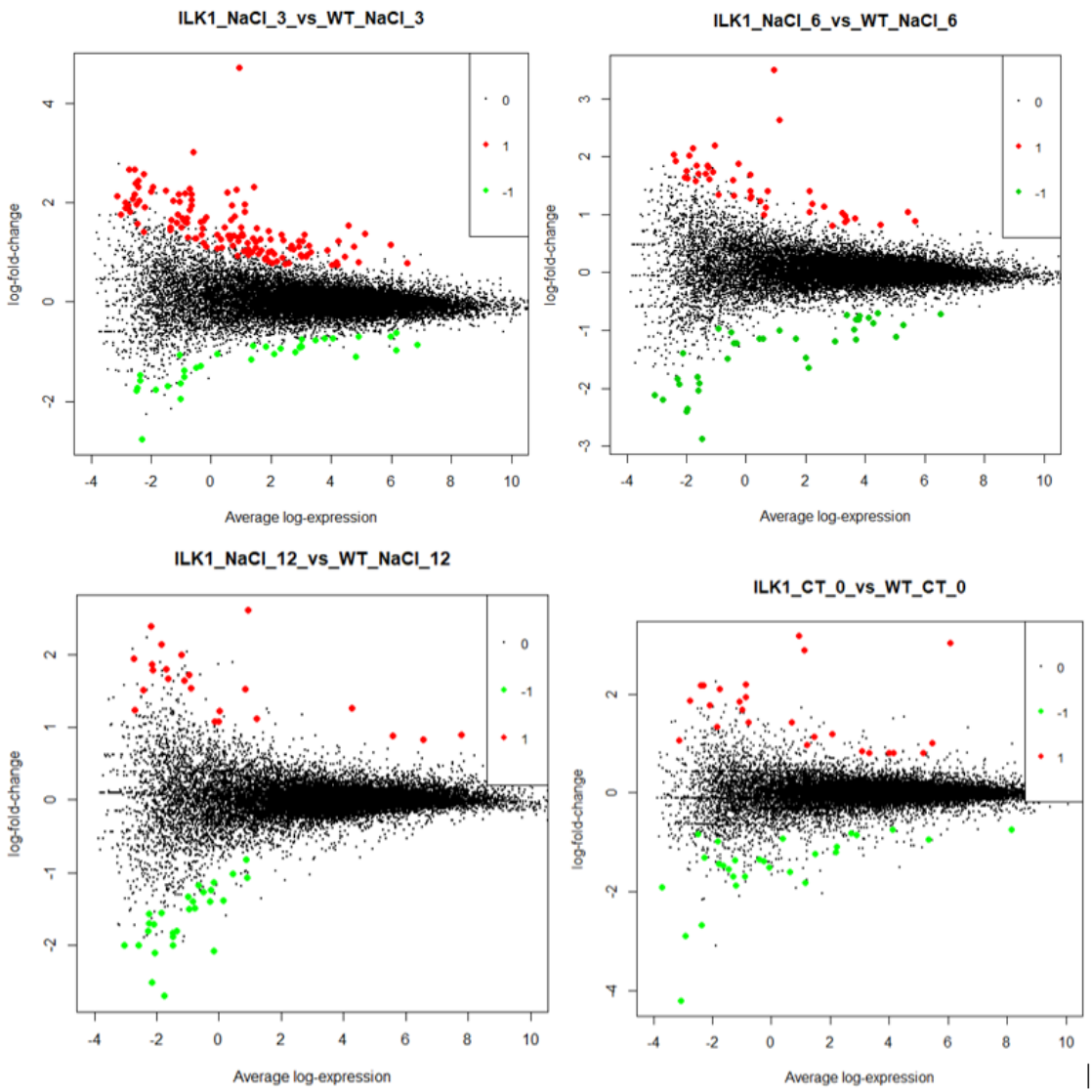

Figure 2.7

Batch 1 type comparisons analysis expression data results at 3 hrs., 6 hrs., 12 hrs. and control at 0 hrs respectively

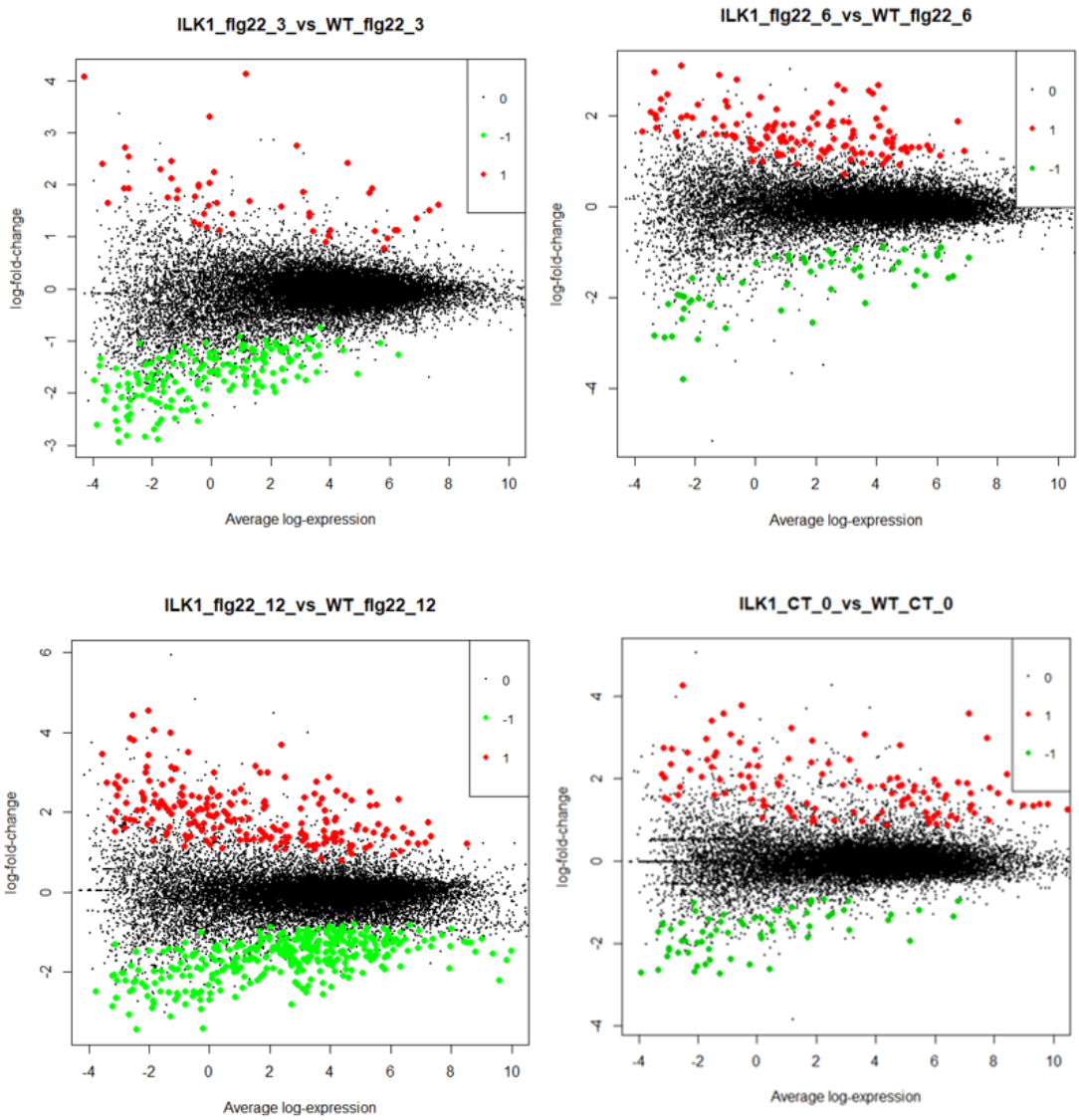

Figure 2.8

Batch 2 type comparisons analysis expression data results at 3 hrs., 6 hrs., 12 hrs. and control at 0 hrs respectively

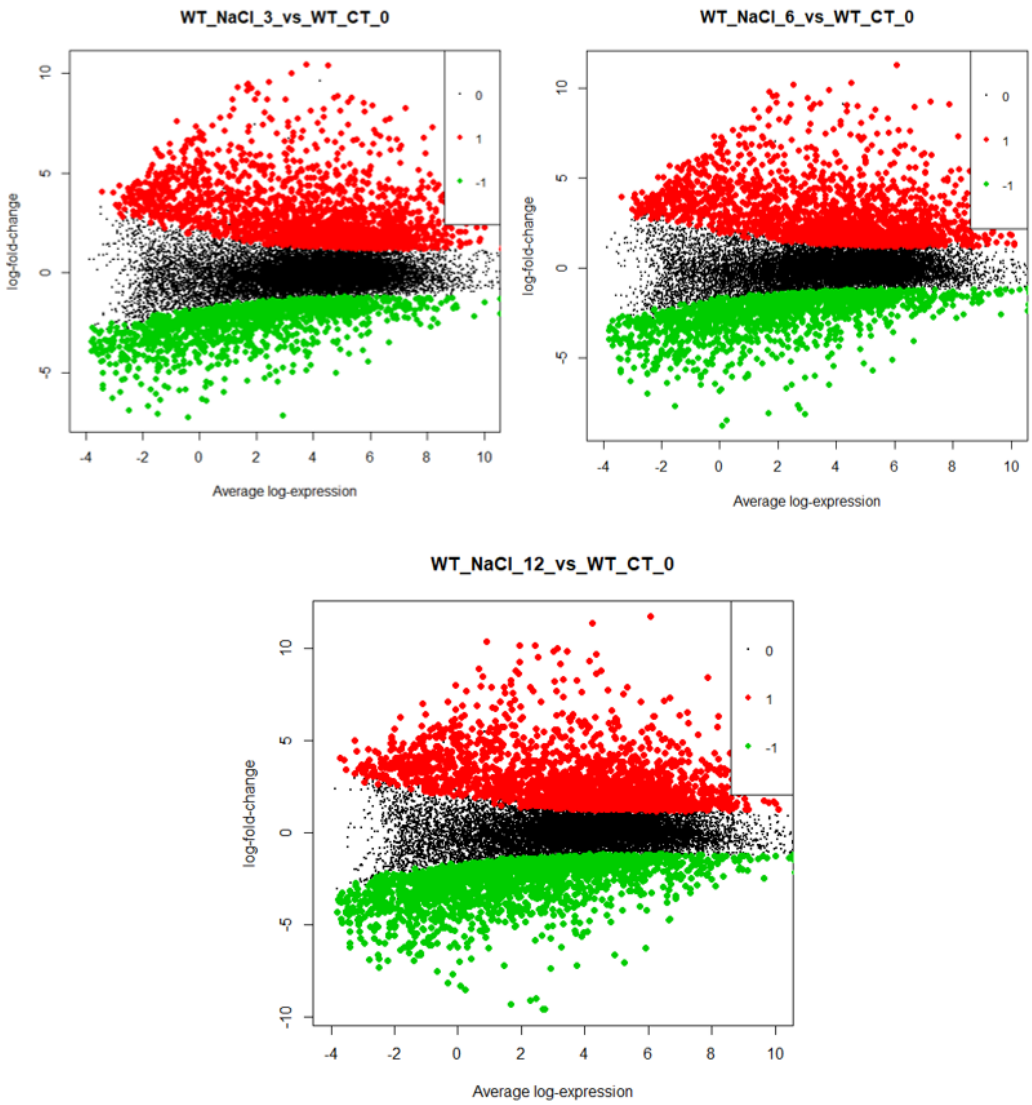

Figure 2.9

Batch 1 time comparisons analysis expression data results for the WT type sample

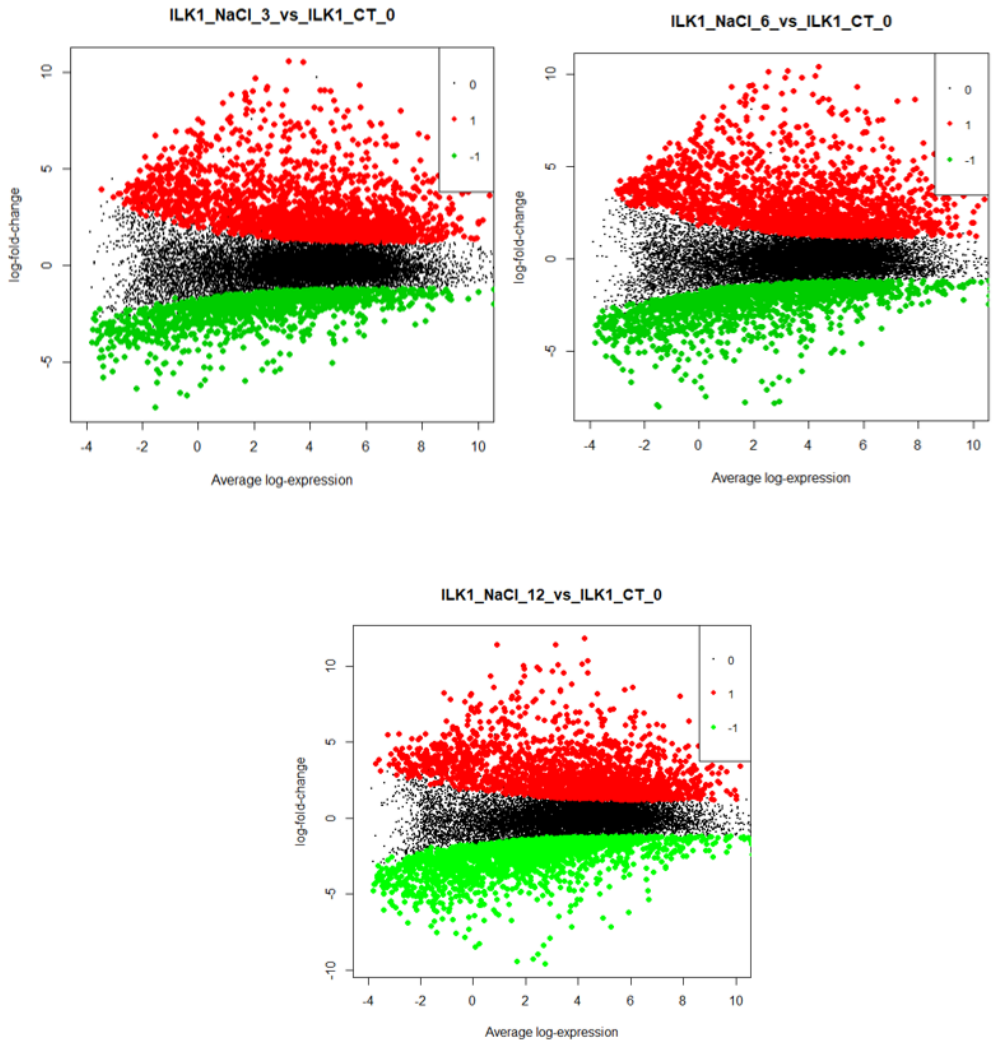

Figure 2.10

Batch 1 time comparisons analysis expression data results for the ILK1 type sample

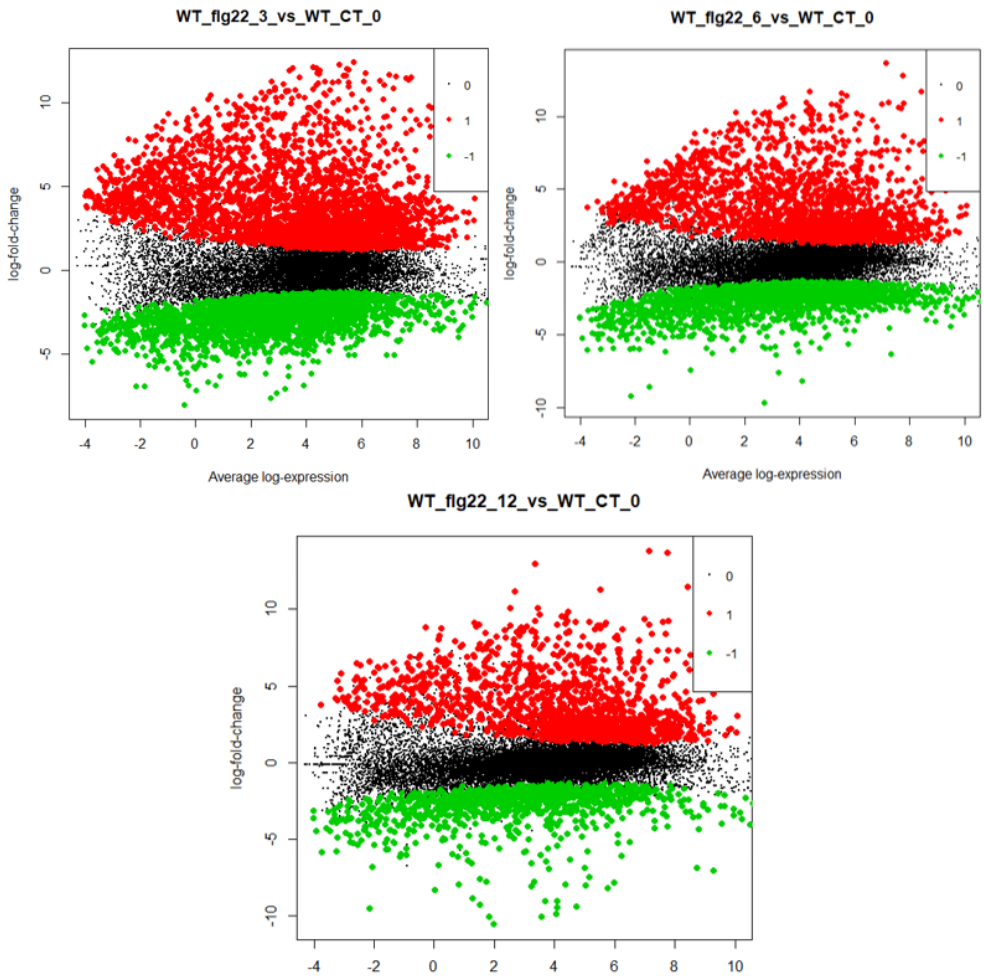

Figure 2.11

Batch 2 time comparisons analysis expression data results for the WT type sample

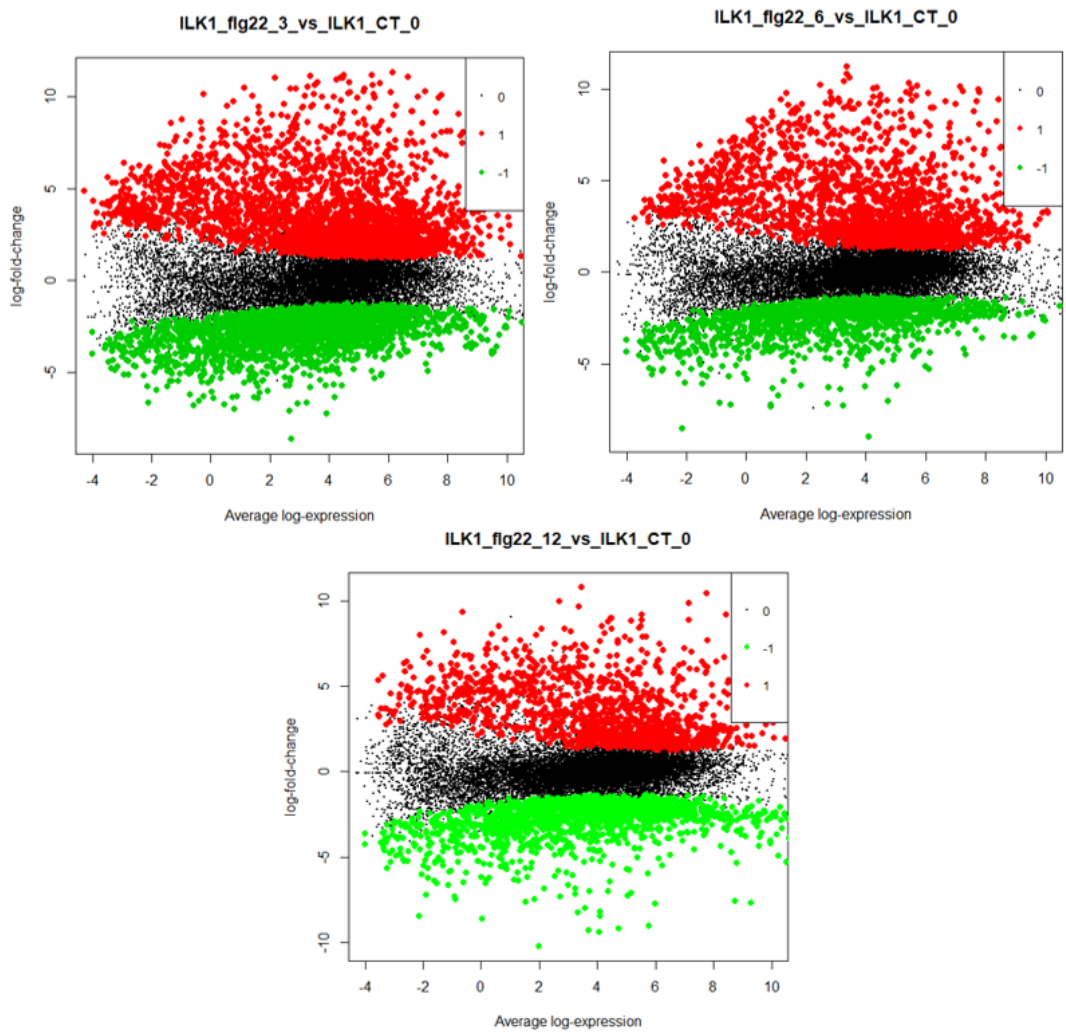

Figure 2.12

Batch 2 time comparisons analysis expression data results for the ILK1 type sample

CHAPTER III

METHODS FOR ANALYSIS OF SIGNALING NETWORKS FROM TIME SERIES DATA USING WCGNA AND IGRAPH

3.1 Introduction

This chapter discusses the time series analysis of the gene network for the gene expression data. These gene expression data were obtained by utilizing the RNA seq technology previously. The reason to use RNA sequencing technology was to capture the Arabidopsis plants' transcriptomic response with reduced ILK1(Integrin Linked Kinases) expression and the Arabidopsis plants with Wild-type mutants under two different environmental stress conditions and at four different time frames. Firstly, gene correlation network analysis on gene expression data was carried out, then modules or clusters in the DEG genes were identified. The data obtained by the correlation analysis gave an insight into the co-expressed genes. The data then was used to construct a gene co-expression network using genes as the nodes and their correlation as the edges. The goal was to study the interaction pattern found in our genes at different time points and use this interaction pattern for later analysis. After obtaining the preliminary results, different tools were utilized to create and analyze the co-expressed network from gene expression data. The results obtained could help study the pathway formed in the genes and help construct a network to find such patterns. Weighted correlation network analysis (WGCNA) pulled the patterns or clusters in our gene expression data set and found the hubbed genes on each of those clusters to accomplish this

task. WGCNA (Weighted gene Correlation network analysis) is an R software package, has great significance as it helps reduce high dimensional RNA seq gene expression data. It is also helpful for integrating multi-scale data such as complex phenotype or traits of the gene expression data. Originally it was developed to analyze the gene expression data and micro RNA data. In WGCNA, highly correlated (highly interconnected genes) clusters are found using the hierarchical clustering technique. The module eigengene is the first principle component of the module representing a given module's gene expression profile. It explains the highest proportion of variance among the genes in a particular module. The research in this chapter performed statistical analysis using R package WGCNA, which includes network construction, module detection, topological overlap matrix construction of the gene expression data by deciding them into two different batches. Each phenotype (Col0 and ILK1) was analyzed separately to find biologically interesting modules in our gene expression data then Go enrichment analysis was performed on each set of our study. Network analysis was completed to understand the gene expression data's functionality on each DEG set and form the gene regulatory network using the module-based analysis. In the later phase of network analysis, igraph was used to visualize and analyze the network's topology using igraph and determine the network's import features using igraph parameters. The data used for the network construction using igraph was obtained from the previous step in WGCNA. The network analysis using the igraph was divided into two different sections. The node-level or gene-level analysis as each node in the graph/network represents a gene and network-level analysis, where we analyze the entire network. Several other pieces of information were obtained from network analysis. Some of them are identification hubbed genes obtained from node level analysis, pattern level analysis

such as community detection in the gene network, and strongly connected components in the genes network are obtained from the entire network-level analysis.

3.2 Gene expression data preprocessing

The previous chapter studied the differentially expressed genes obtained by carrying out time-series comparisons set analysis and genotypic comparison set analysis. The RNA seq data obtained from Illumina HiSeq 2000 was then passed through a computational workflow to detect the DEG genes received from the previous work using the computational workflow was then loaded into R as an R data frame for further correlation analysis. In this step, since the filtering of lowly expressed genes using Edge R has already been performed, the normalized counts of the data were used to complete the analysis. The analysis was carried out with the DEG genes. We used the normalized value of the raw count's data as the input for each part of the given study. WGCNA package was used to determine the related modules found in each set of differentially expressed genes.

3.2.1 Adjacency Matrix

We know that a network is specified by an asymmetric matrix called the adjacency matrix, determining whether the two genes are connected or not connected [41]. The *nxn* adjacency matrix $A = [a\_ij]$ is obtained from nxn similarity matrix $S = [s_{ij}]$, which measures the similarity between the gene expression profiles across all the samples. Similarity matrix $S = [s_{ij}]$ is defined as the absolute value of correlation between each pair of gene ii and jj. For example, the correlation coefficient, Pearson correlation, could be used to find out the co-expression in a pair of genes in our dataset. These correlation coefficients can be negative (for negative relation) and positive (for positive correlation) for a pair of genes in our dataset. In this dissertation, the similarity matrix

consisting of correlation matrix is used. The baseline methodology is to study the interaction between the genes. The similarity matrix is mathematically transformed into a WGCNA adjacency matrix which obeys the scale-free topology [41]. The requirement here in this chapter is to construct a scale-free network for gene expression data. The term weight properties refer to the network, whether; it is weighted or unweighted. We can select the network to be signed or unsigned and determine the interaction strength for a network. For a network whose entries are 0 or 1 for an unweighted network, particularly specifies whether the two genes are connected or not connected. A network whose entries are real numbers (in the case of a weighted network) determines the strength of the connection between a pair of genes. We can also choose the signed network as the graph property to account for negative and positive correlations. The aim was to find out genes that are overexpressed for the network formation that makes up a module. This research used a signed network strategy to account for both positive and negative correlations as these correlation coefficients are transformed into the network. Additionally, in this project, Pearson correlation was used to find the correlation between the genes for forming a similarity matrix. The relationship between the adjacency matrix and similarity matrix is given as 0.5*(1+similarity matrix)^{sft}. Here *sft* is the soft threshold power taken from the Soft thresholding measure.

A Soft thresholding measure was used to choose the adjacency matrix power. There are few factors to consider when selecting the power as this parameter defines the sensitivity and specificity of the connection strength between two pairs of genes. The Scale-free topology method criterion was used to choose the appropriate soft threshold. The task is to turn the WGCNA matrix into a Topological Overlap Measure (TOM) to minimize noise and spurious associations. The topological overlap matrix is the measure of the strength of connectivity between the two nodes. It finds out

how each gene is a topological overlap on all the other genes based on the common neighbors' factor that each of these genes (nodes) shares [75]. In other words, if someone considers placing a framework of the entire network, TOM is simply the co-occurrence between two nodes by factoring in all the other nodes each is connected to. The equation 3.1 introduced in the paper [41] helps to find the WGCNA adjacency matrix. Topological overlap matrix is given as:

$$w_i = \frac{\sum_u a_{ui} a_{uj} + a_{ij}}{min\{K_j K_i\} + 1 - a_{ij}} \tag{3.1}$$

Here K_i and K_j are the network connectivity of a node i and j, which is defined as the number of direct connections with other nodes. u is a random neighbor node. For a pair of genes, the TOM is high if the pair of genes have many shared neighbors, which in turn implies that genes similar expression pattern [41]. To continue with the gene clustering, we need to find out the dissimilarity matrix, which is given in the equation below

$$d_i = 1 - w_i \tag{3.2}$$

3.2.2 Gene Module detection and intermodular connectivity

After calculating the overlap matrix, the next step is module detection. In this reference, the modules are the clusters of closely interconnected nodes. The clusters of closely interconnected nodes, in this case, are the genes with high topological overlaps (collected from the TOM matrix). The genes with a higher association are clustered together. High topological overlaps mean each gene's affinity with the other genes. These clusters are obtained using an unsupervised clustering technique using the WGCNA package. The WGCNA package performs average linkage hierarchical clustering to get these clusters using the dissimilarity matrix obtained from Topological Overlap Matrix [41]. The R function *hclust* is used to create a dendrogram using a hierarchical clustering

tree then call the hierarchical clustering function. WGCNA detects the modules as branches and cuts the dynamic tree to perform module identification. A cluster dendrogram is obtained of all the genes. In some cases where the identified modules are way too high, further merging was performed even after completing the initial clustering. The idea is to find out unique modules whose expression profiles are very similar. Further identification and merging of the module whose expression profiles are very similar was carried out. To carry out this process, WGCNA first quantifies the co-expression similarity of the entire module. After that, it calculates the eigengenes and clusters these eigengenes based on their correlation. Later the WGCNA, then merge the modules with a similar expression profile. The resultant plots of the merged clusters obtained on DEG genes using WGCNA functions are discussed later.

3.2.3 Network Analysis and visualization

This section covers the basics of importing, visualizing, and analyzing the network graph obtained from WGCNA. The node represents the genes in the data, and the links between the nodes represent the correlation between the pair of genes. The adjacency matrix is converted into a network data frame for better visualization and analysis. Edges that had a correlation value lesser than 0.5 were removed. A network data frame had three columns, one for the source node, one for the target node, and one for the weight that contains the value of the correlation between the genes in the source node and target node. In the network, the nodes' color depicts the module's color, and the nodes' size represents the node's degree (number of links of the particular node). The description of the degree of nodes is given in the next section.

In this section, the analysis have been divided into two parts:

1. Node and edge level analysis level analysis

2. Entire network/graph level analysis

This kind of analysis is called network topology analysis and it depends on location of nodes. In Node and edge level analysis, the node degree and centrality measures are discussed. The centrality index of a graph [60] is calculated using a different methodology, the nodes on node betweenness and closeness measures of nodes we realso analyzed. The critical property of a node is the degree of nodes. In an undirected graph, the node degree is represented as the total number of links. The following important metric for node-level analysis is all kinds of centrality scores. Four different kinds of network centrality scores are studied. The first one is strength centrality. The strength centrality or weighted degree centrality defines the sum of the weights of all the given nodes [11]. The second one is the closeness centrality; the closeness centrality of the node gives us an idea about how far the other nodes are from the particular node taken into consideration [11]. In this context, the nodes/genes with higher closeness centrality signify the nodes/genes that have a stronger correlation with all the nodes/genes. The third type is betweenness centrality. The betweenness of a node is the number of the shortest paths between that particular node and all the other nodes in that network [11]. The node with a high betweenness score acts as the vital node/gene as it might have an enormous impact on the network if the particular node is removed. The last one is Eigenvalue centrality; Eigenvector centrality is defined as the principal eigenvector's values for the network when represented as a matrix. Under this metric, a node's centrality score is proportional to its connections' centrality scores [11]. The metrics mentioned earlier as a histogram to study the distribution of nodes across different metrics have been plotted. The entire network/graph level analysis found how the genes are grouped based on the different topological parameters. The different metrics used here to study are finding the

clusters of the network's strongly connected components and communities. Strongly connected components are the different subgraphs obtained from the original graph, which are topologically strongly connected. For each pair of subgraphs, there is a common path that connects two subgraphs distinctively. The process of finding this subgraph is based on either a depth-first search algorithm or a breadth-first search algorithm. This method helps extract the subgraph and analyze the individual subgraph as these genes in these subgraphs will have a similar value of correlations.

3.3 Results and Discussion

In this section, results were collected for all the sets of DEG genes are discussed. In this type of time-series analysis, DEGs were collected for NaCl treated samples and then left joined on normalized counts for all the genes obtained from the EdgeR results to get the normalized counts of NaCl treated DEG genes'. In total, twenty-four samples were included with subtypes in coexpression analysis, each for each treatment. DEG gene expression data for 418 DEGs for NaCl treated samples were used. So, the expression data for the first set of analyses had 418 rows and 24 columns. Similarly, for the second set of time-series analyses, DEGs for flg22 treated samples were collected and then left joined on normalized counts for all the EdgeR results' genes to get the normalized flg22 treated DEG genes'. DEG gene expression data for 1036 DEGs for flg22 treated samples were used. So, the expression data for the first set of analyses had 1036 rows and 24 columns.

figure 3.2 on page 56 and figure 3.1 on the next page shows the cluster dendrogram created by the average linkage hierarchical clustering for NaCl treated samples and flg22 treated samples respectiely.

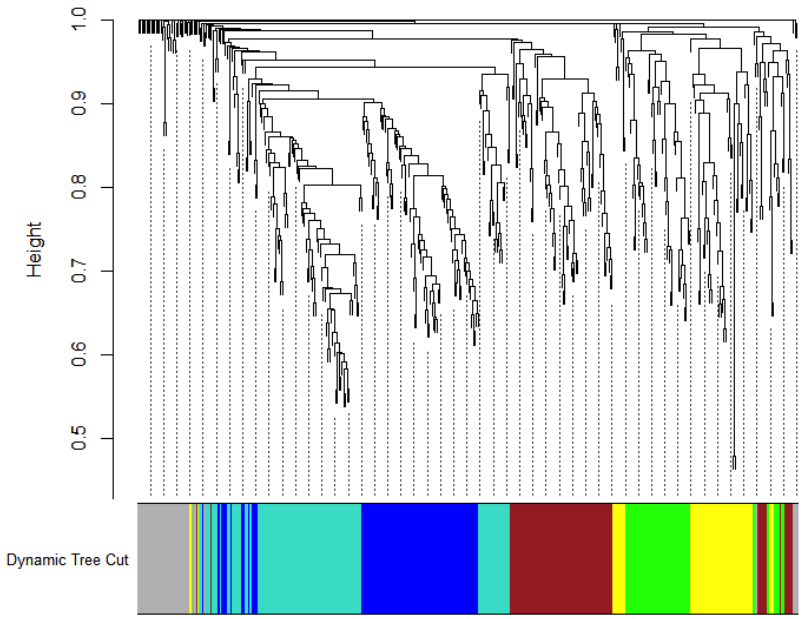

Figure 3.1

Cluster dendrogram created by the average linkage hierarchical clustering for NaCl treatment

Gene and samples were checked for missing values. Next, using average linkage hierarchical clustering, sample outliers are detected. After standardization, no samples were found to be an outlier in any of our analysis sets, So no samples were removed in the analysis. Using the scale-free topology criterion, the threshold power was chosen. To validate the choice of scale-free topology, one must check the R^2 value and the mean number of connections k. After analyzing several plots, a value of 20 was chosen as the threshold power for analysis data that belongs to NaCl treatment and 7 for analysis data that belongs to flg22 treatment. figure 3.3 on page 60 and figure 3.4 on page 61 shows the igraph network visualization NaCl treated samples and flg22 treated samples respectiely.

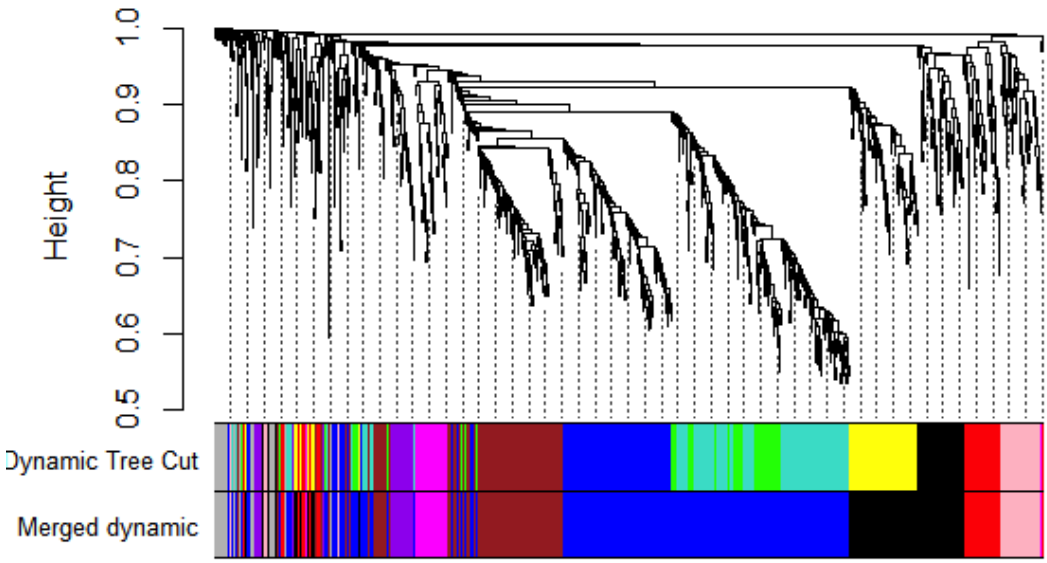

Figure 3.2

Cluster dendrogram created by the average linkage hierarchical clustering for flg22 treatment

figure 3.5 on page 62 and figure 3.6 on page 62 show the igraph node degree comparisons between NaCl treated samples and flg22 treated samples, respectively. In the figures, one can see the different metrics' histograms across all the nodes. From these distributions, we get an idea about the density of connection in our network. The histogram representing the degree of node and strength centrality depicts the connectivity pattern. By looking at the distribution of betweenness centrality scores, one can determine how centralized a network is. It can be seen from the plots for the DEG gene data for flg22 treated samples and DEG gene data for NaCl treated samples that there are many nodes/genes with low closeness centrality and very few nodes with high closeness centrality; this depicts that the network is highly centralized.

In the next step, hubbed genes were determined, basically the highly importing genes in the topological network. To obtain hubbed genes, the gene with the degree of nodes more significant

Table 3.1

Top Hubbed genes in NaCl treated samples

Cluster module	NaCl treated Samples
blue	AT1G22370
brown	AT2G19970
green	AT3G06530
turquoise	AT5G03210
yellow	AT3G10910

than 30, an eigenvalue centrality scores greater than 0.8, and a strength centrality score greater than 20 in NaCl treated Samples were extracted. Similarly, in flg22 treated samples, the gene with degree of nodes greater than 80, an eigenvalue centrality score greater than 0.8, and a strength centrality score greater than 60 were extracted. table 3.1 and table 3.2 on the next page show the topped hubbed genes obtained for NaCl treated sample and Flg22 treated sample, respectively.

3.3.1 Final Analysis

The final analysis found that genes that were differentially expressed in ilk1-1 post-flg22 treatment were associated with the cell wall synthesis, plasma membrane interface, microtubule cytoskeleton, and plant immunity. The resulting analysis is reported in figure 3.7 on page 63. The analysis found that around thirty DEGs linked to cell wall biosynthesis and modification, around 20 DEGs encoding anchored membrane proteins, 18 immune response DEGs, and 13 microtubule-associated DEGs had the defective regulation ilk1-1 line [20]. Signaling components and response genes mediated by defense hormones abscisic acid (ABA), jasmonic acid (JA), and salicylic acid (SA), and the growth hormone auxin were also found to be overrepresented among ilk1-1 DEGs. These resultant genes included most of the upregulated expression levels at 0

Table 3.2

Top Hubbed genes in flg22 treated samples

Cluster module	flg22 treated Samples
black	AT3G12580
blue	AT2G18690
brown	AT1G74070
magenta	AT5G15950
pink	AT4G23430
purple	AT2G25900
red	AT5G66270

and 12 h post-treatment with flg22 [20]. DEGs were classified functionally using gene ontology (GO) terms using Araport annotations and Panther Gene Ontology (http://pantherdb.org/) tools. figure 3.8 on page 64 shows the result of Functional annotation clustering and enrichment analysis of the repressed and upregulated DEGs in ilk1-1 in the Arabidopsis genome, performed with the GeneOntology Panther tool. figure 3.9 on page 65 shows the Gene Ontology (GO) term clusters of DEGs with high enrichment scores. In both the figures, figure 3.8 on page 64 and figure 3.9 on page 65 repressed DEGs are blue, and upregulated DEGs are yellow.

3.4 Conclusion

In this chapter, the correlation pattern in our DEGs dataset was found using correlation analysis. This correlation pattern was analyzed using hierarchical clustering. From the hierarchical clustering, there were several modules/clusters extracted and further subjected to downstream analysis. From the downstream analysis, the downstream analysis of modules/ clusters results performed on DEGs, some genes were found to be linked to cell wall biosynthesis and modification. In the ILK1 type mutant, some genes were found to be responsible for many other biological processes.

The graphical analysis using the igraph package provided a brief overview of the topological structure of pairwise gene correlation. Through the topological analysis, different interaction patterns within the DEGs could be studied. Through the topological structure analysis, genes based on high interactiveness parameters were extracted.

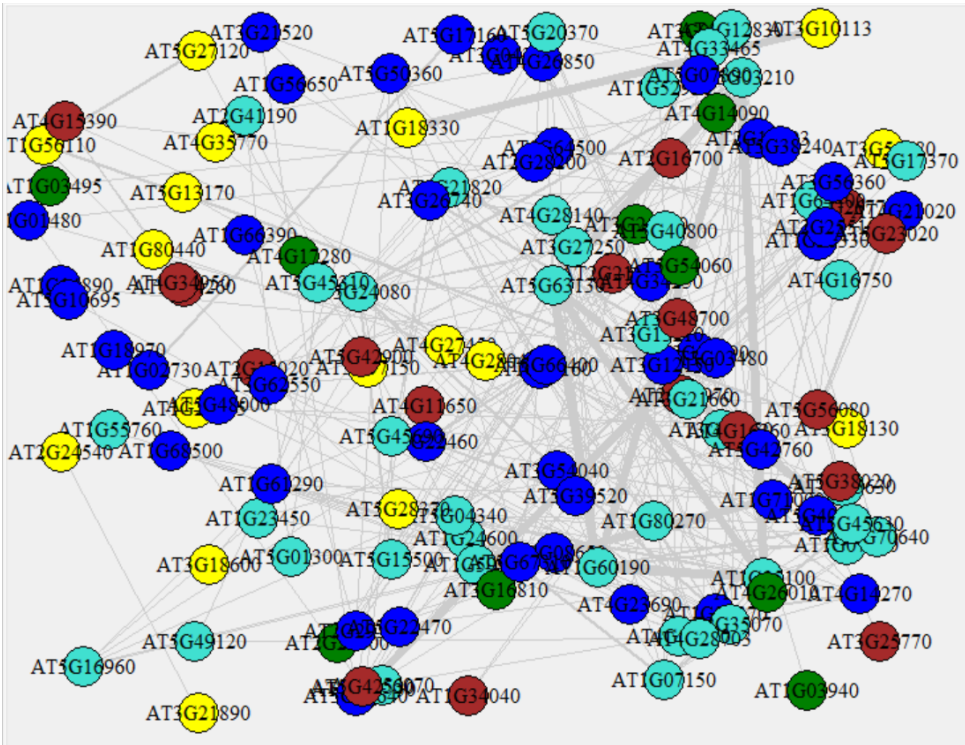

Figure 3.3

Igraph network visualization for NaCl treated samples

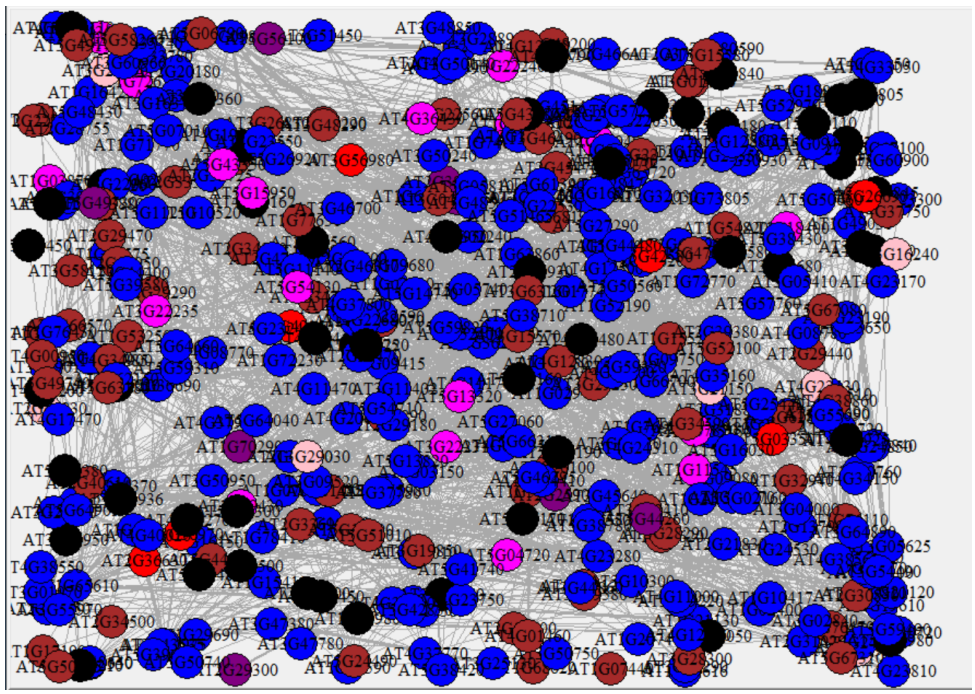

Figure 3.4

Igraph network visualization for flg22 treated samples

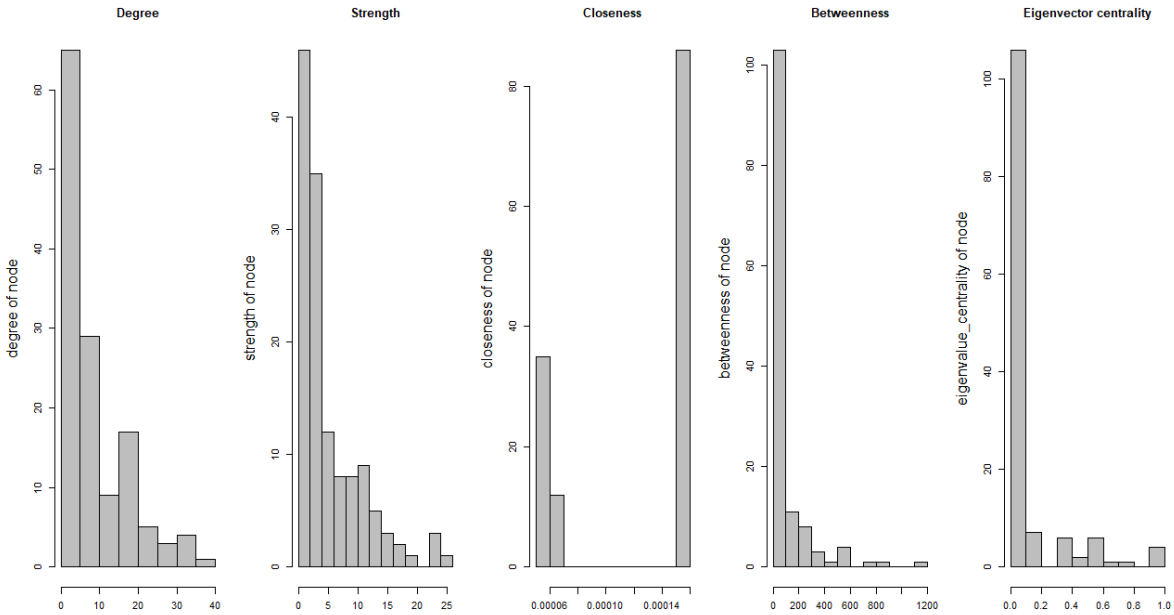

Figure 3.5

Node degree comparison for NaCl treated samples

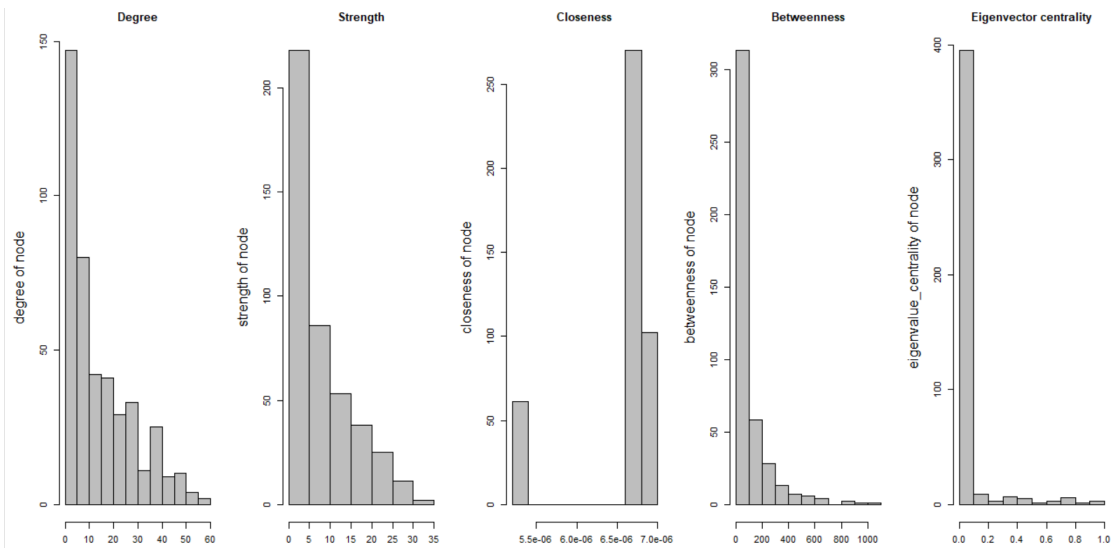

Figure 3.6

Node degree comparison for flg22 treated samples

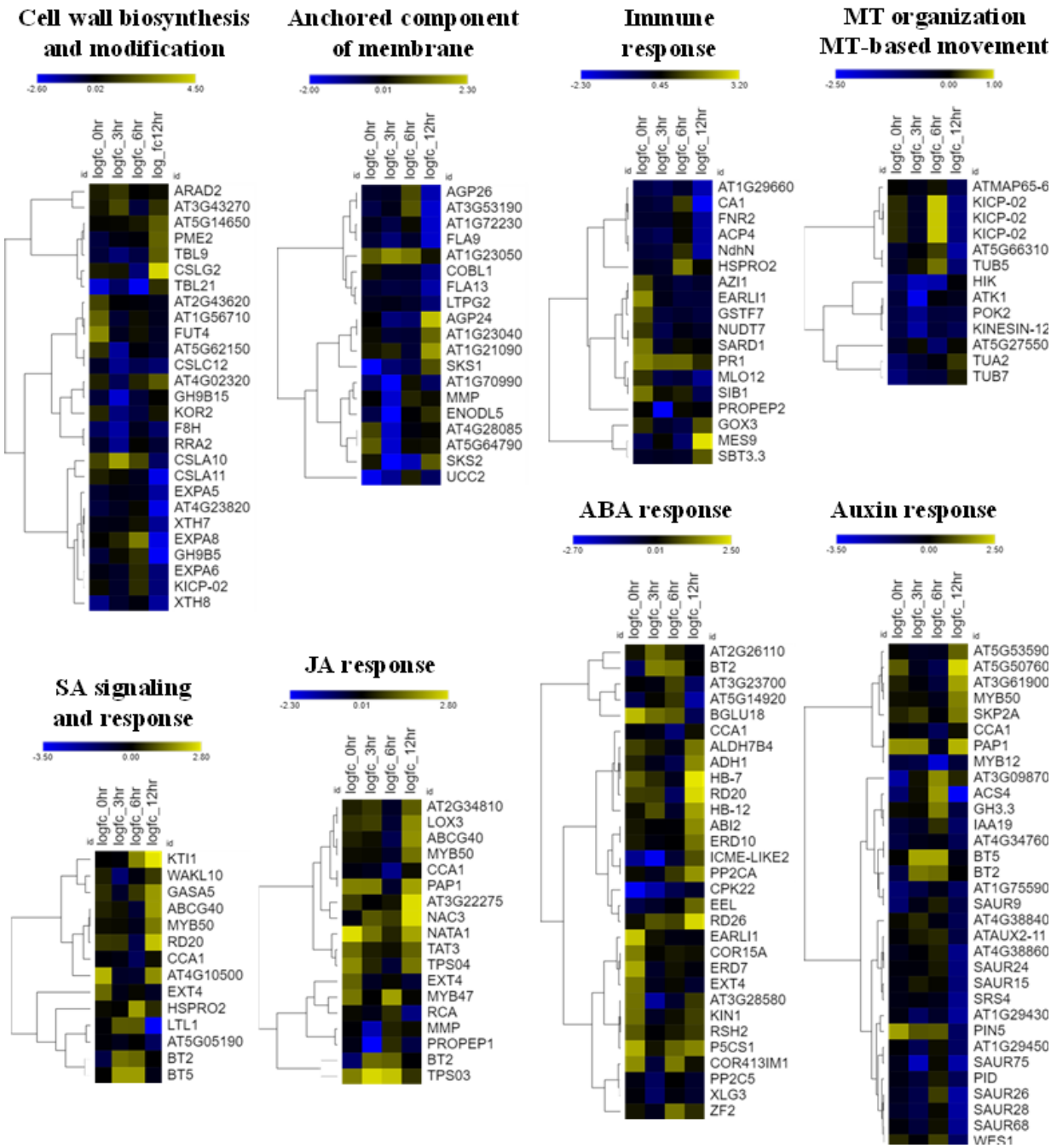

Figure 3.7

Heat map dendrograms of differentially expressed genes (DEGs) in ilk1-1 clustered according to expression (logarithm of fold-change, logFC) values in controls and at 3, 6- and 12-hours post-treatment (hpt) with flg22. Color scheme: blue: downregulated, and yellow: upregulated DEGs.

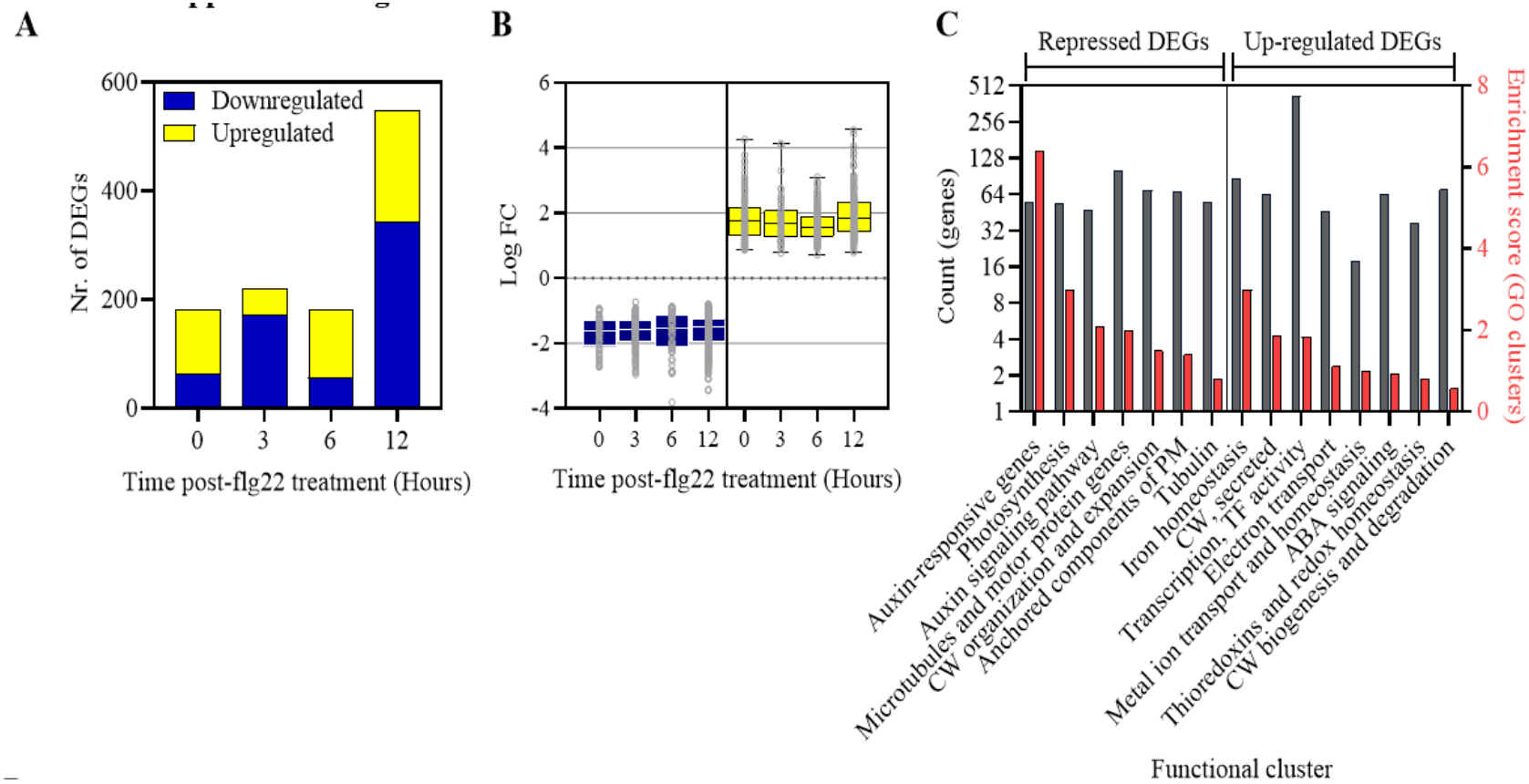

Figure 3.8

Functional annotation clustering and enrichment analysis of the repressed and upregulated DEGs in ilk1-1 relative to the all protein-coding genes in the Arabidopsis genome, performed with the GeneOntology Panther tool (http://pantherdb.org/).

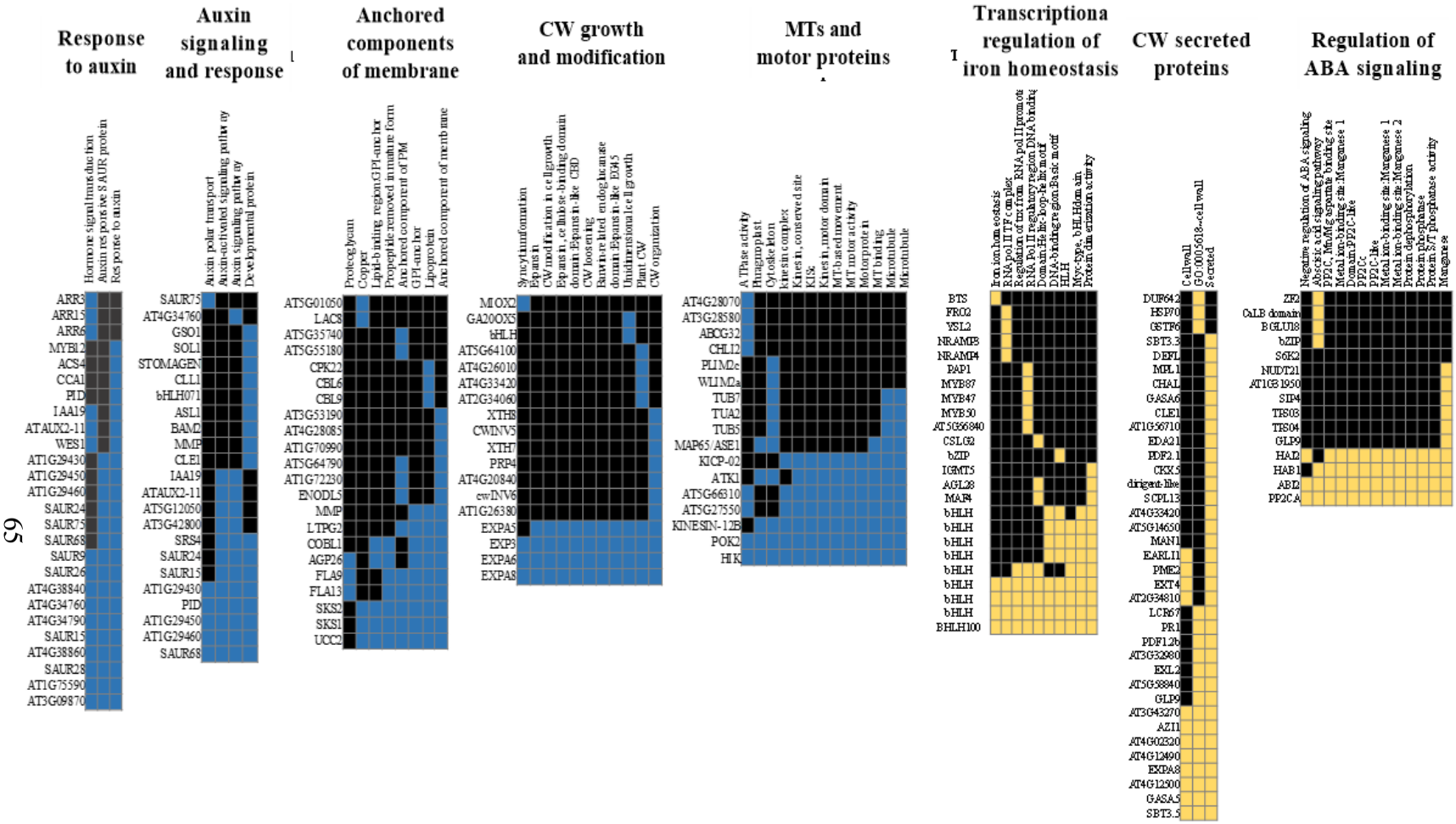

Figure 3.9

Gene Ontology (GO) term clusters of repressed (blue) and upregulated (yellow) DEGs with high enrichment scores. In all these figure the repressed DEGs are blue and upregulated DEGs are yellow.

CHAPTER IV

NETSEEKR: A NETWORK ANALYSIS PIPELINE FOR RNASEQ TIME SERIES DATA

4.1 Introduction

The core software pipeline presented in this chapter performs alignment of reads, differential gene expression analysis, gene ontology enrichment analysis, and network analysis of differentially expressed genes. It includes various parameter setups and exemplifies two different methodologies for differential gene expression analysis along with the pipeline. The input of the pipeline are sets of files containing raw reads from the high throughput sequencing step. These reads were previously processed with quality control and trimmed using Trimmomatic [10] software.

The first step of the pipeline is to align processed reads to genome positions using the gene annotation file. NetSeekR currently implements two read mapping tools, STAR, and Kallisto, to allow comparative transcript quantification evaluation. The next step in the pipeline is the identification of differentially expressed genes.

The data obtained in the previous step is a subset for mapping counts data to conditions, loaded into an R data frame, and converted into matrix format. Two options are available at this stage of the pipeline edgeR [58], and Sleuth [52] offering two options for gene expression modeling, complementary methods for identification of differentially expressed genes, and multiple methods for data normalization and visualization. Both software tools implement statistical methodologies for car-

rying out various operations on gene counts data such as filtering, normalization, multidimensional plotting, and clustering.

The next stage of the pipeline aims to use statistical tools to predict gene networks and to compute functional overrepresentation of differentially expressed genes obtained in the previous step. There are currently included two methods to create and analyze correlated gene expression networks and to infer regulatory networks. Gene network analysis aims to identify pathways associated with the experimental treatment by mining differential gene expression patterns. To accomplish this task, Weighted Correlation Network Analysis (WGCNA) [41] was used to identify patterns or clusters in gene expression data and the Dynamic Regulatory Events Miner (DREM) [61] to identify regulatory patterns that drive the observed gene expression. The NetSeekR pipeline uses the WGCNA package's functionality to perform network construction, module detection, and topological overlap matrix construction of gene expression.

The pipeline's output can be used to find out biologically interesting modules by correlating gene expression changes with phenotype changes when provided by the experimental design. The pipeline also uses GO enrichment analysis to mine the functionality of gene expression data on selected sets of genes identified in The analysis. The pipeline implements DREM to infer regulatory networks from time series of gene expression or series of treatments and/or genotype variation data. Finally, the igraph [15] R packages are used to conduct network analysis on the differentially expressed gene (DEG) networks using custom scripts for mapping overlapping nodes between DEG sets and gene networks from public data sources and for visualization of network analyses results.

The pipeline processing begins by reading several arguments from a configuration file and making a directory tree to store data. Arguments to the pipeline include a string specifying the

analysis type, covariates to account for differential testing, a path to a differential gene expression sample comparison matrix (DGECM), a path to an experimental design matrix on which DGECM is based, parameters for differential gene expression analysis (a significance level for statistical testing of differential gene expression), a path to a directory containing raw read sequences, a path to a selected output directory, a path to a reference genome, and Boolean flags for specifying whether to implement Kallisto or STAR pipelines or both. The directory tree is structured. The current working directory is the top-most node, with subdirectories within the tree for DREM, edgeR, Kallisto, network analysis, Sleuth, and WGCNA data.

The first two steps of the pipeline consist of building a transcriptome index and quantifying reads. Both STAR and Kallisto operations are executed from the R environment through the Linux terminal by passing a bash script assembled from arguments. Arguments are passed to NetSeekR from the configuration file to match the type of pipeline (STAR or Kallisto) being executed. The R-implemented bash scripts for index-building operations use a genome annotation file reference in the configuration file (using the path as an input argument) to generate the transcriptome index for Kallisto/STAR in the specified data storage directory created in the directory tree.

The read quantification step is also directed to the Linux terminal from within the R code. The quantification commands sent to the terminal are concatenated together in a bash script using variables given to the configuration file's function. The bash script for running the STAR/Kallisto quantification method is called from within the R environment. The STAR/Kallisto quantification method creates a directory for each sample quantified with directory names derived from sample identifiers. Arguments to the command include: the shell script name, the read data directory path,

the path to the Kallisto or STAR index file, three digits (used for manipulating the read data file names), a directory path for storing quantification results and the log file directory path.

Running a bash script from within the R environment has the advantage that large datasets (mRNA sequencing read data files) do not need to be loaded into R's memory, saving time and avoiding memory size issues. The bash script iterates over all mRNA sample datasets in the order in which they appear in the directory and compares their orientation. Quantification data and the design matrix are accessed in the next step for differential gene expression (DGE) computation. The experimental design matrix (DM) is a file that consists of paths to samples and respective characteristics defined in the experimental design before sequencing.

The experimental design file is string-processed to provide a dataset with references to DGE software variables to use. The design matrix used as input has to conform to a specified format and be edited with a file editing program. The DM is used in tandem with the DGECM supplied to the pipeline; the DGECM groups together the DM samples to be compared when testing for differential gene expression. The DGECM columns are combined in the R code row-wise with a 'logical or' string between each sample identifier to select test samples. The number of rows in the DGECM corresponds to the total number of sample comparisons in the analysis, each row corresponding to one comparison instance. The cells contain sample identifiers matching the samples described in the experimental design file.

The last component of the pipeline is a network analysis of the data. This involves processing WGCNA input (differentially expressed genes obtained from edgeR/Sleuth and their estimated expression values) to generate correlation networks. Next, GO enrichment analysis is carried out on the same set of rearranged data using topGo software [4]. The last step is network construction

and visualization using igraph. The pipeline workflow is shown in figure 4.1 on page 81 [28]. The details of pipeline implementation are explained in the next sections.

4.1.1 Reads mapping and differentially gene expression analysis

This section explains the reads mapping and differentially gene expression analysis using two different methods; in the first method, reads mapping through STAR was performed and then the data was analyzed through EdgeR, while in the second method, reads mapping was performed through Kallisto and then the data was analyzed through Sleuth.

4.1.1.1 STAR and edgeR

The pipeline implementation uses STAR aligner software to carry out the read alignment of RNASeq time series data. The STAR aligner's input consists of the reference genome and the annotation file that need to be downloaded or accessed from public databases using tools such as biomaRt [23]. STAR alignment consists of two steps: 1) generating genome indexing from target genome annotation files and 2) alignment of reads in FASTAQ files using genome indexing information. After mapping the FASTAQ file to the reference genome, STAR stores the result of the alignment file in a SAM or BAM format [42] in the pipeline-specified directories. FeatureCounts software [43] summarizes the reads counts mapped from the RNA sequencing files in the previous task. Next, the workflow uses an edgeR pipeline for differential gene expression analysis. The RNA -seq pipeline uses various statistical methodologies available in this package for carrying out filtering, normalization, multidimensional plotting, identification of genes with significant expression changes, clustering, etc. A typical processing in edgeR includes the following steps: 1) loading the data as an R dataframe and then combine into a matrix where each cell in a matrix

represents the count of the number of the reads for a particular gene; 2) scaling and normalization of counts per million (CPM) and Log counts per million (Log-CPM) data; 3) filtering out the genes that are not expressed; 4) removing any data artifacts or batch effect in the using robust normalization of the data (edgeR trimmed mean of M-value (TMM) method normalization); 5) creating a DGEList object to hold the read counts that remains after the filtering; 6) plotting multidimensional scaling plots to perform the analysis the inter-sample relationships; 7) setting up the model by taking the DGEList object and the groups and formed the design matrix; 8) using voom to perform additional data normalization to remove batch effect and other processing biases; 9) forming the contrast matrix for the analysis of the comparison of interest; 10) fitting the model data - this step is carried out using Limma function lmfit and contrast.fit for contrast matrix; empirical Bayes moderation can be applied to contrast matrix; 11) using treat method of Treat function provided by Limma package [57] to set log-FC value requirements. The *treat* method allows to set decisions based on p-value results from the empirical Bayes moderator t-statistics and the target log-FC threshold; and 12) examining differentially expressed data partitioned into upregulated and downregulated genes and storing them in ".csv" files for each of comparison; generating the mean difference plot to study the upregulated and downregulated genes.

4.1.1.2 Kallisto and Sleuth

Kallisto is a command-line tool used for quantifying transcript abundances by pseudo-alignment. An R script was designed to pseudo-align, using Kallisto, batches of trimmed mRNA reads (according to the sample processing configuration file), and test, using Sleuth, for differentially expressed genes. The pipeline script uses as input a reference genome annotation file and the mRNA read

datasets. Three other input files are also necessary: one file containing an edited design matrix, another with arguments to the pipeline, and the third indicating sample comparisons to make when testing potential differentially expressed genes. Genome annotation files can be imputed either through the biomaRt R package or provided as input after download from a public database. One downside to the biomaRt gene extraction method is that some annotations might not be up to date (i.e., for A. thaliana currently, only the TAIR10 annotation is available, and not Araport11 [13]). One advantage of using biomaRt is that annotations are available for each gene so that descriptions can be returned in the differential expression analysis alongside each gene's statistics. To perform differential gene expression analysis, Sleuth is programmed to iterate over the comparison sets specified in the DGECM file. Sleuth provides both transcripts- and gene-level differential gene expression analysis functionality. A Boolean flag is written in the arguments to specify the analysis level. The code captures each of three cases: transcript-level, gene-level, or both. Gene-level analysis requires an additional dataset with transcript identifiers mapped to corresponding gene names. Alternatively, Kallisto count data can be analyzed in edgeR after a data transformation involving gene isoform removal.

4.1.2 Network inference

This section explains the different methods used for inferring gene regulatory and gene correlation networks.The WGCNA package was used and performed analysis on the differentially expressed genes obtained from the previous step to obtain a gene correlation network. The TF(Transcriptional Factor) gene interaction data was obtained from DREM to obtain a gene regulatory network and performed the DREM analysis on the DEG genes.

4.1.2.1 Gene Correlation Networks Analysis

NetSeekR implements statistical network analysis using WGCNA, including network construction, module detection, and topological overlap matrix construction of the gene expression data. WGCNA is an R software package used to infer gene networks from transcriptomics data by applying topological constraints derived from complex networks' statistical analysis. It includes additional methods for integration of phenotype or traits data with the gene expression data. In WGCNA, highly correlated gene clusters called modules are identified using a hierarchical clustering technique. Module eigengenes was also computed which is the first principle component representing a given module's gene expression profile, and assessed the proportion of variance among the genes in a particular module. In the NetSeekR pipeline, a data preprocessing step was performed consisting of filtering of lowly expressed genes from the output of Sleuth/edgeR. The genes with an excessive number of missing samples was also filtered out to carry out further analysis. Then one can visualize the clustering of the samples using an Euclidean distance measure. The gene correlation network is specified by the adjacency matrix obtained from a similarity matrix that measures the similarity between the gene expression profiles across all the samples. The similarity matrix contains the absolute value of the correlation between series of expression data for each pair of genes in the dataset. These series for large-scale experimental designs was constructed to span time points, treatments, conditions, genotypes, etc. Pearson correlation was used to calculate the similarity of pairs of genes in the dataset. An adjacency function transforms the similarity matrix containing co-expression similarities into the adjacency matrix containing connection strengths. A typical choice is biweight midcorrelation (bicor), as it is more robust and less susceptible to outliers while capturing monotonic relations between genes. Soft thresholding measure was used to set

the adjacency matrix power. The implementation uses the *pickSoftthreshold* function to analyze network topology to select the appropriate soft threshold. Next, the Topological Overlap Matrix (TOM), a measure of connectivity strength between the two nodes based on common neighbors' overlap, is computed. After calculating the overlap matrix, the next step is module detection.

The Modules are the clusters of closely interconnected nodes, which are genes with high topological overlaps obtained using the unsupervised clustering technique provided by the WGCNA package. The pipeline performs average linkage hierarchical clustering using the dissimilarity matrix obtained from Topological Overlap Matrix. A cluster dendrogram of all the genes was obtained using the R function *hclust*.In the cluster dendrogram, the modules are detected as branches, and dynamic tree cut is used to perform module identification. A further step is the reduction of the number of modules that resulted after clustering. First the co-expression similarity of modules was quantified by calculating the eigengenes and clustering them based on their correlation. Then the modules with a similar expression profile were merged. For module display, the TOM plot was used, which summarizes the co-expression network, showing the dissimilarity matrix's values.

4.1.2.2 Gene Regulatory Networks (GRN) analysis

To infer GRNs, count data was used from the read mapping pipelines as input for DREM. First, a dataset rearrangement was performed that ensures the data series structure associates with the GRN analysis design (typically series span time points and/or treatments). Networks are graphed for differential expression gene sets to show overlap between an input network (default is the DREM input network) and the differentially expressed genes; mean read counts per gene

are also represented. Count data from the pseudo-alignment performed by STAR/Kallisto is used to produce DREM input data. If needed, reads for multiple gene models are aggregated together by summing count data for each gene. Aggregated gene quantification sets are saved to unique files based on sample identifiers in a data directory (from the directory tree). A data transformation is implemented on the edited quantification data (aggregate read counts) such that read counts are arranged in a time series format for each condition and each replicate. The time series rearrangement results are stored in uniquely named files in a directory to be accessed by DREM.

DREM is executed for each dataset corresponding to a time series for a genotype, condition, and technical replicate. Time series datasets discussed prior are accessed iteratively for analysis; the argument defaults file provided by DREM2.0 (defaults.txt) is edited in the R code by inserting paths to the time series datasets proper cells, accounting for technical replicates. All arguments to DREM other than those used for iterating over time-series datasets need to be specified in the default template file prior to running the pipeline. NetSeekR calls an instance of the DREM GUI for each genotype, condition, and technical replicate dataset. Files generated for DREM input should be used in the DREM GUI. Gene tables for each path should be saved, once DREM finishes executing, to the DREM analysis-specific directory with a literal 'path' string to be extracted in network analysis.

4.1.3 Gene Ontology Enrichment Analysis

After identifying the WGCNA modules, the GO enrichment analysis was performed for each of the modules obtained in the analysis from DEG sets. The aim is to identify significantly

overrepresented gene categories and biological processes associated with the treatment performed in experimental design. Since the GO package included with WGCNA (anRichment [20]) does not provide support for many genomes (including plant genomes), the package topGO [4] was additionally added to perform this analysis. Using topGO, overrepresentation of GO terms was tested.

4.1.4 Network analysis

Further network analysis is provided using the igraph R package. One network analysis method consists of searching for overlapping nodes between differential gene expression sets and an existing gene network from public data sources. For example, such a network file is provided by DREM for six organisms. Other networks can be obtained from public databases such as: GeneMania [72], IntAct [32], STRING [66] and KEGG [30]. Further, count data derived from Kallisto or STAR results can be used to visualize average counts per gene with regard to the replicates belonging to each differential gene expression comparison set. Beta coefficients from Sleuth or log-fold changes from edgeR can also be visualized per gene in each extracted network. The processing of input files proceeds with reducing multiple sources and multiple targets (gene identifiers) to generate unique edges. Initially, edges in the network file may contain multiple unique and non-unique gene nodes. The reduction ensures edges are unique, leaving one edge per row in the dataset. DEG gene sets are then overlapped with nodes in the provided network, generating sets of differentially expressed nodes in each comparison set. Read counts of DEG genes are represented over each DEG network, and per gene, statistics are computed. Overlapping nodes are visualized in graphs where positive or negative beta coefficients are represented as distinct shapes, estimated counts

are represented as size, and beta coefficients (scaled to a range from 0 to 1) are represented by shape opaqueness. When using input from Sleuth, beta coefficients for multiple gene isoforms are averaged; alternatively, Sleuth analysis can be performed for genes instead of gene models. EdgeR analysis produces log-fold-change information that can be visualized of the target networks using this package.

Metrics can be collected to characterize network structures and to provide insight into key pathways associated with the experimental treatment. Node degree, centrality, shortest pathways, diameter, clustering coefficient, etc., can be calculated on differential gene expression graphs. To illustrate these capabilities, a method to generate networks of differentially expressed genes have been implemented to identify the hub genes of each comparison set. The node centrality score was calculated for all the clusters and identified the hub genes. The hub genes were obtained from each cluster to find top genes from the network. The top genes list was then intersected with the most significant GO enrichment genes calculated using Kolmogorov-Smirnov testing in the topGO package. To visualize network information, The module information obtained from WGCNA (cluster membership information of each gene) were incorporated and combined it with the nodes(genes) data of the network data frame, assigning each node the color of the module to which that particular gene belonged to. The functions implemented by NetSeekR are briefly described in table 4.1 on page 82.

4.2 Results and Discussion

NetSeekR allows integrated management of the RNASeq tools, comparative analysis of gene expression implemented using multiple pipelines, and network analysis. It can provide support

for sample comparison using PCA and MDS analyses, comparative analysis of DEG gene sets using UpSetR [25], visualization and analysis of gene correlation networks, and visualization and analysis of gene regulatory networks. In figure 4.2 on page 86 [28] visualization capabilities related to network analysis were enumerated for an experimental design that includes time-series data of four selected time points, two stress factors, and two genotypes. Comparisons on statistics of gene expression estimates between a spliced read aligner and a pseudo-aligner can be used to understand the dynamics of gene expression better. The PCA and MDS plots show the sample correlation and its dynamics as a function of treatment and genotype as calculated with STAR-edgeR (SE) and Kallisto-Sleuth (KS) pipelines, respectively (a sample PCA plot is shown in figure 4.2 on page 86.A) [28]. The upset analysis allows DEG comparison between the SE and KS pipelines (figure 4.2 on page 86.B) [28].

Next, the construction of correlation networks was performed to analyze the dynamics of differentially expressed genes and correlate this dynamic with observed phenotypes. Parametrization of WGCNA controls the scale-free topology constraints that shape the structure of the correlation network. Also, network constraints derived from publicly available interactome data can be superimposed to the resulting topology using network analysis software.

A network analysis function implemented in NetSeekR builds network structures from the correlation of differentially expressed genes calculated in WGCNA.

figure 4.2 on page 86.C shows the network constructed for differentially expressed genes for two WGCNA modules (defined by the respective colors) with node sizes proportional to their degree.

The package includes a regulatory network analysis function as well. In this function, results from the WGCNA and DREM analysis were combined to build regulatory bipartite graphs of

significant Transcriptional Factor (TF) - gene interactions in the analyzed data. The modules obtained in the previous step in the WGCNA analysis were taken as input along with the lists of significant genes for each comparison set we use from the EdgeR/Sleuth output files. The significant genes were filtered out based on the p-value. TFs from AtTFDB from the Arabidopsis Gene Regulatory Information Server (AGRIS) [26] database were identified. AtTFDB contains information on approximately 1,770 transcription factors (TFs) grouped into 50 families based on conserved domains. This output was then aaded to the DREM output containing the list of regulatory TFs in the data. This data was then converted into an incident matrix which was then use to build a bipartite graph where one set is the list of TFs and the other the list of genes that those TFs are regulating (figure 4.2 on page 86.D).

The NetSeekR pipeline also provides integration with several GO analysis R packages that can be used for assessing overexpression of differentially expressed genes, clusters/modules of genes with correlated gene expression, or performing GO analysis of other combinations of gene sets obtained from the pipeline.

4.3 Conclusion

NetSeekR, a new integrated pipeline for large-scale experimental designs that include RNASeq time-series observations of gene expression dynamics of multiple treatments and multiple genotypes was designed. The pipeline vertically integrates several reads mapping and analysis tools with regulatory and correlation network tools and provides additional network analysis, performance analysis, and network visualization. The methodology takes advantage of the increasing availability of efficient data analysis pipelines to generate a flexible integration of genomics analysis

tools from reads mapping to gene network analysis. This integration allows the rapid design of gene expression analyses, easy comparison of several pipelines (using reads alignment and pseudo-alignment), differential gene expression analysis, network analysis, facilitating genomics discovery from large-scale NGS data. The pipeline provides network prediction and analysis capabilities by integrating the inference of regulatory and correlation networks with network structure analysis and visualization tools. In this way, the pipeline bridges genomics data analysis results to systems biology modeling and simulation.

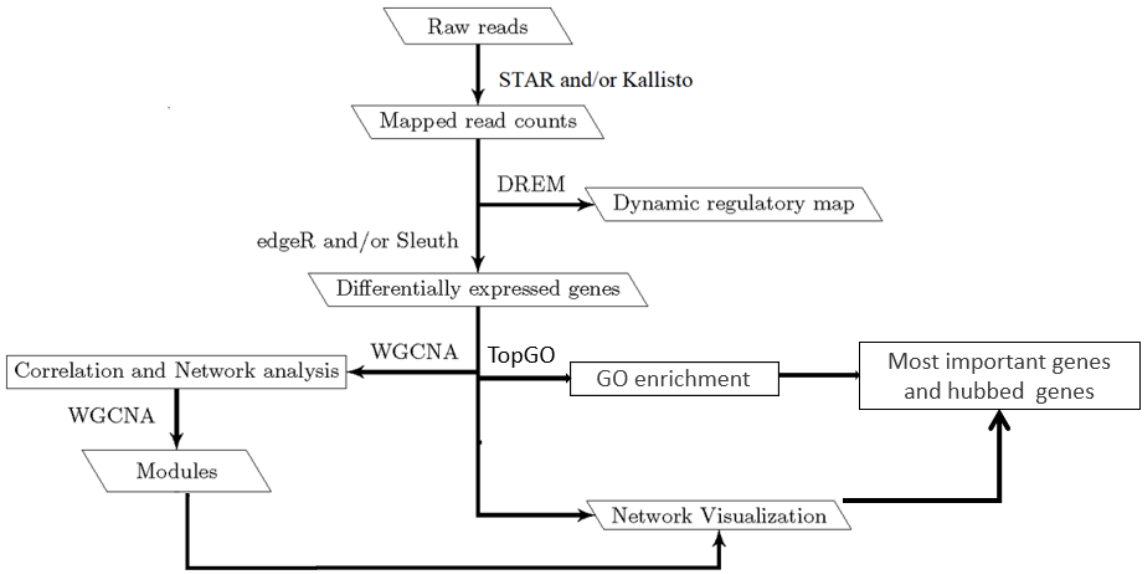

Figure 4.1

Workflow of the NetSeekR pipeline

Table 4.1

Brief description of functions implemented in NetSeekR

Function_name	Description
implement_alignment	Extract all arguments from the configuration file, decide which pipeline to run, assemble arguments for each decided pipeline, and implement each decided pipeline.
extract_pipeline_input_from_configuration	Convert arguments from configuration file into a named list data structure.
decide_alignment_tool	Determine which pipeline(s) to execute and write directory trees for each tool chosen.
write_kallisto_directory_tree	Create directory tree for Kallisto results.
write_STAR_directory_tree	Create directory tree for STAR results.
assemble_alignment_arguments	Subset arguments particular to an alignment tool and ensure proper command formatting to execute.
align_tool_bool	Check for tool existence in the decision data structure.
subset_tool_arguments	Subset tool arguments from the decision data structure and pass them to tool-specific command processing functions.
implement_kallisto	Write bash scripts for Kallisto index building and quantification, then execute them.
implement_STAR	Write bash scripts for STAR index building and mapping, then execute them.
implement_feature_counts	Write bash script for feature counts to execute.
implement_differential_gene_expression	Extract alignment function output and execute differential gene expression testing depending on the upstream alignment or pseudo-alignment software used.
split_and_unlist_conditions	Separate covariates from list of conditions.
write_differential_testing_file_name	Combine conditions from each sample the comparison set to create a file name with.
implement_sleuth	Run Sleuth on one sample the comparison set at a time.
structure_covariates_for_DGE_testing	Searches for conditions in the comparison sets which vary in a column

Table 4.1

(continued)

Function_name	Description
separate_conditions	Separate condition column into multiple columns based on the covariates.
select_sleuth_mode	Determine at what level to run Sleuth: gene, transcript, or both.
gene_level_analysis_data_structure	Removes splicing notation from one quantification file result to use for gene-level Sleuth analysis.
extract_single_testing_condition	Select a single covariate to test with by identifying the condition column with varying contents across samples.
transcript_prep	Sleuth_prep implementation was specific to transcript analysis.
gene_prep	A Sleuth_prep implementation specific to gene analysis.
run_sleuth_pca	Plot PCA plots for a comparison.
save_sleuth_pca	Save PCA plots for a comparison.
run_sleuth_fit	Measurement error model fitting with Sleuth.
run_sleuth_lrt	Perform likelihood ratio test.
extract_sleuth_lrt_results	Extract Likelihood Ratio test results from a Sleuth object.
save_sleuth_results	Save LRT results to a file for a comparison set.
implement_edgeR	Conduct differential testing on STAR mapped reads with edgeR.
filter_counts_data	Filter the lowly expressed genes for edgeR analysis.
edgeR_preliminary	Process input data for Edge R analysis followed by the analysis and set the directory where data has to be saved.
implement_network_analysis	Performs the network visualization and finds out hub genes in the network using Go enrichment analysis also.
wgcna_input_data	Takes the input data and convert it into differentially expressed expression data for WGCNA analysis.

Table 4.1

(continued)

Function_name	Description
counts_keep	Load the count data into the environment using decisions from which count data type is being sourced.
filter_counts_data	Filter the lowly expressed genes from the data obtained from feature counts.
reset_splice_variants	Remove splice variant notation.
wgcna_data_processing	Preprocess the input data for WGCNA.
differentially_expressed_genes_keep	Find the count data of differentially expressed genes from TSV files of edgeR/Sleuth results regardless of which tool was used.
extract_comparison	Find the comparison file name and remove .tsv string pattern
read_and_filter	Reads the data from Sleuth/edgeR and filters out the differentially expressed genes.
wgcna_plot_sample_tree	Plot to find any outlier sample in the data for all the comparisons.
hclust_distance_matrix	Find the distance matrix to perform clustering
plot_sample_tree	Plot to find any outlier sample in the data.
wgcna_plot_power_results	Histogram plot to analyze and choose correct power value for all the comparisons.
plot_power_results	Histogram plot to analyze and choose correct power value.
wgcna_plot_power_histogram	Plot to choose the correct power from scale free topology for all the comparisons.
plot_power_histogram	Plot to choose the correct power from scale free topology.
wgcna_clustering	Perform clustering and saves result in their respective directory.
clustering	Perform clustering and produces gene dendrogram on the data.
implement_GO_enrichment	Checks the DEG data on which GO analysis to be performed and performs GO analysis.
enrichment	Test and performs GO analysis for significantly enriched genes in the deg gene sets.

Table 4.1

(continued)

Function_name	Description
extract_filename_go	Extracts the comparison set for which GO enrichment and network analysis to be performed.
post_process_GO_results	Associate file names with GO results.
DREM_main	Extract required data for executing DREM.
load_read_data	Load count data.
rearrange_count_data	Create data sets formatted for DREM.
write_DREM_time_series_data	Write DREM-formatted data sets to files.
write_default_files	Create DREM configuration file and write data.
insert_DREM_input_to_defaults	Create default files to execute DREM for samples.
write_new_default_file_name	Create a file name to write DREM defaults script.
mapping_network_analysis	Apply network analysis to each comparison dataset.
network_analysis	Finds the network of differentially expressed genes and collect the hub genes of each comparison.

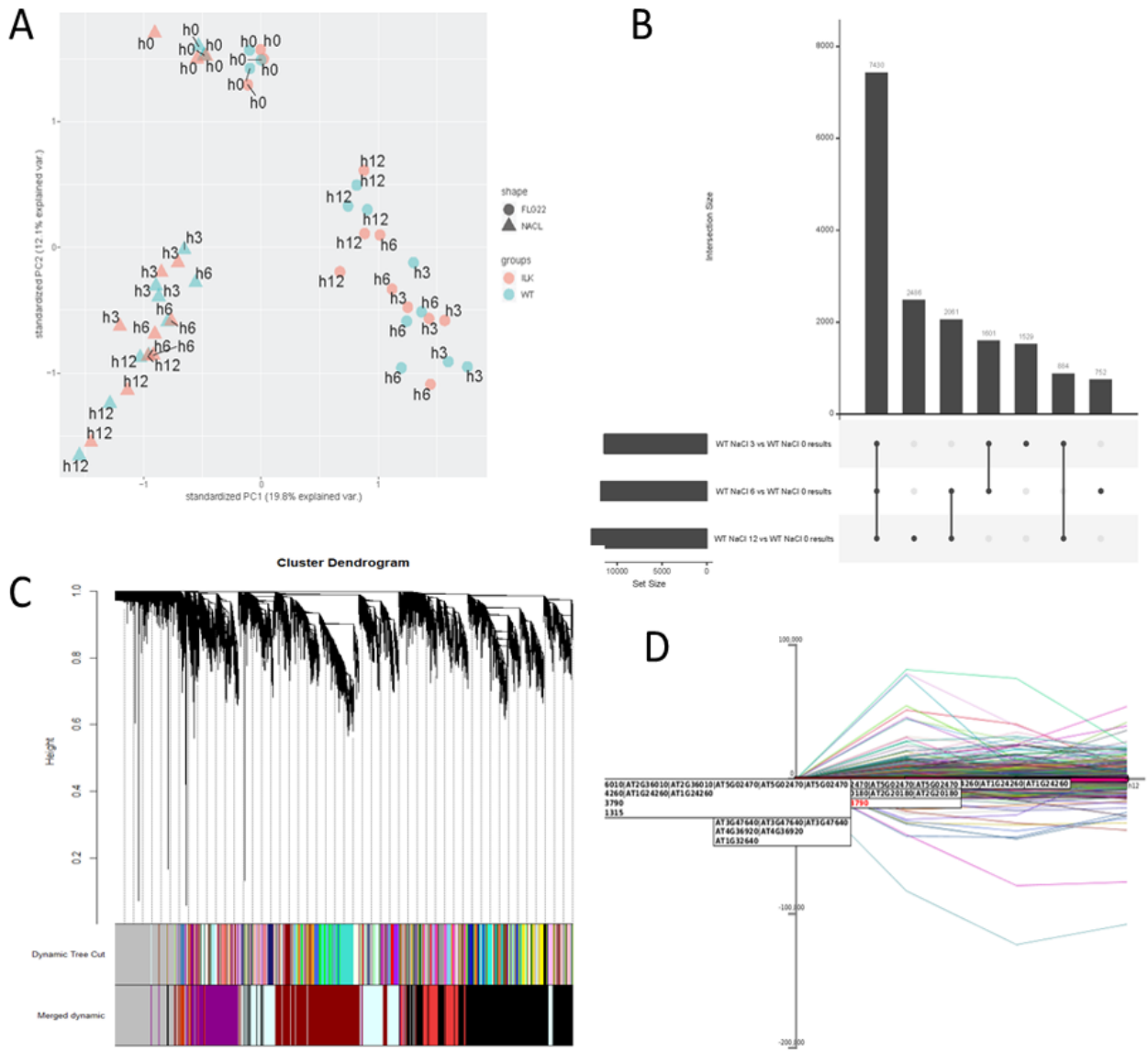

Figure 4.2

A) PCA plot from Sleuth analysis of time-series of RNASeq data; B) UpSetR comparison of time series of RNASeq data; C) Dendrogram from WGCNA analysis of time series of RNASeq data; D) DREM analysis of time series of RNASeq data.

CHAPTER V

METHODS FOR INFERENCE IN GENE REGULATORY NETWORK USING LOOPY BELIEF PROPAGATION ALGORITHM

5.1 Introduction

Probabilistic graphical modeling is a statistical modeling technique that captures the conditional independence and dependency between random variables in the graph. It describes a probabilistic model using a graph structure. Graphical manipulations can be used to express the complex computations that require inference and learning [71]. One of the significant applications of probabilistic graphical modeling is its use in image denoising and image segmentation. This chapter proposes a probabilistic graphical model for gene regulatory networks. The modeling technique proposed here involves the representation of the data in the form of a graph. In this chapter, an undirected factor graph is constructed out of a gene regulatory network with the data obtained from the public database. The interaction parameters used in the graph were obtained from the correlation measures and read counts for the given set of genes. The loopy belief algorithm was used to perform inference on the graphical model/graph. In the method used in the chapter, the belief message-passing algorithm was used using the sum-product algorithm to determine each variable node's marginal probabilities/beliefs in a graph. The main task is to compute efficiently marginal distribution of a single variable when all discrete other variables' joint distribution or energy function is already known. To accomplish the task, the Sum Product algorithm of the

loopy belief propagation method was used. The gene regulatory network that was used here is a Transcriptional-factor and gene interaction model. The data structure used in the graphical modeling in this chapter was a factor graph. A factor graph is a bipartite probabilistic model with two nodes: a factor node and a variable node. In this work, the variable nodes are all the genes (TF and regulated genes), whereas the factor nodes are all the interaction parameters between a gene and a Transcription factor. In this model, genes are considered variables, and the factors are energy functions obtained from the correlations and counts gene expression data taken as the factor node. A further description of implementing PGM and drawing inference on the graph using a loopy belief algorithm is given in the next section.

5.2 Probabilistic graphical modeling

In a graphical model, a graph is denoted as $G = (V, E)$ consists of set of nodes V and set of edges E. Nodes are associated with the random variables so for each node i ($i \in V$), let x_i denote the random variables where $i = 1, 2, 3....n$ [37]. The edges between the pair of nodes determine the probabilistic interaction between the two nodes. There are two types of graphical modeling Directed Graphical Models (DGMs), or Bayesian Networks (BNs) and Undirected Graphical Models (UGMs) or Markov Random Fields (MRFs) [37].

5.2.1 Bayesian Network representation

In the Bayesian network, the random variable x_i has an associated conditional probability distribution associated with it. The representation of the Bayesian network is based on directed cyclic or acyclic graph representation. A directed edge from a random variable x_i to another random variable x_j represents the conditional probability relation between the random variables [37]. The

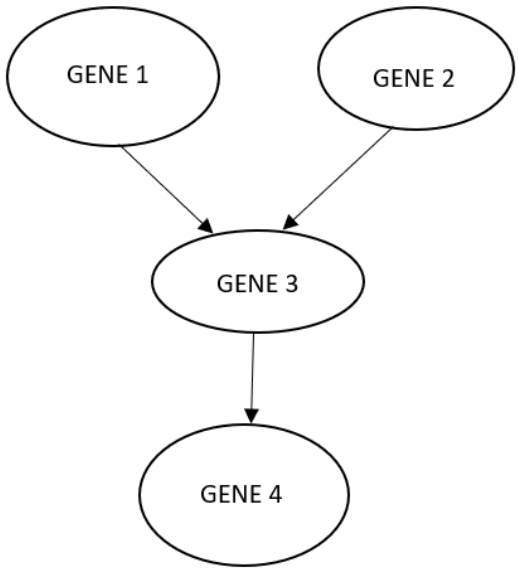

Figure 5.1

Bayesian Network

concept of causality defines the relationship between two random variables as one random variable can be the cause of the second random variable. The variable that causes is called the parent node, whereas the variable caused by any parent variable is called the child node. The Bayesian network defines the independency rule such that the probability of the child node is in one of its states depends directly on the state of its parent node. The figure 5.1 shown below represents an example of how Bayesian network can be used in the causality method in gene networks. In the figure, for the nodes which do not have any parent node, let's define the probability as $p(G_1)$ and $p(G_2)$ for GENE 1 and GENE 2, respectively. For nodes that are caused by their respective parent node are given conditional probability as $p(G_3|G_2, G_1)$

The over-all joint probability distribution for N such random variable can be defined as

$$p_b(x_1....x_N) = \prod p(x_i|Par(x_i)) \tag{5.1}$$

where $Par(x_i)$ is the probability of the parent nodes.

5.2.2 Markov Random Fields and Loopy belief Propagation

Markov random field or Markov net is a category of undirected graphical models because the edges are undirected in a Markov random field holds the node independence rule, which is given as

$$\forall i \in V, X_i \perp X_{V-i}|X_{N_i} \tag{5.2}$$

here $N_i = j\{i, j\} \in \varepsilon$ denotes the set of neighbor of node i in the graph G and $X_i \perp X_j|X_k$ means that for given X_k ,X_i and X_j are independent [37].

To parameterize the structure in the Markov network, factors are used. The parameterization in the Markov network is used to integrate the graph nodes and edges with a set of parameters. It is is done in the same way as associating conditional probability distribution in the Bayesian network; but, the parameterization in the Markov network is not as intuitive as in the Bayesian network. Factors in Markov Network may or may not correspond either to probabilities or conditional probabilities.

5.2.3 Factor graph

In a Markov Random Field, to represent conditional dependency or independence between two variables, a probability distribution parameterization method called factors is used to compute

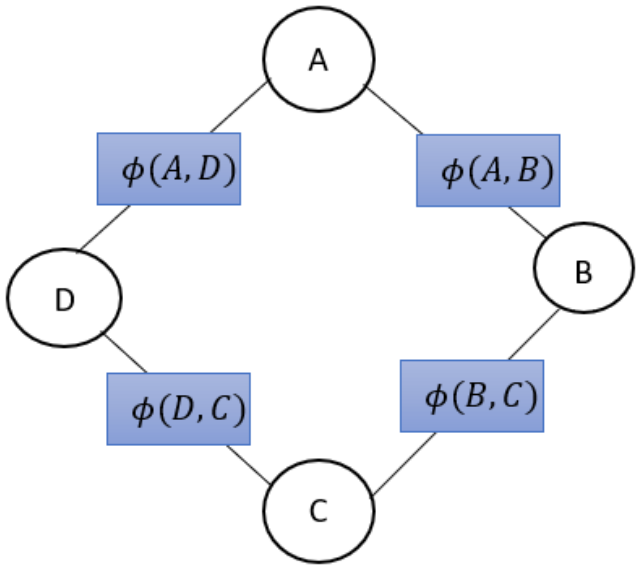

Figure 5.2

Factor graph

marginals probabilities. So to know the probability distribution function of all variables in a probabilistic graphical model, one needs to use the technique to model the property in dependent variable and their interaction parameters.

The factor graph represents a graphical model containing the variable node, the factor node. Factor node usually determines the interaction between a pair of variable nodes or a set of nodes by assigning a score to the joint probability distribution between the nodes.

For example, in a factor graph, given in figure 5.2 below, the representation of joint probability between any two nodes, say B and D, can be done by assigning a score to the interaction between these two nodes is $\phi(B, D)$. The interaction between any two nodes or a set of nodes is called factors.

Similarly any other pair of nodes of the same graph can be taken and find the similar score or factor, $\phi(A, B)$, $\phi(A, C)$,$\phi(D, C)$ and etc. and then defining the probability as normalized score(these scores could be any function) and take it as factor products:

$$\tilde{p}(A, B, C, D) = \phi(A, B)\phi(A, C)\phi(D, C)\phi(B, C) \tag{5.3}$$

The final probability is then defined as

$$p(A, B, C, D) = \frac{1}{Z}\tilde{p}(A, B, C, D) \tag{5.4}$$

where $Z = \sum_{A,B,C,D} \tilde{p}(A, B, C, D)$ is a normalizing constant that ensures that the distribution sums to one.

5.2.3.1 Gibbs Distribution in a Factor graph

One way of understanding factorization technique much further is by defining the energy parameters for all the possible assignment in the given space and then using the normalization factor, this energy parameter is converted into probability functions. In some cases one could use the maximal clique potential technique to find the factor graph.

Consider a joint probability distribution $p(x_i|y)$ where $x_i = \{x_1, x_2, \ldots, x_n\}$

$$p(x_i|y) = \sum_{x_1} \ldots \sum_{x_{n-1}} \sum_{x_n} p(x|y) \tag{5.5}$$

for all discrete variable. So for variables $x_1, ..., x_n$ in an undirected graph G, where nodes are denoted as x_i. According to Clifford theorem, the probability $p(x_i)$ over all the variables of x_i could be converted in the form of cliques C in Graph G. C has the form $\phi_c(x_c)$ which denotes the energy function of cliques. Z is the normalization factor.

$$Z = \sum_{x_1 \ldots x_n} \prod_{c \in C} \phi_{c(x_c)} \tag{5.6}$$

$\phi_c(x_c)$ denotes the clique's potential function, which is a real-valued function.

5.2.4 Pairwise Markov Random field

This chapter introduces the concept of pairwise Markov Random fields. The transformation of such a network into a cluster graph using the Gibbs sampling method is fairly straightforward. Here, only the pairwise interaction is only taken into account. The energy function is calculated for the pairwise interaction instead of using a clique of several nodes and calculating the energy of those cliques. In the pairwise Markov random field, the concept of the Gibbs sampling method makes sure that the interaction between a pair of nodes is uniform across all the edges of the graph. The interaction between a pair of nodes is converted into an energy function, which is later converted into an exponential function to complete the uniform distribution requirement. So the energy of interaction is a function of a univariate potential/energy over each variable node x_i and a pairwise potential over the pair of variables nodes that form an edge. Later the same principle of the Gibbs sampling method is used over all the energy function calculated. In This chapter, the interaction parameter between gene and Transcription Factor was defined using energy parameters or cliques for the graph G. The gene-TF probabilistic model transformed into the energy functions between two variables, a gene, and a TF. This kind of implementation is a form of Markov Random field where the interaction parameter in the form of an exponential energy function transforms into a factor between two variables. The equation for finding the energy of interaction between a pair of nodes and then converting it into a form that follows the Gibbs sampling method is discussed

in the next section. The final normalized interaction constraint between a pair of nodes can be considered similar to joint probability distribution between a pair of nodes, giving us the idea of the probability of interaction between a TF and gene node given the gene node.

5.3 Modelling gene regulatory network using factor graph

This section explains the modeling technique using Markov Random Fields and loopy belief propagation to find the gene's differential expression using gene expression data and a gene regulatory network. The gene expression data were obtained for flg22 treatment from feature counts. The Agris database [16] [48] was used to find the TFs that regulate the genes. The energy function has three parts; the first part corresponds to the univariate potential for each gene node x_i. To find the univariate potential, the probability of occurrence were found first for all the genes and TFs based on the expression value of both the genes and the TFs. P_g corresponds to probability score of a gene, P_{TF} corresponds to the Probability of TF. n_g is the initial state of the expression value of a gene with probability P_g. n_{TF} is the initial state of the expression value of a TF with probability P_{TF}. P_T is the threshold z score that the user could set. p_T is used to make decision for the state of ng and n_{TF}.It is taken from the prior knowledge based the up regulated genes, down regulated genes and their corresponding probabilities.

$$1 - (P_g - 0.5)n_g \tag{5.7}$$

The decision on initial state of n_g and n_{TF} is taken as:

$$state(n_g) = \begin{cases} n_g = 1, & \text{if } P_g > P_T \\ n_g = -1, & \text{if } p_g < P_T \\ n_g = 0, & \text{otherwise} \end{cases}$$

The second part of the energy function corresponds to the interaction between the gene node and the TF node. The correlations measure were taken between the normalized expression value of gene and the normalized expression value of a TF to find the interaction with their states multiplied. R defines how the TF acts on the target gene; whether it inhibits the gene, then it is -1 or activates the genes, then it is +1. The term, $corr(n_g * x_g, n_{TF} * x_{TF})$ is obtained using Pearson correlation. The second part of the energy function is given as

$$1 - ||corr(n_g * x_g, n_{TF} * R * x_{TF})|| \tag{5.8}$$

The third and last part of energy function is the penalty term used to penalize the energy function if it becomes too high and is dependent on state of gene,n_g. The third part of energy function uses $max((n_g)^2, K)$ as the term to penalize of the state of n_g used. K is constant kept at 0.05 for the modeling done in this chapter.

The final energy function is given as:

$$E(x) = 1 - (P_g - 0.5)n_g + 1 - ||corr(n_g * x_g, n_{TF} * R * x_{TF})|| + max((n_g)^2, K) \tag{5.9}$$

This energy parameter is converted to form of a joint probability distribution by using the following equation

$$p(x) = exp(-E(x)) \tag{5.10}$$

The above probability distribution function is used in the factor graph to set the factor potential between each pair of respective nodes. The following section described the mechanism to perform the message passing in the factor graph.

5.4 Belief propagation in Gene-TF factor graphs

The graph was constructed using the igraph package. A factor data structure was formed using the python class inheritance method to hold the probability density function data in the Factor graph. The variable nodes were initialized with the parameter augmented from state. The energy parameter between each gene pair was calculated using the equation 5.9. The energy parameter was then converted to probability distribution between the pair of genes using the equation 5.10. After obtaining the factor graph and initializing the graph with the initial set of parameters, the next task is to propagate and receive the belief messages from one to all other nodes. The figure 5.3 on the following page shows the message passing algorithm in Factor graph. In the figure the factor node is marked as black nodes and gene node are marked as white nodes.

let x_i denote the variable node and $Ne(x_i)$ denote the neighbors of variable node x_i ($i = 1, 2, 3....n$ for n number of variable nodes). Assuming f_j denote the factor node and $Ne(f_j)$ denotes the neighbors of factor nodes ($j = 1, 2, 3....n$ for n number of factor nodes). To understand the message passing algorithm, the message from a given variable node to a given factor node on the edges (denoted by E) is defined as $\mu_{x_i \to f_j \in E}$ and from a given factor node to a given variable node on the edges E as $\mu_{f_j \to x_i \in E}$. The message passing in this chapter is bi-directional.Message passing from gene node to transcriptional factor and from transcriptional factor to genes as a feedback loop. The message passing uses sum product algorithm.

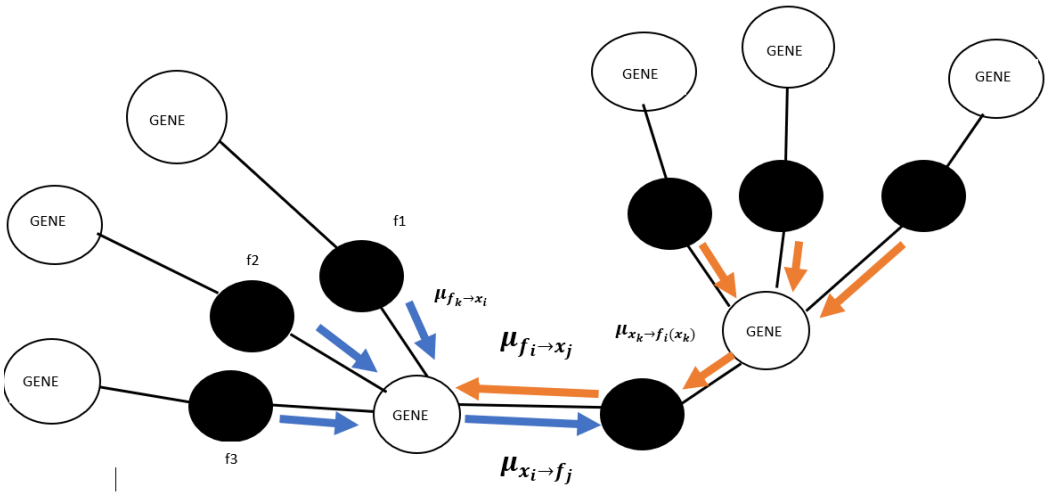

Figure 5.3

Message passing in gene regulatory network

The first step of message passing is from the TF node to the gene node. The message is first passed from the TF variable node to the factor node; it will follow the product rule. So, the product of the message from all the neighboring factor nodes except the recipient factor node is taken. The message passing is according to the equation (5.11).

$$\mu_{x_i \to f_j} = \prod_{f_k \in Ne(x_i)/f_j} \mu_{f_k \to x_i} \tag{5.11}$$

The second phase of the message passing involves passing message from factor node to variable gene node. In this part the sum rule is used. In this case, the product of the factors were taken with the messages from all the other node except the variable node where the message has to be sent and then take the summation of all of that neighboring node which doesn't include the recipient variable node. The equation for this message passing is given in equation (5.12).

$$\mu_{f_i \to x_j} = \sum_{Ne(f_i)/x_j} (f_i(X) \prod_{x_k \in Ne(f_i)/x_j} \mu_{x_k \to f_i(x_k)}) \tag{5.12}$$

Considering only working in the pairwise interaction model, so the summation term is of no use as there will be only one factor for each pairwise interaction between a gene and a TF. So the equation used will be given as in equation (5.13).

$$\mu_{f_i \to x_j} = f_i(X) \prod_{x_k \in Ne(f_i)/x_j} \mu_{x_k \to f_i(x_k)} \tag{5.13}$$

One have to check whether each node has passed the message and also received the message. so the message passing technique After the forward message passing is performed, the reverse directional message passing is also checked to give the feedback message to each gene node. After all the nodes have received the message (after passing through the number of iterations), One can calculate the marginal probability distribution or beliefs at each variable gene node. This can be done by simply normalizing the message received at each node and then calculating the beliefs at each node. The normalized message or final belief message will be compared to the threshold again, and the initial state *ng* will be set accordingly. This message passing will continue until the difference localized sum for beliefs for a subgraph in the entire network is too small.

After the multiple variables have received the message (after passing through number of iterations), one can calculate the beliefs at each variable node by the formula given below:

$$b_{x_i}(x_i) = \prod_{j \in Ne(x_i)} \mu_{j \to i}(x_i) \tag{5.14}$$

The belief messages $b_{x_i}(x_i)$ is then normalized $b_{x_i}(x_i)$ for all the variable nodes i.

5.4.1 Loopy belief propagation implementation

After obtaining the graph, the next step was to perform inference to identify the differential expression of the genes in the dataset. The algorithm passing has three main steps: message

initialization, message passing, message update.The initialization step uses the initial values of the initialization obtained $p(x)$ for factor nodes and *state(n_g)* for variable nodes. The message passing step is conducted in the forwarding direction (from TF nodes to gene nodes) and feedback direction (from gene nodes to the TF nodes). The message passing takes place in the form of dictionaries in python. Each node is assigned an old dictionary variable m_{old} and a new dictionary m_{new}. These dictionaries are assigned a couple of key values: Start node and End Node. The start node and end node specify the direction in which the message has to be passed. The old dictionary stores the value of previous iterations, while the new dictionary stores the value at the current iteration. The values in the dictionary keep on updating at each iteration. At initialization the old dictionaries of all the nodes are initialized with the values at initialization while the new dictionaries are initialized as zero. The description for the entire algorithm is given below.

1. Initialize the message at each node:
   (a) Initialize m_{old} for Factor node using $p(x)$ and m_{old} for variable node using state of genes/nodes.
   (b) Initialize m_{new} for each node (variable node and factor node) as zero.
2. Forward message passing algorithm and update:
   (a) Update the m_{new} of nodes where TF is the start node, and the corresponding factor nodes are the end nodes. This is done by passing the message of the variable TF node to the neighboring factor node using the product rule of the sum-product algorithm given in equation (5.11) and using the values in m_{old} dictionaries for the corresponding nodes used in message passing.
   (b) Update the m_{new} of nodes where factor node is the start node, and the corresponding target gene nodes are the end nodes. This is done by passing the message from the factor nodes to the corresponding target gene node using the sum rule of the sum-product algorithm given in equation (5.13) and using the values in m_{old} dictionaries for the corresponding nodes used in message passing.
   (c) Check for normalized belief for the message at each target node if it is connected to other subsequent nodes; otherwise, if the target gene is a sink node, then the message passed in the second step will only become the belief message for the target gene. The belief at each target node is calculated using the given equation (5.14).

The two step forward message passing example from the TF node to Factor node and then from Factor node to the target gene node is shown in figure 5.4 on page 103.

3. Feedback message passing algorithm and update:

   (a) Change the state n_g for target genes based on the message received by the target gene in the second step of the forward message passing and compare it with the threshold.

   (b) Update the m_{new} of nodes where the target node is the start node, and the factor node is the end node by changing the factor node's potentials using the equation (5.10). This is done because of the probability of interaction between gene and TF changes with the change of state of the target gene. The message passed from the target gene node to the corresponding neighboring factor nodes using the target gene state as a feedback process.

   (c) Update the m_{new} of nodes where the factor node is the start node, and the TF node is the end node based on the message received by the TF node as feedback. This is done by taking the normalized product of all the messages received from the factor node using the equation (5.14).

   (d) Change the state n_{TF} for target genes based on the message received by the TF gene in the previous step and compare it with the threshold.

   (e) For the next iteration, initialize the m_{old} dictionaries of all the nodes with the values obtained in the m_{new} dictionaries of the corresponding nodes. The message passing continues until the beliefs have a very low difference in subsequent iterations.

   (f) To check for convergence, the difference of the sum of the belief message received at all the variable nodes at iteration t and $t + 1$. If the difference between the sum of the belief messages is too small, The state at which the convergence takes place would be the final state for the differential expression of the genes for the particular subgraph of the network.

The two step feedback message passing example from the Target node to Factor node and then from Factor node to the TF gene node is shown in figure 5.5 on page 104.

5.5 Results

The figure figure 5.6 on page 105shows a TF- gene interaction Markov network. This network is obtained by using the probabilistic graphical modeling approach. The graph/network obtained below is called the factor graph. To construct the graph, I borrowed some of the visualization packages and functions from the repository [39]. The white circled nodes are the variable gene

nodes that represent the genes or TF or both. The black circled nodes represent the factor nodes that signify the joint probability distribution or the interaction parameter between each pair of TF gene interaction nodes. After obtaining the graph, the data structure of the variable node was loaded with the initial state and the factor node with the data structure that holds the interaction parameters for each pair of nodes. The graph was then divided into several subgraphs. For each subgraph, the corresponding start node and end node were checked to ensure correct message passing using the igraph package. The algorithm for smaller subgraphs was run, and a level of convergence could be reached by checking the local variables for convergence. However, the global parameter such as the threshold probability value and the minimum convergence value needed to be appropriately tuned to reach convergence. This is a great difficulty in this algorithm. However, the message passing is sequentially, but the parallel processing for all the subgraphs simultaneously could also be done to check for convergence with local variables. The parameter tuning must be checked across all the subgraphs, and the global parameters need to be set at a particular variable. It aids in the convergence of algorithms for all the subgraphs. The final state of the gene must be compared to that obtained from the differential gene analysis done in the previous chapters.

5.6 Conclusion

This chapter used a data-driven approach to find local minima for the convergence at each TF node. The algorithm requires a proper energy optimization technique to find the local minima for each TFs. The biggest challenge in setting up the energy equation is finding whether the energy parameters converge to a global minimum. Interestingly, the TF that acts upon a gene can also become a gene and be controlled by a different TF; there were several loops obtain in the

network. Many other energy optimization methods could be used to find the global minima for the belief message passing. Even though the algorithm proposed in the chapter works on finding a local minima for each TF node, it could be used to find the expression of a set of the gene. The differential expression of these sets of genes could be used for further downstream analysis.

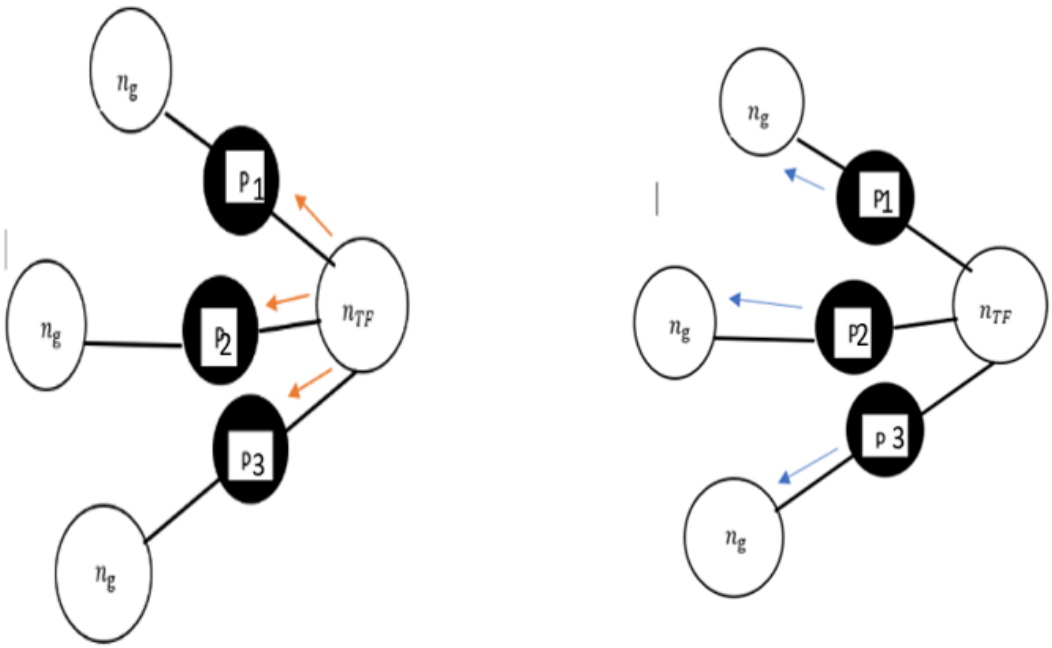

Figure 5.4

Forward Message Passing: (a) Forward Message passing from TF node to Factor node using product rule of Sum Product Algorithm. (b) Forward message passing from Factor node to target gene node using sum rule of Sum Product Algorithm.

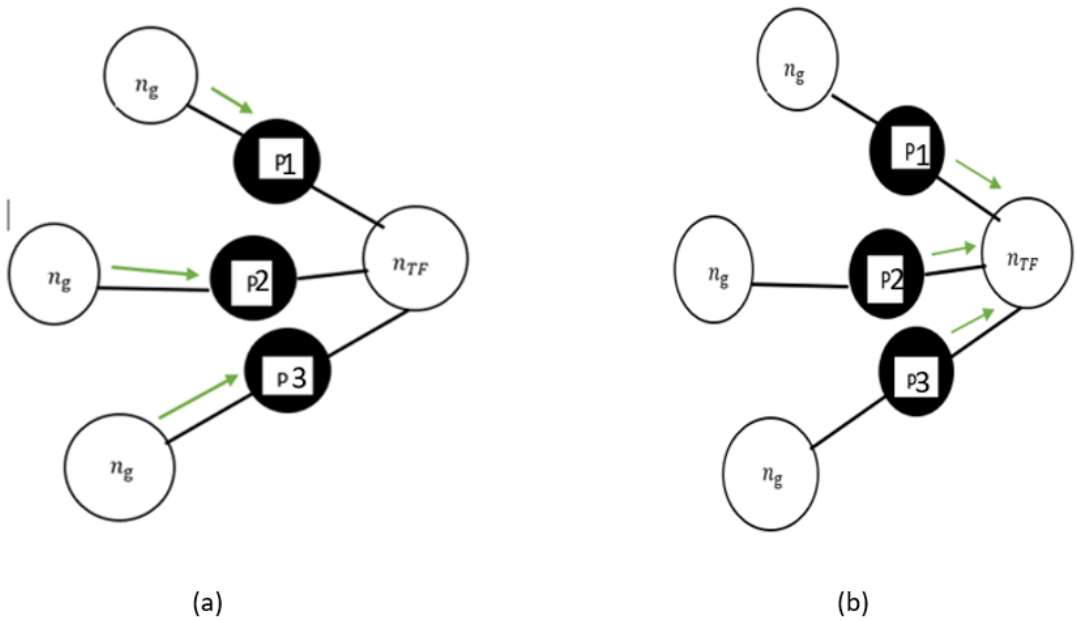

Figure 5.5

Feedback Message Passing: (a) Feedback Message passing from Target node to Factor node is calculated using the energy update. (b) Feedback message passing from Factor node to TF gene node is computed using belief message normalization

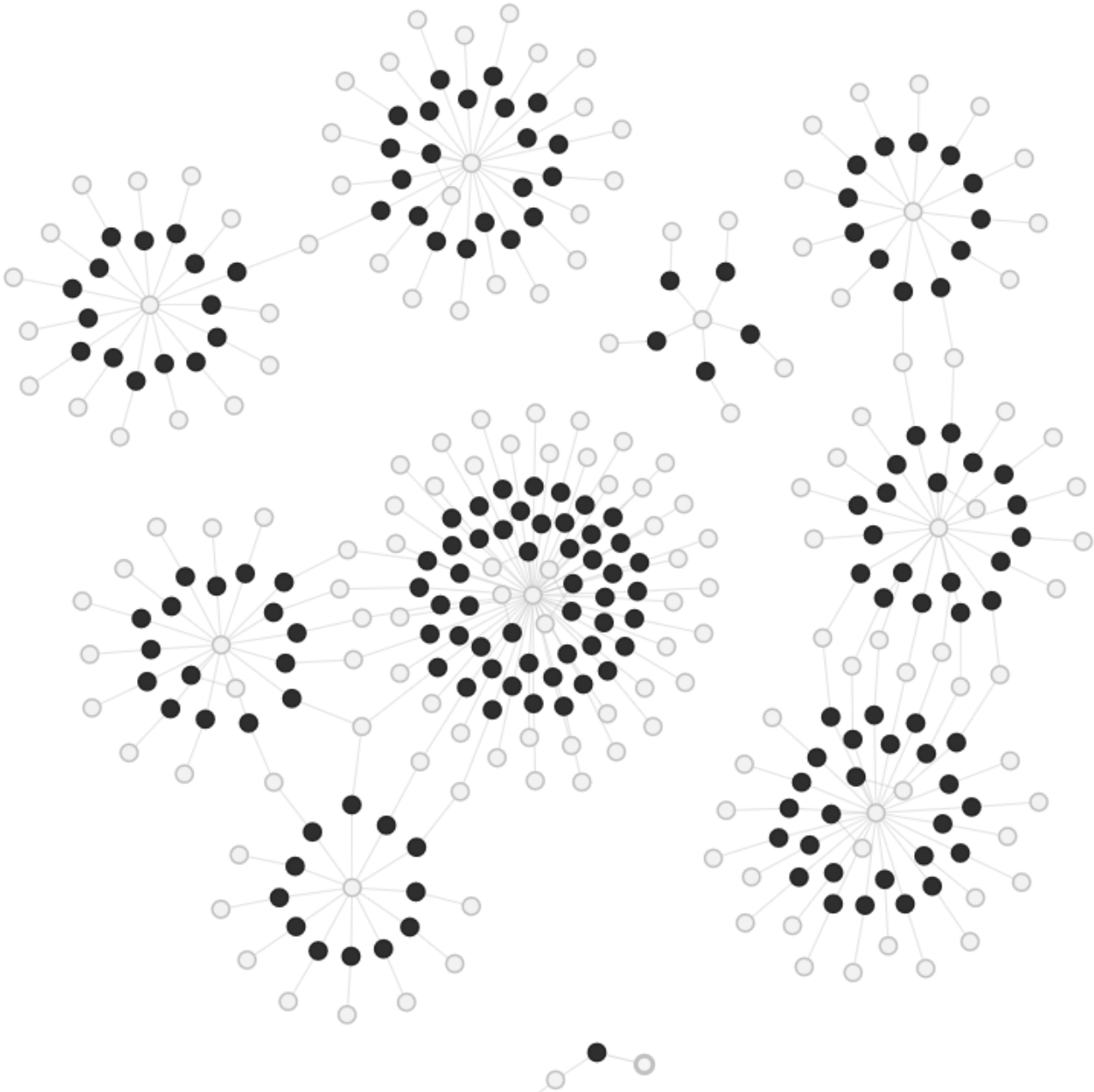

Figure 5.6

Sample Markov random field obtained for Arabidopsis samples treated with flg22 treatment

CHAPTER VI

CONCLUSIONS

6.1 Summary

This dissertation uses advances in bioinformatics tools used in RNA sequencing technology as a practical methodology for mining and analyzing genomics data. We also used machine learning and probabilistic graph modeling algorithms to study gene networks and draw inferences from the data. This dissertation studied Arabidopsis plants treated with bacterial flagellin and NaCl treatment at different time points and found out the effect of biotic stress and abiotic stress on the plant ILK1 and Wild-type mutant Arabidopsis plant. In this work, RNA sequencing technology was used to find the differentially expressed genes from the given samples of Arabidopsis plant subjected to different stress treatments/experiments at a given time frame. Network analysis was used to study the correlation pattern, detect clusters, and use the network topological approach to extract essential information from the differentially expressed genes. Probabilistic graphical modeling and belief propagation algorithm-based inference method was used to determine the genes' differential expression using gene regulatory network as input. We also designed and implemented an R software pipeline that contains the capacity to carry out the mining and analysis of differentially expressed genes into one software package.

In Chapter 2, mapping the reads from RNA seq data using the software STAR aligner was performed, and then the data was further processed and quantified using FeatureCounts. Finally,

R packages Limma and EdgeR were used to perform the differential analysis on the RNA- seq data to obtain and identify the differentially expressed genes. The data were normalized using Voom normalization. The results were obtained for ten different kinds of comparison analysis, each for Flg22 and Nacl separately. The comparison analysis included genotypic comparison analysis and phenotypic comparison analysis. Each comparison sets analysis was tested for the required log-FC value threshold. Linear model fitting was performed for the comparison set of interest using linear modeling in limma. Next, the empirical Bayes moderation test was carried out to give a sense of differentially expressed genes in each different set of comparisons set. Fil the DEGs obtained were stored in CSV files to conduct the next level of downstream analysis.

In chapter 3, correlation patterns amongst the differentially expressed genes (obtained from chapter 2) were studied in the stress study by constructing gene co-expression networks. We used the R software package WCGNA to analyze the correlation pattern. We integrated the networks analyzed with the multiple bioinformatics tools and studied the resulting gene network's topology using the R package igraph. The WGCNA package helped detect modules for each set of differentially expressed genes obtained from the previous chapters. The modules or clusters of genes obtained for each comparison sets were then tested for their functional enrichment using various public databases. There were significant results obtained for flg22 treatment. The correlation values obtained from the WGCNA were used to construct a correlation network. The correlation network helped find some of the hubbed genes based on different network topological analysis tools like centrality scores. The functional gene enrichment tools and databases were also used to find the biological functionality of the essential set of genes obtained from the network's topological

analysis. Besides, we used another gene prediction tool, DREM, to identify transcription factors that control gene expression dynamics in time series data.

To compile the tools and packages mentioned above, we developed a software package that consists of a programmatic framework using R software and Linux shell commands. The idea was to implement the data mining and data analysis steps mentioned above into a single software package. We called this R package, NetSeekR as the main objective is to infer and analyze gene networks from the RNA-Seq time series data. Chapter 4 describes the structure and functionality of NetSeekR. The main functionalities of this package include alignment of processed reads to genome positions using the gene annotation file, identification of differentially expressed genes, predicting gene networks, module detection of differentially expressed genes, gene ontology enrichment analysis, and topological network analysis of differentially expressed genes.

The final chapter, Chapter 5 of this dissertation, was dedicated to identifying the gene expression value of genes using gene regulatory networks. This task was performed using a belief propagation algorithm on a probabilistic graphical model. This algorithm uses the gene expression dataset obtained from the previous chapter and a literature-based public database to obtain information about TF-gene interactions as input to draw the probabilistic graph. The inference algorithm is based on the energy minimization method, which would require further improvement. The graph we used was a sparse network of around 70000 edges.

From work mentioned in the dissertation, we found ILK1 expression causes widespread defects in the transcriptional program when activated by the bacterial elicitor flg22. We found some of the target genes that were associated with cell wall integrity and immunity. The significant contribution of the work in this dissertation is given in the next section.

6.2 Contributions

The major contribution of the research performed in the dissertation is given below:

1. From the data mining and analysis of DEGs, several genes responsible for several biological processes were obtained. The analysis implied that genes obtained were linked to cell wall biosynthesis and modification, encoding anchored membrane proteins for Arabidopsis Thaliana. Some genes found were related to immune responses, and their functioning is found in Arabidopsis Thaliana. Some microtubule-associated genes were obtained in the output had defective regulation in the ilk1-1 line for Arabidopsis Thaliana. All these genes were obtained for Arabidopsis Thaliana post-treatment with flg22. Signaling components and response genes mediated by defense hormones abscisic acid (ABA), jasmonic acid (JA), and salicylic acid (SA) and the growth hormone auxin were overrepresented among ILK1-1 DEGs and included a majority of genes with up-regulated expression levels at 0 and 12 h post-treatment with flg22.

2. With NetseekR, a novel gene expression data analysis pipeline was designed by compiling all the components used for sequencing and analyzing RNA sequencing data. The pipeline can perform alignment of reads, carry out differential gene expression analysis, perform gene ontology enrichment analysis, and network analysis of differentially expressed genes. The pipeline also holds the capacity to allow the user to chose different tools for aligning the reads and carrying out differential gene expression analysis of the genes by including two different options for each step. The pipeline has features for direct execution and manual execution if the user wants to change the parameters set by default.

3. A loopy belief propagation algorithm for biological network inference method was designed. Implementation of belief propagation algorithm was performed on a graphical model designed for transcriptomics analysis using gene expression data and TF-gene regulatory network information. Using the network inference algorithm proposed in the dissertation, the differential expression of genes was inferred. Although the algorithm requires further development, it can be used to analyze the biological significance of genes expression changes in the context of available TF regulatory network information without using downstream analysis of genes regulatory pathways could be used to understand the biological process associated with the gene transcriptional reprograming.

6.3 Further Research

The analysis carried out in this research assumes the counts' data to be distributed according to the negative binomial distribution. Other discrete distributions could also be used to find the differentially expressed genes. In this dissertation, correlation networks and gene-TF regulatory

networks have been investigated using WGCNA and igraph. Several other network analysis tools are available on Bioconductor packages that could provide several other topological analysis tools to investigate the interaction pattern in the data. Also, a Similar network analysis could be used for the Protein-protein interaction for Arabidopsis and other plants from the network inference algorithm.

The pipeline NetSeekr also combines the software Packages that assume the count distribution to have a negative binomial distribution. To make the pipeline versatile for all kinds of data, one could combine other software packages that work on other discrete data distributions.

The inference model used in the research works on minimizing the distance between the counts of genes and thus minimizing the energy parameter to obtain higher energy of interaction for the gene and TF. One could find other energy parameters based on TF- gene interaction binding sites and could use it as a parameter to find the interaction between gene and TF. The inference model provides the differential expression of genes and TFs. These differentially expressed genes and TFs could further be used for other downstream analyses, as mentioned in chapter 3, to find the critical set of genes and TFs and infer a particular biological process related to the specific treatment.

REFERENCES

[1] R. Agrahari, A. Foroushani, T. R. Docking, L. Chang, G. Duns, M. Hudoba, A. Karsan, and H. Zare, "Applications of Bayesian network models in predicting types of hematological malignancies," *Scientific Reports*, vol. 8, no. 1, 2018, p. 6951.

[2] E. M. Airoldi, "Getting Started in Probabilistic Graphical Models," *PLOS Computational Biology*, vol. 3, no. 12, 2007, pp. 1–5.

[3] A. Ajroud, M. Omri, H. Youssef, and S. Benferhat, "Loopy Belief Propagation in Bayesian Networks : origin and possibilistic perspectives," *ArXiv*, vol. abs/1206.0976, 2012.

[4] A. Alexa and J. Rahnenfuhrer, *topGO: Enrichment Analysis for Gene Ontology*, 2018, R package version 2.34.0.

[5] S. Anders and W. Huber, "Differential expression analysis for sequence count data," *Genome Biology*, vol. 11, no. 10, 2010, p. R106.

[6] S. Anders, P. T. Pyl, and W. Huber, "HTSeq—a Python framework to work with high-throughput sequencing data," *Bioinformatics*, vol. 31, no. 2, 2014, pp. 166–169.

[7] S. Andrews et al., "FastQC: a quality control tool for high throughput sequence data,", 2010.

[8] A. Anjum, S. Jaggi, E. Varghese, S. Lall, A. Bhowmik, and A. Rai, "Identification of Differentially Expressed Genes in RNA-seq Data of Arabidopsis thaliana: A Compound Distribution Approach," *Journal of computational biology : a journal of computational molecular cell biology*, vol. 23, no. 4, apr 2016, pp. 239–247.

[9] E. U. Azeloglu and R. Iyengar, "Signaling networks: information flow, computation, and decision making.," *Cold Spring Harbor perspectives in biology*, vol. 7, no. 4, apr 2015, p. a005934.

[10] A. M. Bolger, M. Lohse, and B. Usadel, "Trimmomatic: a flexible trimmer for Illumina sequence data," *Bioinformatics*, vol. 30, no. 15, aug 2014, pp. 2114–2120.

[11] S. P. Borgatti, "Centrality and network flow," *Social Networks*, vol. 27, no. 1, 2005, pp. 55–71.

[12] Y. Chen, A. T. L. Lun, and G. K. Smyth, "From reads to genes to pathways: differential expression analysis of RNA-Seq experiments using Rsubread and the edgeR quasi-likelihood pipeline [version 2; peer review: 5 approved]," *F1000Research*, vol. 5, no. 1438, 2016.

[13] C.-Y. Cheng, V. Krishnakumar, A. P. Chan, F. Thibaud-Nissen, S. Schobel, and C. D. Town, "Araport11: a complete reannotation of the Arabidopsis thaliana reference genome," *The Plant Journal*, vol. 89, no. 4, Feb. 2017, pp. 789–804.

[14] A. Conesa, P. Madrigal, S. Tarazona, D. Gomez-Cabrero, A. Cervera, A. McPherson, M. W. Szcześniak, D. J. Gaffney, L. L. Elo, X. Zhang, and A. Mortazavi, "A survey of best practices for RNA-seq data analysis," *Genome Biology*, vol. 17, no. 1, 2016, p. 13.

[15] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal*, vol. Complex Systems, 2006, p. 1695.

[16] R. V. Davuluri, H. Sun, S. K. Palaniswamy, N. Matthews, C. Molina, M. Kurtz, and E. Grotewold, "AGRIS: Arabidopsis Gene Regulatory Information Server, an information resource of Arabidopsis cis-regulatory elements and transcription factors," *BMC Bioinformatics*, vol. 4, no. 1, 2003, p. 25.

[17] J. Díaz and E. R. Alvarez-Buylla, "Information flow during gene activation by signaling molecules: ethylene transduction in Arabidopsis cells as a study system," *BMC Systems Biology*, vol. 3, no. 1, 2009, p. 48.

[18] M. V. DiLeo, G. D. Strahan, M. den Bakker, and O. A. Hoekenga, "Weighted Correlation Network Analysis (WGCNA) Applied to the Tomato Fruit Metabolome," *PLOS ONE*, vol. 6, no. 10, 2011, pp. 1–10.

[19] M.-A. Dillies, A. Rau, J. Aubert, C. Hennequet-Antier, M. Jeanmougin, N. Servant, C. Keime, G. Marot, D. Castel, J. Estelle, G. Guernec, B. Jagla, L. Jouneau, D. Laloë, C. Le Gall, B. Schaëffer, S. Le Crom, M. Guedj, and o. b. o. T. F. S. C. Jaffrézic Florence, "A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis," *Briefings in Bioinformatics*, vol. 14, no. 6, 2012, pp. 671–683.

[20] H. S. P. B. D. F. P. G. V. P. S. C. Dimlioglu, G., "Distinct contributions from Integrin-Linked Kinases to plant cell wall integrity and innate immunity," Unpublished paper, 2020.

[21] A. Dobin, C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T. R. Gingeras, "STAR: ultrafast universal RNA-seq aligner.," *Bioinformatics (Oxford, England)*, vol. 29, no. 1, jan 2013, pp. 15–21.

[22] A. Dobin and T. R. Gingeras, "Mapping RNA-seq Reads with STAR.," *Current protocols in bioinformatics*, vol. 51, sep 2015, pp. 11.14.1–11.14.19.

[23] S. Durinck, P. T. Spellman, E. Birney, and W. Huber, "Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt.," *Nature protocols*, vol. 4, no. 8, 2009, pp. 1184–1191.

[24] N. Friedman, "Inferring Cellular Networks Using Probabilistic Graphical Models," *Science*, vol. 303, no. 5659, 2004, pp. 799–805.

[25] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, K. Hornik, T. Hothorn, W. Huber, S. Iacus, R. Irizarry, F. Leisch, C. Li, M. Maechler, A. J. Rossini, G. Sawitzki, C. Smith, G. Smyth, L. Tierney, J. Y. Yang, and J. Zhang, "Bioconductor: open software development for computational biology and bioinformatics.," *Genome biology*, vol. 5, no. 10, 2004, p. R80.

[26] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, jun 2002, pp. 7821–7826.

[27] B. C. Haynes, E. J. Maier, M. H. Kramer, P. I. Wang, H. Brown, and M. R. Brent, "Mapping functional transcription factor networks from gene expression data.," *Genome research*, vol. 23, no. 8, aug 2013, pp. 1319–1328.

[28] D. F. Himangi Srivastava and G. V. Popescu, "NetSeekR: A network analysis pipeline for RNASeq time series data," Unpublished paper, 2020.

[29] D. W. Huang, B. T. Sherman, and R. A. Lempicki, "Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources.," *Nature protocols*, vol. 4, no. 1, 2009, pp. 44–57.

[30] M. Kanehisa, M. Araki, S. Goto, M. Hattori, M. Hirakawa, M. Itoh, T. Katayama, S. Kawashima, S. Okuda, T. Tokimatsu, and Y. Yamanishi, "KEGG for linking genomes to life and the environment.," *Nucleic acids research*, vol. 36, no. Database issue, jan 2008, pp. D480–4.

[31] A. Kanterakis, G. Potamias, and G. P. Patrinos, "Chapter 4 - An Introduction to Tools, Databases, and Practical Guidelines for NGS Data Analysis," *Human Genome Informatics*, C. G. Lambert, D. J. Baker, and G. P. Patrinos, eds., Academic Press, 2018, pp. 61–89.

[32] S. Kerrien, B. Aranda, L. Breuza, A. Bridge, F. Broackes-Carter, C. Chen, M. Duesbury, M. Dumousseau, M. Feuermann, U. Hinz, C. Jandrasits, R. C. Jimenez, J. Khadake, U. Mahadevan, P. Masson, I. Pedruzzi, E. Pfeiffenberger, P. Porras, A. Raghunath, B. Roechert, S. Orchard, and H. Hermjakob, "The IntAct molecular interaction database in 2012," *Nucleic acids research*, vol. 40, no. Database issue, jan 2012, pp. D841–D846.

[33] A. Khattak, *Natural Variation in Arabidopsis thaliana Growth in Response to Ambient Temperatures*, doctoral dissertation, jun 2014.

[34] D. Kim, B. Langmead, and S. L. Salzberg, "HISAT: a fast spliced aligner with low memory requirements," *Nature Methods*, vol. 12, no. 4, 2015, pp. 357–360.

[35] D. Kim, G. Pertea, C. Trapnell, H. Pimentel, R. Kelley, and S. L. Salzberg, "TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions," *Genome Biology*, vol. 14, no. 4, 2013, p. R36.

[36] A. Kirkley, G. T. Cantwell, and M. E. J. Newman, "Belief propagation for networks with loops," *Science Advances*, vol. 7, no. 17, 2021.

[37] D. Koller, N. Friedman, L. Getoor, and B. Taskar, "Graphical Models in a Nutshell,", 2007.

[38] M. Koornneef and D. Meinke, "The development of Arabidopsis as a model plant," *The Plant Journal*, vol. 61, no. 6, 2010, pp. 909–921.

[39] A. Krasikov, "Belief-Propagation,", https://github.com/krashkov/Belief-Propagation, 2019.

[40] P. Lamesch, T. Z. Berardini, D. Li, D. Swarbreck, C. Wilks, R. Sasidharan, R. Muller, K. Dreher, D. L. Alexander, M. Garcia-Hernandez, A. S. Karthikeyan, C. H. Lee, W. D. Nelson, L. Ploetz, S. Singh, A. Wensel, and E. Huala, "The Arabidopsis Information Resource (TAIR): improved gene annotation and new tools.," *Nucleic acids research*, vol. 40, no. Database issue, jan 2012, pp. D1202–10.

[41] P. Langfelder and S. Horvath, "WGCNA: an R package for weighted correlation network analysis," *BMC Bioinformatics*, vol. 9, no. 1, 2008, p. 559.

[42] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, and . G. P. D. P. Subgroup, "The Sequence Alignment/Map format and SAMtools," *Bioinformatics*, vol. 25, no. 16, aug 2009, pp. 2078–2079.

[43] Y. Liao, G. K. Smyth, and W. Shi, "featureCounts: an efficient general purpose program for assigning sequence reads to genomic features.," *Bioinformatics (Oxford, England)*, vol. 30, no. 7, apr 2014, pp. 923–930.

[44] D. J. McCarthy, Y. Chen, and G. K. Smyth, "Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation," *Nucleic Acids Research*, vol. 40, no. 10, 2012, pp. 4288–4297.

[45] H. Mi, A. Muruganujan, D. Ebert, X. Huang, and P. D. Thomas, "PANTHER version 14: more genomes, a new PANTHER GO-slim and improvements in enrichment analysis tools.," *Nucleic acids research*, vol. 47, no. D1, jan 2019, pp. D419–D426.

[46] K. Murphy, Y. Weiss, and M. I. Jordan, "Loopy Belief Propagation for Approximate Inference: An Empirical Study," *UAI*, 1999.

[47] Y. Ni, P. Müller, L. Wei, and Y. Ji, "Bayesian graphical models for computational network biology," *BMC Bioinformatics*, vol. 19, no. 3, 2018, p. 63.

[48] S. K. Palaniswamy, S. James, H. Sun, R. S. Lamb, R. V. Davuluri, and E. Grotewold, "AGRIS and AtRegNet. A Platform to Link cis-Regulatory Elements and Transcription Factors into Regulatory Networks," *Plant Physiology*, vol. 140, no. 3, 2006, pp. 818–829.

[49] T. L. Pedersen, *tidygraph: A Tidy API for Graph Manipulation*, 2019, R package version 1.1.2.

[50] G. Pei, L. Chen, and W. Zhang, "Chapter Nine - WGCNA Application to Proteomic and Metabolomic Data Analysis," *Proteomics in Biology, Part A*, A. K. Shukla, ed., vol. 585 of *Methods in Enzymology*, Academic Press, 2017, pp. 135–158.

[51] M. Pertea, D. Kim, G. M. Pertea, J. T. Leek, and S. L. Salzberg, "Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown," *Nature Protocols*, vol. 11, no. 9, 2016, pp. 1650–1667.

[52] H. Pimentel, N. L. Bray, S. Puente, P. Melsted, and L. Pachter, "Differential analysis of RNA-seq incorporating quantification uncertainty," *Nature Methods*, vol. 14, no. 7, 2017, pp. 687–690.

[53] S. C. Popescu, E. K. Brauer, G. Dimlioglu, and G. V. Popescu, "Insights into the Structure, Function, and Ion-Mediated Signaling Pathways Transduced by Plant Integrin-Linked Kinases," *Frontiers in Plant Science*, vol. 8, 2017, p. 376.

[54] H. S. R. Rajula, M. Mauri, and V. Fanos, "Scale-free networks in metabolomics.," *Bioinformation*, vol. 14, no. 3, 2018, pp. 140–144.

[55] F. Rapaport, R. Khanin, Y. Liang, M. Pirun, A. Krek, P. Zumbo, C. E. Mason, N. D. Socci, and D. Betel, "Comprehensive evaluation of differential gene expression analysis methods for RNA-seq data.," *Genome biology*, vol. 14, no. 9, 2013, p. R95.

[56] P. Richterich, "Estimation of errors in "raw" DNA sequences: a validation study," *Genome research*, vol. 8, no. 3, mar 1998, pp. 251–259.

[57] M. E. Ritchie, B. Phipson, D. Wu, Y. Hu, C. W. Law, W. Shi, and G. K. Smyth, "limma powers differential expression analyses for RNA-sequencing and microarray studies," *Nucleic Acids Research*, vol. 43, no. 7, 2015, pp. e47–e47.

[58] M. D. Robinson, D. J. McCarthy, and G. K. Smyth, "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data," *Bioinformatics*, vol. 26, no. 1, 2009, pp. 139–140.

[59] M. D. Robinson and A. Oshlack, "A scaling normalization method for differential expression analysis of RNA-seq data," *Genome Biology*, vol. 11, no. 3, 2010, p. R25.

[60] G. Sabidussi, "The centrality index of a graph," *Psychometrika*, vol. 31, no. 4, 1966, pp. 581–603.

[61] M. H. Schulz, W. E. Devanny, A. Gitter, S. Zhong, J. Ernst, and Z. Bar-Joseph, "DREM 2.0: Improved reconstruction of dynamic regulatory networks from time-series expression data," *BMC systems biology*, vol. 6, aug 2012, p. 104.

[62] S. C. Schuster, "Next-generation sequencing transforms today's biology," *Nature Methods*, vol. 5, no. 1, 2008, pp. 16–18.

[63] R. Stark, M. Grzelak, and J. Hadfield, "RNA sequencing: the teenage years," *Nature Reviews Genetics*, vol. 20, no. 11, 2019, pp. 631–656.

[64] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov, "Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles," *Proceedings of the National Academy of Sciences*, vol. 102, no. 43, 2005, pp. 15545–15550.

[65] K. K. H. Svoboda and W. R. Reenstra, "Approaches to studying cellular signaling: a primer for morphologists," *The Anatomical record*, vol. 269, no. 2, apr 2002, pp. 123–139.

[66] D. Szklarczyk, A. Franceschini, M. Kuhn, M. Simonovic, A. Roth, P. Minguez, T. Doerks, M. Stark, J. Muller, P. Bork, L. J. Jensen, and C. von Mering, "The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored.," *Nucleic acids research*, vol. 39, no. Database issue, jan 2011, pp. D561–8.

[67] C. Trapnell, D. G. Hendrickson, M. Sauvageau, L. Goff, J. L. Rinn, and L. Pachter, "Differential analysis of gene regulation at transcript resolution with RNA-seq.," *Nature biotechnology*, vol. 31, no. 1, jan 2013, pp. 46–53.

[68] C. Trapnell, A. Roberts, L. Goff, G. Pertea, D. Kim, D. R. Kelley, H. Pimentel, S. L. Salzberg, J. L. Rinn, and L. Pachter, "Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks.," *Nature protocols*, vol. 7, no. 3, mar 2012, pp. 562–578.

[69] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, and L. Pachter, "Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation.," *Nature biotechnology*, vol. 28, no. 5, may 2010, pp. 511–515.

[70] P. Veenstra, C. Cooper, and S. Phelps, "The use of Biweight Mid Correlation to improve graph based portfolio construction," *2016 8th Computer Science and Electronic Engineering (CEEC)*, 2016, pp. 101–106.

[71] C. Wang, N. Komodakis, and N. Paragios, "Markov Random Field modeling, inference & learning in computer vision & image understanding: A survey," *Computer Vision and Image Understanding*, vol. 117, no. 11, 2013, pp. 1610–1627.

[72] D. Warde-Farley, S. L. Donaldson, O. Comes, K. Zuberi, R. Badrawi, P. Chao, M. Franz, C. Grouios, F. Kazi, C. T. Lopes, A. Maitland, S. Mostafavi, J. Montojo, Q. Shao, G. Wright, G. D. Bader, and Q. Morris, "The GeneMANIA prediction server: biological network integration for gene prioritization and predicting gene function.," *Nucleic acids research*, vol. 38, no. Web Server issue, jul 2010, pp. W214–20.

[73] J. Yedidia, W. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," 2003.

[74] L. Yin, C.-H. Huang, and J. Ni, "Clustering of gene expression data: performance and similarity analysis.," *BMC bioinformatics*, vol. 7 Suppl 4, no. Suppl 4, dec 2006, p. S19.

[75] A. M. Yip and S. Horvath, "Gene network interconnectedness and the generalized topological overlap measure," *BMC Bioinformatics*, vol. 8, no. 1, 2007, p. 22.

[76] Z. H. Zhang, D. J. Jhaveri, V. M. Marshall, D. C. Bauer, J. Edson, R. K. Narayanan, G. J. Robinson, A. E. Lundberg, P. F. Bartlett, N. R. Wray, and Q. Y. Zhao, "A comparative study of techniques for differential expression analysis on RNA-seq data," *PLoS ONE*, vol. 9, no. 8, 2014, pp. 1–11.

APPENDIX A

CODE FOR METHODS FOR INFERENCE IN GENE REGULATORY NETWORK USING LOOPY BELIEF PROPAGATION ALGORITHM

A.1 Code

```
import pandas
import pandas as pd
import numpy as np
import scipy.stats
import math
import igraph
import random
k=0.05
# read the files
rt=pd.read_csv("tf_data12.tsv", sep = '\t')
data = pd.read_csv ("counts.csv")
#regulatory network data
network_data_1= rt[rt.TF.isin(data.geneid)].\
reset_index(drop=True)
network_data_1= rt[rt.Gene.isin(data.geneid)].\
    reset_index(drop=True)
network_data_1 = network_data_1[network_data_1.TF != \
                                network_data_1.Gene]
network_data_1=network_data_1.\
    drop_duplicates(['TF', 'Gene'])
```

```
network_data_1=network_data_1[0:255]
genes=pd.concat([network_data_1.TF,network_data_1.Gene]).\
    reset_index(drop=True)
# grouping the data according to TF
grouped = network_data_1.groupby(["TF"])
df1=[grouped.get_group(x) for x in grouped.groups]


#check source node and target node
source_gene= network_data_1.TF.unique()
dat_source_counts=data[data["geneid"].\
                isin(source_gene)].reset_index(drop=True)
target_gene= network_data_1.Gene.unique()
dat_target_counts=  data[data["geneid"].\
                isin(target_gene)].reset_index(drop=True)
target_gene= list(np.unique(genes))
data_target_gene=  data[data["geneid"]\
                .isin(target_gene)].reset_index(drop=True)
gene2=list(data_target_gene.geneid)
data_target_gene= data_target_gene.set_index("geneid")
data_target_gene['sum']=data_target_gene.sum(axis=1)
data_target_gene = data_target_gene['sum']
```

```
total=data_target_gene.sum()
data_target_gene=np.log(data_target_gene*100000/total)
total=data_target_gene.sum()
data_target_gene=data_target_gene/total
data_target_node= {'gene_id':gene2,'potential':\
                   list(data_target_gene)}
data_target_node = pd.DataFrame (data_target_node, \
                    columns=['gene_id','potential'])
#first term,
gene=list(set(list(network_data_1['Gene'])))
tf=list(np.unique(network_data_1['TF']))
def pote_term(gene,data):
    for i in range(0,len(data)):
        if(gene=="".join(data['gene_id'][i])):
            return data.iloc[i,1]
#set states of genes
def n_g_state(pot1,data_target_node):
    r=np.mean(data_target_node['potential'])
    if  pot1 > r:
        return 1
    elif pot1 < r:
```

```
            return -1
        else:
            return 0
#set states of TF
def n_tf_state(pot1,data_target_node):
    r=np.mean(data_target_node['potential'])
    if  pot1 > r:
        return 1
    elif pot1 < r:
        return -1
    else:
        return 0
#first term  of Energy function
def first_term(n_g,pot1):
    pot2 = 1-(pot1-0.5)*1
    pot3=1- (pot1-0.5)*-1
    pot4= 1-(pot1-0.5)*0
    return min(pot2,pot3,pot4)
# second term of Energy function
def second_term(n_tf,n_g,gene1,gene2,dat_target,dat_source):
    for i in range(0,len(dat_target)):
```

```
        if(gene2==""".join(dat_target['geneid'][i])):
            counts_2=dat_target.iloc[i,1:25]
            total=counts_2.sum()
            counts_2=counts_2*100000/total

    for i in range(0,len(dat_source)):
        if(gene1==""".join(dat_source['geneid'][i])):
            counts_1=dat_source.iloc[i,1:25]
            total2=counts_1.sum()
            counts_1=counts_1*100000/total2
    corr= abs(scipy.stats.pearsonr(1*counts_1,\
                                   n_tf*counts_2)[0])
    corr1= abs(scipy.stats.pearsonr(-1*counts_1,\
                                    n_tf*counts_2)[0])
    corr2= abs(scipy.stats.pearsonr(0*counts_1,\
                                    n_tf*counts_2)[0])
    r= 1-corr
    r1=1-corr1
    r2=1-corr2
    return min(r,r1,r2)
# third term of Energy Function
```

```
def third_term(n_g,K):
    pen= max(1,K)
    pen1=max(-1^2,k)
    pen3=max(0,k)
    return min(pen,pen1,pen3)



Energy=[]
factor_pote=[]
# set factor potential
for m in range(len(df1)):
    for i in range(len(df1[m])):
        gene1=df1[m].iloc[i].TF
        gene2=df1[m].iloc[i].Gene
        pote1= pote_term(gene1,data_target_node)
        pote2= pote_term(gene2,data_target_node)
        n_tf=n_g_state(pote1,data_target_node)
        n_g=n_g_state(pote2,data_target_node)
        E1=first_term(pote2,n_g)
        E2=second_term(n_tf,n_g,gene1,gene2,\
```

```
                        dat_target_counts ,dat_source_counts)
        E3= third_term(n_g,0.05)
        E=E1+E2+E3
        final_pot=np.exp(-(E))
        Energy.append(E)
        factor_pote.append(np.exp(-(E)))
# set factor node and factor potential
p=["p_%d" % x for x in range(len(factor_pote))]
factr_df= {'factor': p, 'potential': factor_pote}
factor_dictionary = {p[i]: factor_pote[i] for i in range(len(p))}
factor_df=pd.DataFrame(factr_df)

#convert factor potential as factors
factr=[]
for i in range(0,len(df1)):
    df=df1[i]
    for j in range(0,len(df)):
        factr.append(factor(["".join(df.iloc[j].TF), \
                             "".join(df.iloc[j].Gene)]))
# add factors to factor node
pgm_1 = factor_graph()
```

```
for i in range(len(p)):
    pgm_1.add_factor_node(p[i], factr[i])
plot_factor_graph(pgm_1)
# change factor distribution in graph according to p(x)
def change_factor_distribution(f_name, factor):
    pgm_1.get_graph().vs.find(name=f_name)['factor'] = factor
for f_name in p:
    change_factor_distribution(pgm_1.get_graph().\
            vs.find(f_name)['name'],factor_dictionary[f_name])
```