5-31-2023

Machine learning and network embedding methods for gene co-expression networks

Niloofar Aghaieabiane
New Jersey Institute of Technology, na396@njit.edu

ABSTRACT

MACHINE LEARNING AND NETWORK EMBEDDING METHODS FOR GENE CO-EXPRESSION NETWORKS

by
Niloofar AghaieAbiane

High-throughput technologies such as DNA microarrays and RNA-seq are used to measure the expression levels of large numbers of genes simultaneously. To support the extraction of biological knowledge, individual gene expression levels are transformed into Gene Co-expression Networks (GCNs). GCNs are analyzed to discover gene *modules*. GCN construction and analysis is a well-studied topic, for nearly two decades. While new types of sequencing and the corresponding data are now available, the software package WGCNA and its most recent variants are still widely used, contributing to biological discovery.

The discovery of biologically significant modules of genes from raw expression data is a *non-typical* unsupervised problem; while there are no training data to drive the computational discovery of modules, the biological significance of the discovered modules *can* be evaluated with the widely used *module enrichment* metric, measuring the statistical significance of the occurrence of Gene Ontology terms within the computed modules. WGCNA and other related methods are entirely heuristic and they do not leverage the aforementioned non-typical nature of the underlying unsupervised problem.

The main contribution of this thesis is SGCP, a novel Self-Training Gene Clustering Pipeline for discovering modules of genes from raw expression data. SGCP almost entirely replaces the steps followed by existing methods, based on recent progress in mathematically justified unsupervised clustering algorithms. It also introduces a conceptually novel self-training step that leverages Gene Ontology

information to modify and improve the set of modules computed by the unsupervised algorithm.

SGCP is tested on a rich set of DNA microarrays and RNA-seq benchmarks, coming from various organisms. These tests show that SGCP greatly outperforms all previous methods, resulting in highly enriched modules. Furthermore, these modules are often quite dissimilar from those computed by previous methods, suggesting the possibility that SGCP can indeed become an auxiliary tool for extracting biological knowledge. To this end, SGCP is implemented as an easy-to-use R package that is made available on Bioconductor.

MACHINE LEARNING AND NETWORK EMBEDDING METHODS FOR GENE CO-EXPRESSION NETWORKS

by
Niloofar AghaieAbiane

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

May 2023

APPROVAL PAGE

MACHINE LEARNING AND NETWORK EMBEDDING METHODS FOR GENE CO-EXPRESSION NETWORKS

Niloofar AghaieAbiane

Dr. Ioannis Koutis, Dissertation Advisor — Date
Associate Professor of Computer Science, NJIT

Prof. Baruch Schieber, Committee Member — Date
Professor of Computer Science, NJIT

Dr. Zhi Wei, Committee Member — Date
Professor of Computer Science, NJIT

Dr. Senjuti Basu Roy , Committee Member — Date
Associate Professor of Computer Science, NJIT

Dr. Alexandros Tzatsos , Committee Member — Date
Associate Professor of Medicine and Health Sciences,
George Washington University, Washington, DC

BIOGRAPHICAL SKETCH

Author:	Niloofar AghaieAbiane
Degree:	Doctor of Philosophy
Date:	May 2023

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 2023
- Master of Science in Computer Engineering,
  University of Tehran, Tehran, Iran, 2014
- Bachelor of Science in Computer Engineering,
  Azad University - Central Tehran Branch, Tehran, Iran, 2008

Major: Computer Science

Presentations and Publications:

Niloofar AghaieAbiane, Ioannis Koutis, A Novel Calibration Step in Gene Co-Expression Network Construction, *Frontiers in Bioinformatics*, 1, 2021.

Niloofar AghaieAbiane, Ioannis Koutis, SGCP: A Semi-supervised Pipeline for Gene Clustering Using the Self-training Approach in Gene Co-expression Networks, *arXiv*, 2022.

Learning never exhausts the mind

Leonardo da Vinci

To my parents, who have always been my source of strength and motivation.
To my husband, who has been my constant support and inspiration.
To my advisor, Ioannis, whose guidance and mentorship have been invaluable in shaping my career.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my dissertation advisor, Dr. Ioannis Koutis, for his unwavering support and guidance throughout the research process. His invaluable feedback and insightful comments have contributed significantly to the success of this dissertation.

I am also indebted to the members of my dissertation committee, Drs. Alexandros Tzatsos, Baruch Schieber, Zhi Wei, and Senjuti Basu for their valuable feedback and constructive criticism.

I would like to express my deepest gratitude to the department of Computer Science for providing me with financial support during my Ph.D. studies. This generous support allowed me to focus solely on my research, without the burden of financial constraints, and I am truly grateful. I would like to express my sincere gratitude to the National Sanitation Foundation for providing funding for my research through the grant (#2039863 and #1813374). This support enabled me to carry out my research and make significant contributions to the field of environmental engineering.

I thank my colleagues and friends for their encouragement and support. Their willingness to engage in intellectual discussions and exchange of ideas has been invaluable.

I am also grateful to my family for their unwavering support and encouragement throughout my academic journey. Their love and support have been my greatest source of strength. Finally, I would like to express my deepest gratitude to my husband, who has been my constant source of support and encouragement. His unwavering love, patience, and understanding have been instrumental in helping me complete this dissertation.

TABLE OF CONTENTS

Chapter **Page**

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

Figure Page

LIST OF FIGURES (Continued)

Figure Page

LIST OF FIGURES (Continued)

Figure **Page**

LIST OF FIGURES (Continued)

Figure **Page**

LIST OF FIGURES (Continued)

Figure **Page**

CHAPTER 1

GENE CO-EXPRESSION NETWORKS

High-throughput technologies such as DNA microarrays and RNA-seq are used to measure the expression levels of large numbers of genes simultaneously. To support the extraction of biological knowledge, individual gene expression levels are transformed into Gene Co-expression Networks (GCNs). In a GCN, nodes correspond to genes, and the weight of the connection between two nodes is a measure of similarity in the expression behavior of the two genes. GCN construction and analysis is a well-studied topic. In general, GCNs construction and analysis consists of three steps; (i) construct a fully connected weighted using gene expression data as GCN, (ii) perform network clustering to find clusters of genes commonly called *modules*. (iii) perform gene ontology enrichment to evaluate the module quality. The specific implementation of these three steps can significantly impact the final output and the downstream biological analysis.

In this chapter, the necessary background on the biological information relevant to this dissertation is provided. This information will be presented in Section 1.1. Following this, a detailed explanation of gene expression is given in Section 1.2, gene expression networks are described in Section 1.3, and popular frameworks for analyzing these networks are discussed in Section 1.4.

1.1 Biological Background

Deoxyribonucleic acid (DNA) and Ribonucleic acid (RNA) are two fundamental molecules in cells. Figure 1.1 shows a molecule of DNA and RNA. DNA consists of a long strand of polymers of nucleotide bases (small molecules) each made of sugars and phosphate groups. These bases are referred to as Adenine (A), Guanine (G), Cytosine (C), and Thymidine (T). DNA may be single or double-stranded (well-known double helix). In double-stranded DNA, one of the strands is called a complementary DNA strand and is shown with cDNA. And for any bases in one of the strands, there is a complementary base in the other strand; Adenine always binds (only) with Thymine and Guanine always binds (only) with Cytosine [69, 98]. RNA is a single strand, a polymer of nucleotides. Like DNA it consists of the three bases Adenine (A), Guanine (G), and Cytosine(C), but instead of the Thymine (T), it contains its resemblance, Uracil (U) which is not found in DNA. RNA has multiple types and roles in cells, like transport RNA (tRNA), messenger RNA (mRNA), and ribosomal RNA (rRNA) [69, 98], and mRNA is a type of RNA involved in protein synthesis. The correspondence between DNA and RNA is stated by the "Central Dogma of Molecular Biology" [69] (see Figure 1.2)

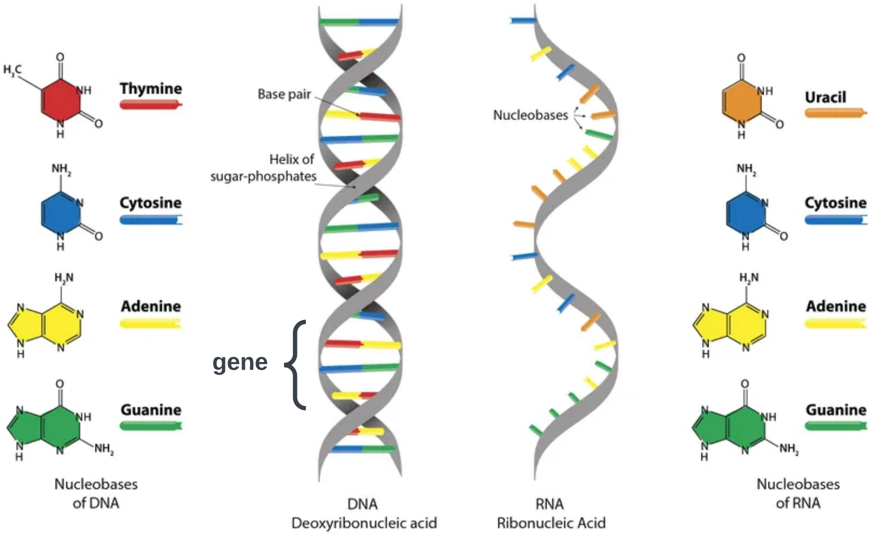


Figure 1.1 A double-stranded DNA and a single-stranded RNA molecule. Two strands of the DNA molecule are called DNA and cDNA (complementary DNA). A DNA molecule consists of Adenine (A), Guanine (G), Cytosine (C), and Thymidine (T) bases. RNA molecule consists of Adenine (A), Guanine (G), Cytosine (C), and Uracil (U) bases. A continuous stretch of nucleotides in DNA is called a gene. Genes carry the information for protein-encoding.

The chromosome, a complete DNA molecule, is very long. In humans, for instance, there are 22 chromosome pairs along with chromosome X and Y with a total 3.1 billion nucleotides. Almost 10% of the chromosome regions are occupied by genes [69, 98]. Genes are a continuous stretch of nucleotides in DNA that code for essential information in the cells. This information is transcribed to produce a functional RNA, commonly proteins [69, 98].

1.2 Gene Expression

Gene expression is the process by which the information from a gene resulted in a gene product, commonly proteins [69, 98]. The correspondence between DNA and a protein is stated by the Central Dogma of Molecular Biology (see Figure 1.2). Proteins are complex molecules that play many critical roles in cells' survival, fitness, and functionality [13, 98]. A cell may need thousands of particular proteins which means that some particular gene information needs to be expressed many times. This in turn increases the abundance of the corresponding mRNA. Therefore, by measuring the levels of mRNA, one can get an understanding of the intensity of the gene expression [98, 68]. There are two major technologies to quantify the intensity of gene expression; DNA Microarrays and RNA-sequencing (RNA-seq) [43, 98, 68]. The general workflow of these technologies is illustrated in Figure 1.3.

1.2.1 DNA microarrays

DNA microarray is a tool to measure the intensity of the RNA transcripts of genes in given cells. Throughout this method, the process of hybridization is used; hybridization is the process where DNA and its complementary are combined into a single molecule by binding each nucleotide to each complementary nucleotide [98].

A DNA microarray consists of a solid surface on which strands of polynucleotides, called probes, have been attached or synthesized in a fixed position, called

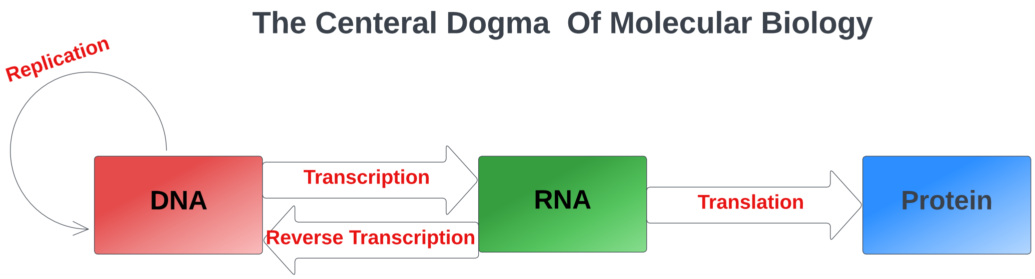
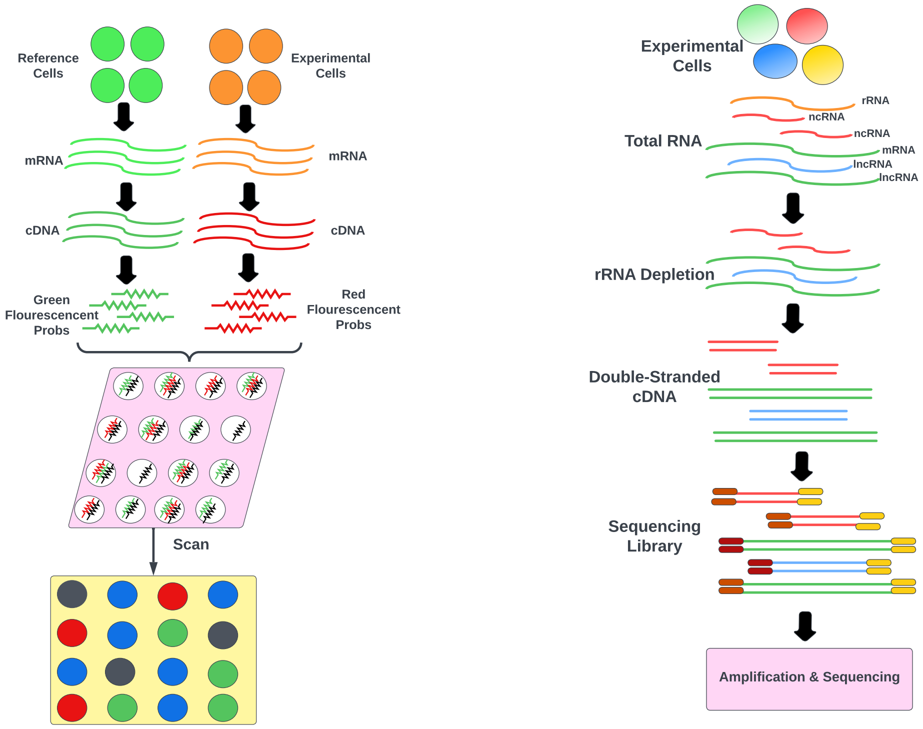


Figure 1.2 The Central Dogma of Molecular Biology. A DNA molecule results through the replication process. An RNA molecule results from DNA through the transcription process, and reversely, DNA results from RNA through the reverse transcription process. Protein results from RNA through the translation process.



(a) (b)

Figure 1.3 High-throughput gene expression technologies; DNA microarray and RNA-sequencing. (a) General workflow for DNA microarray.(b) General workflow for RNA-sequencing.

spots. In other words, probs are gene-specific DNA fragments and spots that represent a known DNA sequence or gene. The general workflow of DNA microarray is depicted in Figure 1.3a. (i) the mRNAs are collected from a set of experiment samples and a set of reference samples. These mRNAs are called targets. (ii) targets are converted into their corresponding cDNA. (iii) cDNAs in each sample are labeled with a fluorescent dye of a specific color. For example, the experimental cDNAs are labeled with a red fluorescent dye, whereas the reference cDNAs are labeled with a green fluorescent dye. (vi) cDNAs are passed through the probes for hours. The targets bind by hybridization to the probes on the array with which they share sufficient sequence complementary. (v) The array then is washed which eliminates the cDNA which has not hybridized. At this point, each prob on the microarray may be bound to a certain quantity of target that, following our basic assumptions, should be proportional to the level expression of the gene represented by that probe. (vi) A laser light is used to determine the number of cDNAs hybridized in the probes. (vii) A scanner captures the light and then it produces an image so that there is a grid of shined spots corresponding to each probe. (viii) The image is transformed into a number showing the expression of each gene [98].

These gene expression numbers are raw data, and it needs to undergo preprocessing steps, background correction, and normalization, to obtain the final ready gene expression profiles. Background correction means removing signals due to non-specific hybridization. Throughout the DNA microarray process, some signals may occur due to the binding of the small quantity of the sample to non-complementary chains. In background correction, these signals will be removed. Normalization is the process in which non-biological variation in measures signal intensity levels is removed, so the biological differences in gene expression can be appropriately detected. Typically normalization attempts to remove global effects [98, 48].

Several methods have been proposed for DNA microarray normalization, and among them, global normalization (i.e., scaling), quantile methods, intensity-dependent normalization, and robust multi-array average (RMA) are the most popular [83, 15, 89, 45]. Global normalization uses a global factor, mainly the median of log intensity ratios, to normalize the expression values. Multiplying expression by this factor equalizes the mean (median) intensity among compared chips [83]. The goal of the quantile method is to make the distribution probs intensities for each array in a set of arrays the same [15]. Intensity-dependent methods compensate for intensity-dependent biases. For example, locally weighted linear regression (lowess) analysis has been proposed as a normalization method that can remove intensity-dependent effects in the $\log_2 ratio$ values [89]. RMA assumes that all chips have the same background distribution of values. It fits an additive model by iteratively reweighted least-squares or median polish.

1.2.2 RNA-sequencing

RNA sequencing (RNA-seq), also called next-generation sequencing (NGC), is another quantitative measurement of gene expression through massively parallel RNA-sequencing. This technology relies on a sequence of DNA instead of an RNA sequence. The general workflow of RNA-seq is depicted in Figure 1.3b. (i) The RNA molecules, mainly consisting of mRNA and long non-coding RNA (lncRNA), must be captured. During this process, ribosomal RNA (rRNA) must be eliminated. (ii) RNA molecules must be converted into double-stranded cDNAs with a defined size range. To this end, captured RNA is subject to RNA fragmentation in a certain size range. The reverse transcription (RT) will be used to produce the double-stranded cDNAs using RNA fragmentations. RT is a mechanism that produces cDNA from RNA molecules. Alternatively, intact RNAs can be reverse transcribed and then the full-length cDNA can be fragmented. (iii) cDNAs form a library. (iv) The library is sequenced to read depth of $10 - 30$ million reads per sample on high throughput platforms. (v) An adapter sequence on the cDNA ends should be placed for amplification on sequencing. Aligning and/or assembling the sequence reads to a transcriptome, quantifying that overlap transcripts and filtering reads that overlap transcripts. This results in count data also called raw count data [43, 108].

Raw count data need to undergo a normalization step to reduce the systematic technical effect such as depth of sequencing and gene length [2, 63, 94]. This is because more sequencing depth produces more read count for a gene expressed at the same level and differences in gene length generate unequal read count for genes expressed at the same level (longer the gene more the read count). Normalized expression units help to remove batch effects [43, 2, 63, 94, 28, 123, 23]. To this end, several methods have been proposed mainly based on standardizing the data between samples by scaling the number of reads in a given library to a common value across all sequenced libraries in the experiment [94]. Method preference must rely on the type of the data, RNA-seq platform, data scale, and the downstream analysis [2]. Additionally, it has been shown that various methods almost have similar results [63]. Among the methods, differential expression analysis for sequence (DESeq) [6], relative log expression (RLE) [70] (also called DESeq2), trimmed mean of M-values (TMM) [94], reads per kilobase per million mapped reads (RPKM) [79], fragments per kilobase of exon per million mapped fragments (FPKM) [112], reads/counts per million mapped (R/CPM) [23], and transcripts per million (TPM) [61] are the most popular.

DESeq and DESeq2 are based on the negative binomial distribution, with variance and mean linked by local regression, and present an implementation that also gives scale factors [6, 70, 63]. TMM method estimates scale factors between samples using the weighted trimmed mean of the log expression ratios and can be incorporated into currently used statistical methods for differential expression analysis [94, 63]. RPKM is widely used in single-end RNA-seq normalization which rescales gene counts to correct for differences in both library size and gene length [63, 79]. FPKM is similar to RPKM but is used for paired-end RNA-seq experiments [112]. TPM is independent of the mean expressed transcript length and is thus more comparable across samples and species [61]. All these methods have advantages and disadvantages. RPKM is not recommended for differential expression analysis [28]. DESeq and TMM methods are more robust to different library sizes [123]. It has been shown that RPKM, FPKM, and TMM normalize the sequencing depth which can differ significantly between samples and perform poorly when the transcript distribution differs between samples [23]. CPM is a basic gene expression unit that normalizes only for sequencing depth (depth-normalized counts). The RPM is biased in some applications where the gene length influences gene expression, such as RNA-seq.

1.2.3 DNA microarrays versus RNA-sequencing

It has been shown that both methods have their own advantages and disadvantages and they can serve as complementary to each other [73]. RNA-seq is capable of measuring the non-coding RNA while these areas are not measurable by DNA microarrays [115, 122] and it has been turned out that these regions have a paramount role in disease [47]. On the other hand, DNA microarrays are more reliable and cost-effective than RNA-seq [73]. RNA-seq has higher accuracy for a low number of transcripts [115] but they are more sensitive in comparison with DNA microarray [26]. Although both measures enable the identification of a large number of differentially expressed genes (DEG) and almost 78% of DEGs identified by DNA microarray have overlap with RNA-seq, using RNA-seq one can potentially identify more differentially expressed protein-coding genes compared to DNA microarray [91]. More details of their advantages, disadvantages, and differences can be found in [115, 26, 91, 73].

$$G_{m\times n} = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1}, & g_{2,2} & \cdots & g_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{m,1} & g_{m,2} & \cdots & g_{m,n} \end{pmatrix}$$

Figure 1.4 Gene expression data set. $G_{m,n}$ shows the gene expression matrix where m and n are the number of genes and samples, respectively and the entry $g_{i,j}$ is the expression intensity of gene i in sample j.

1.2.4 Gene expression public database

One of the public libraries available for gene expression data is NCBI Gene Expression Omnibus (GEO, `https://www.ncbi.nlm.nih.gov/geo/` (2022)) [9]. This database contains both DNA microarray and RNA-seq data, and each data set is identified by a unique accession code. For example, gene expression data with accession code $GSE7636$ identifies a DNA microarray of "*Arabidopsis thaliana*" with 4 samples. It also contains additional information like the summary of the data, type of data (DNA microarray), overall design of the experiment, contributor(s), citation, submission date, last update, and gene expression platform. In the following web page you can find the information of $GSE7636$; `https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE7636` (2022).

In this dissertation, all gene expression data has been downloaded from the NCBI database. We use both DNA microarray and RNA seq. DNA microarrays were normalized using the robust multi-array average method, and all RNA-seq had been already normalized and published by other research. The RNA-seq units used are RPKM, FPKM, TMM, CPM, and RLE (see Chapter 3). Here, we use matrix $G_{m\times n}$ as the gene expression input where m and n are the number of genes and samples, respectively as it is seen in Figure 1.4.

1.3 Gene Co-expression Networks

Gene co-expression networks (GCNs) are undirected weighted/unweighted graphs in which nodes correspond to genes, and the strength of the link between each pair of nodes is a measure of similarity in the expression behavior of the two genes [111]. GCNs are used to perform tasks such as gene functional annotations [101, 72], raveling the biological process of plant organisms [30] and essential genes microalgae [82], assigning unknown genes to biological functions [72, 116], and recognizing disease mechanisms [84] e.g., for coronary artery disease [65]. The goal is to group the genes in a way that those with similar expression patterns fall within the same cluster, commonly called *module.* GCNs analysis consists of three main steps [34, 115].

i ***Network Construction***: The adjacency matrix of GCNs is constructed by applying a similarity measure on the expression values of gene pairs. It includes the following sub-steps.

   (a) ***Similarity Function Calculation***: It applies a similarity function on the input gene expression data and returns a similarity matrix.

   (b) ***Adjacency Matrix Computation***: It uses the similarity matrix to produce the adjacency matrix of the network.

ii ***Network Clustering***: Genes are clustered using unsupervised network clustering algorithms and produce modules.

iii ***Module Evaluation***: Modules are analyzed and interpreted for gene functionality using gene ontology enrichment.

1.3.1 Network construction

Network Construction step may consist of two main steps; ***Similarity Function Calculation***, and ***Network Adjacency Matrix Calculation***. In the end, this step produces an adjacency matrix of $A_{m \times m}$ where m is the number of genes and entry $0 \leq a_{i,j} \leq 1$ indicates the pairwise association between gene i and gene j.

Similarity function calculation In this step, a similarity function must be used to capture the pairwise association of genes. The output is a matrix $S_{m \times m}$ where m is the number of genes, and entry $s_{i,j} = s_{i,j}$ indicates the similarity value between gene i and j. Figure 1.5 shows this matrix. So far multiple methods have been proposed [66] with their own advantages and disadvantages [102], and among them, correlation and mutual information functions have paid the most attention [109, 74, 18, 106].

Majority of studies have used correlation functions [106, 44, 55, 109, 56, 121]. Pearson correlation is the most widely used method. This measure is unable to detect

$$S_{m \times m} = \begin{pmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,m} \\ s_{2,1}, & s_{2,2} & \cdots & s_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ s_{m,1} & s_{m,2} & \cdots & s_{m,m} \end{pmatrix}$$

Figure 1.5 Similarity matrix $S_{m,m}$ where m is the number of genes. Value $s_{i,j}$ shows the relationship between gene i and gene j.

nonlinear relationships [118] meaning that possibly some meaningful relationships remain uncovered [100] and also it is susceptible to outliers [55, 118]. If the relationship of two genes is nonlinear, (e.g., the joint distribution is Gaussian), then the corresponding Pearson coefficient is a small number approaching zero. Non-parametric correlations like Spearman or Kendall correlations can capture nonlinear associations and are less susceptible to outliers [113]. However, these correlations are less powerful than Pearson correlation [102]. These methods can be the Pearson correlation alternatives [100]. Mutual information (MI), on the other hand, is another method based on information theory. MI is a measure of mutual dependence [104] that captures the nonlinear relationships as well [106, 5]. Several studies have used MI or modifications of MI for pairwise relationships [18, 109, 24, 11, 74], like [31, 74].

Several studies have compared correlation and MI-based similarity functions. In a comparative study, it is shown that computing the MI between each pair of genes is expensive [106] and it is required to discretize the data [106, 118]. It is also shown that correlation methods have advantages over MI; first, with a few sample sizes, the correlation coefficient can be estimated precisely while MI is underpowered [106]. Additionally, MI is more susceptible to outliers than correlation and in the case of linear relationships, correlation methods are more precise. [106]. Finally, it has been shown that MI is not superior to correlation functions [57]. Nevertheless, in another study, it is shown that both correlation-based methods and MI-based methods perform well in global network construction in both simulation and real data application [5]. Maximal information coefficient (MIC), an extension of the MI, also has an inferior performance to correlation and MI [93, 106, 105, 52]. Finally, it turned out that the vast majority of the gene pairwise associations are linear [109, 106] justifying the high application of the Pearson correlation as the measure of similarity.

Recently a new similarity measure,"CCor", has been proposed [44]. To compute the association between each pair of gene i, j, CCor uses the information across all the genes. The authors have compared the performance of their method with Pearson Correlation, Partial Correlation, and topological overlap matrix (TOM) [121]. It is shown that CCor has a similar performance to TOM, and it also has a higher performance than the other methods. The author suggests that the CCor measure can better capture the association between gene pairs and find more meaningful clusters of genes by using information across all the genes [44]. Wang et al. [118] have proposed two association measures, $W1$ and $W2$, for detecting local dependencies throughout the expression data using count statistics, in particular, time series data. The authors have suggested that with comparison to Pearson Correlation, Spearman Correlation, Renyi Correlation, Hoeffding's D, Distance Covariance, MI, and MIC, measure $W1$ has more power for detecting local dependencies, whereas $W2$ detect more general associations, and it is more sensitive to noise [118].

Several functions have been proposed and used as the similarity function. Each of these methods is based on a statistical assumption(s) and has different computational costs [66]. In essence, the core of these measures is to detect a direct association between the observed values [104]. However, selecting the correct similarity function may not be clear, since the underlying biological relationships of the genes are blind [113].

Adjacency matrix computation This step involves generating the final network for the GCN. The network is represented as an $m \times m$ matrix, known as the adjacency matrix A, where m is the number of genes. The values of A satisfy the condition $0 \leq a_{i,j} = a_{j,i} \leq 1$, for $i, j = 1, 2, \ldots, m$. If the similarity matrix S doesn't already have the properties of an adjacency matrix, it undergoes a conversion process to obtain an adjacency matrix. Otherwise, S is used directly as the adjacency matrix. This ensures that the resulting matrix represents the relationships between the genes in the network accurately Figure 1.6 shows the output of this step.

Networks can be categorized into two types: *unweighted* and *weighted*. Various methods have been proposed to binarize the similarity matrix S and create an unweighted GCN. Some studies calculate the adjacency matrix by assessing the statistical significance of hypothesis testing [100]. Other studies use direct *p-values* to binarize the adjacency matrix, assuming that the correlation coefficients are based on a t-distribution for a small number of samples [114]. Hard threshold τ can also be

$$A_{m \times m} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,m} \\ a_{2,1}, & a_{2,2} & \cdots & a_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,m} \end{pmatrix}$$

Figure 1.6 Adjacency matrix $A_{m,m}$ where m is the number of genes. This matrix represents either a weighted or unweighted fully connected network of gene co-expression networks. Either is $a_{i,j} = 0$ or $a_{i,j} = 0$ if the network is unweighted. $0 \leq a_{i,j} \leq 1$ if the network is weighted.

utilized to binarize the adjacency matrix, such as in the "ARACNE" algorithm [74]. However, unweighted networks only indicate the presence or absence of an interaction between two genes, losing information on the strength of the connection. Finding an appropriate hard threshold value can also be challenging. A large value of τ may result in fewer node connections and loss of noise in the network, making it less likely to find modules. In contrast, a small value of τ can lead to a dense network with more noise and meaningless modules [121].

Weighted GCNs have been demonstrated to outperform unweighted GCNs and exhibit greater robustness [121]. To create an adjacency matrix A from the similarity matrix S, various techniques have been proposed [100, 115]. One such method involves converting Pearson correlation coefficients r to z values via Fisher transformation, as shown in Equation (1.1).

$$\begin{aligned} z_{i,j} &= \frac{1}{2} \ln\left(\frac{1 + r_{i,j}}{1 - r_{i,j}}\right) \\ &= arctanh(r_{i,j}) \end{aligned} \tag{1.1}$$

where $r_{i,j}$ is the Pearson correlation coefficient between gene i and j. The resulting z values can then be standardized using Equation (1.2)

$$a_{i,j} = \frac{z_{i,j} - \mu_i}{\sigma_i} \tag{1.2}$$

where μ_i, and σ_i are the expected value and variance over the gene i, respectively. This transformation guarantees that for a given gene the underlying distribution is standard normal with defined variance proportional to the size of the samples [99, 103]. Final values of the adjacency matrix range from -1 and $+1$.

As was discussed, correlation functions are the most popular similarity functions for similarity matrix calculation. Correlation functions range from -1 to $+1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. In [121] two methods have been proposed to convert the correlation coefficient similarity matrix into an adjacency matrix. In the first approach, it is suggested to supersede the absolute value of the negative coefficients using Equation (1.3), where $s_{i,j}$ is the correlation coefficient between gene i and j. These networks are called ***unsigned gene co-expression networks***. This approach results in clustering the genes that have high negative and positive correlation coefficients in the same cluster, while these genes must not be placed in the same module. Let $g_{1,2}$ and $g_{2,3}$ be a large negative and positive correlation coefficients between g_1 with g_2 and g_2 with g_3, respectively. If $g_{1,2}$ is replaced by its absolute value, then g_1, g_2, g_3 may be clustered in the same module which leads to disrupting the structure of the networks [121]. Additionally, the use of absolute transformation makes it impossible to distinguish between gene repression and activation [115]. In the second approach, known as ***signed gene co-expression network***, the coefficients are transformed using Equation (1.4). And, since min and max values in a correlation-based network are -1 and $+1$ [106], respectively, this results in Equation (1.5). Therefore, values less than and greater than .5 indicate the negative and positive coefficients, respectively. In signed GCNs genes that have positively similar co-expressed patterns are clustered in the same module while this never happens for those that have negatively co-expressed patterns. This transformation replaces the negative coefficient by some small number and therefore eliminates the negative co-expressed gene interactions. **It is shown that signed GCNs have a better performance in comparison with unsigned GCNs** [75].

$$a_{i,j} = \mid s_{i,j} \mid \tag{1.3}$$

$$a_{i,j} = \mid \frac{s_{i,j} - \min}{\max - \min} \mid \tag{1.4}$$

$$a_{i,j} = \frac{1 + s_{i,j}}{2} \tag{1.5}$$

1.3.2 Network clustering

The goal of GCNs is to group the genes in a way that those with similar expression patterns fall within the same network cluster, commonly called *module*. The underlying assumption is based on guilt by association (GBA) which states that genes with similar expression patterns are more likely to have similar biological functions [115, 35]. So, it is possible to infer the functionality of unknown genes based on known genes that fall in the same module. Modules are identified using unsupervised graph algorithms. Clustering is the process of grouping a set of objects such that those within a single cluster are highly similar to one another, while objects in different clusters are maximally dissimilar [27]. Different clustering methods use different similarity measures to identify how much two objects are alike. The most widely used similarity measure is the *Euclidean distance*, which calculates the distance between two points in a multidimensional space. If X_n and Y_n are two vectors of size n, then their Euclidean distance is given by Equation (1.6). The lesser the distance, the closer the two objects are. Clustering is completely a blind task meaning that there is no prior information about the belonging of an object to a particular cluster nor is there information about the number of the clusters. And, clusters can be in any arbitrary shape and size. Details of the network clustering on unweighted graphs can be found in [62]. Clustering algorithms generally can be categorized into two classes; *hard clustering* (or non-fuzzy clustering) and *soft clustering* (or fuzzy clustering). In hard clustering, each data point must be assigned to at most one cluster while in soft clustering a data point may be assigned to more than one cluster. In other words, for each node, the probability assigned to each cluster is calculated [29].

$$d_{X,Y} = \sqrt{\sum_{1}^{n}(X_i - Y_i)^2} \tag{1.6}$$

In GCNs, the similarity between two genes i and j is calculated during the network construction step and is denoted as the weight $a_{i,j} = a_{j,i}$ of the network. The network clustering step by itself has a fundamental role in downstream GCN performance. Different clustering may result in different modules of different shapes and sizes. Additionally, a single clustering algorithm with different parameter values also may lead to different modules of different shapes and sizes. Thus, the clustering algorithm along with its setting must be chosen with consideration [115, 27]. So far several algorithms have been proposed and used to detect modules in the network and among them, hierarchical clustering, partitioning, and neural networks have paid the most attention [115]. A comprehensive overview of clustering methods used in gene expression data sets can be found in [46]. Figure 1.7 illustrates these methods.

Hierarchical clustering has two main versions; divisive and agglomerative. Divisive hierarchical clustering is a top-down algorithm in which at the beginning all the genes are considered to be in a single cluster. This cluster then is subdivided into smaller clusters and this process successively continues until the predefined desired number of clusters is produced. Agglomerative hierarchical clustering is the reverse of divisive hierarchical clustering and it is a bottom-up algorithm. In this clustering algorithm, in the beginning, every gene is considered a single cluster. Then the two clusters that are the most similar are merged into a single cluster and this process successively continues until the predefined desired number of clusters are produced [80, 27]. Divisive and agglomerative hierarchical clustering can split or merge clusters based on three approaches; single linkage, average linkage, and complete linkage. In single linkage, two clusters C_1 and C_2 are merged if the pairwise similarity value, one gene in C_1 and the other is in C_2, is the highest in comparison with all other pairwise similarities in other clusters. In average linkage, two clusters C_1 and C_2, are merged if the average over pairwise similarity values, one gene in C_1 and the other are C_2 is the highest in comparison with all other pairwise similarities in other clusters. In complete linkage, two clusters C_1 and C_2 are merged if the smallest over pairwise similarity values, one gene in C_1 and the other from C_2, is the highest in comparison with all other pairwise similarities in other clusters. Merging the clusters also can be done by comparing the centroids per cluster. Two clusters are merged if their pairwise centroids similarity value is the highest in comparison with other pairwise values in other clusters. Centroid is a gene that represents the cluster and there are several ways that describe how to identify the centroids [80, 27, 35].

In partitioning methods, at first, the whole genes are subdivided into a predefined number of clusters. Next, the algorithm tries to gather the genes that are the most similar in the same cluster in a way that those genes have the most dissimilarity with the other genes. kmeans is the most popular version of the partitioning clustering method. It takes the network along with k as the predefined number of clusters. It then randomly generates k centroids. Next, for each gene, it calculates the distance to each centroid and assigns the gene to the cluster that has the highest similarity value. Afterward, the centroid of each cluster is recomputed by considering the new genes which have been assigned to the cluster, and by default is the average over the genes belonging to the same cluster. This process is successively continuous until the genes cluster assignment is not changed. Initial centroids have a significant role in clustering performance. Different initial centroids lead to different clusters and as a consequence different performances. Therefore it is strongly recommended to run the kmeans algorithm several times. Defining the number of clusters in advance also is an essential challenge in this algorithm [80, 27].

Self-organizing map (SOM) is an unsupervised learning algorithm based on artificial neural networks that can also be used for clustering. This algorithm, like k-means, requires the number of clusters k to be specified in advance. SOM starts by randomly choosing k centroids that are linked in a grid structure. In each iteration, a gene is chosen and then the closest centroids along its neighbor are moved toward the gene. This process continues until the movement of the centroids is negligible or it is almost zero [27].

Figure 1.7 compares hierarchical, partitioning, and SOM clustering algorithms on a data set of 40 genes with 2 expression values. The expression intensities are shown by the Experiment 1 and Experiment 2 panel on the right of each figure. Genes in this data are divided into four correct labels. (a) Shows the original genes with their corresponding correct labels. x-axis and y-axis show the expression intensities, Experiment 1 and Experiment 2, colorfully. The panels on the right show the two expression values and their corresponding label. For example, genes with both expression value green are labeled red, or genes with a red value of Experiment 1 and a green value of Experiment 2 value are labeled green. From top to down labels are red, green, blue, and purple. (b) Illustrates hierarchical clustering. The panel on the right shows the hierarchical clustering dendrogram with two expression values. Dendogram cut produces four clusters. As it is seen, hierarchical clustering failed to capture the clusters correctly. Some of the points in clusters blue and purple in the original data (part (a)) have been considered in the single purple cluster by this algorithm. Similarly, some points in cluster red in the original data are considered in the single with green genes by hierarchical clustering. (c) Illustrates the kmeans clustering. The number of clusters is four, and stars show the cluster centroid. The panel on the right shows the two expression values with the corresponding cluster centroid. For example, genes with both green expression values are assigned to the cluster with centroid red. As it is seen, kmeans failed to capture all the clusters correctly. Some of the points in clusters blue and purple in the original data (part (a)) have been considered in the single purple cluster by kmeans. Similarly, some points in cluster green and red in the original data are considered in the single green cluster by kmeans. (d) Illustrates the SOM algorithm that organized the clusters into a grid structure. The panel on the right shows the expression value with the corresponding cluster label. For example, genes with both green expression values are assigned to cluster red. SOM also failed to capture the clusters correctly. In cluster blue, a few points in cluster red are mislabeled by SOM, and cluster purple is absolutely correct [27].

Hierarchical clustering is the most widely unsupervised algorithm used in GCNs [58]. However, it is shown that kmeans approach performs the best on gene expression data sets while hierarchical clustering algorithms tend to produce worse than random results [35, 27, 36, 46, 81]. There are advantages of kmeans that makes a superior solution for an unsupervised method to hierarchical clustering. In kmeans, the assignment of the genes to modules is reexamined multiple times based on the information of the gene to the centroids. Whereas, in hierarchical clustering, the assignment of genes to a module examines only for one time based on local information of the genes' pairwise distance [81]. Both approached are sensitive to outliers [27, 36, 81], and generally kmeans is more sensitive to number of cluster [36, 27, 81]. Finally, it has been shown that the enrichment of the cluster tends to be higher at a lower number of clusters [36].

1.3.3 Module evaluation

Once the modules (cluster of genes) are produced, their quality needs to be determined. The quality of the modules can be determined either internally by using statistical analysis of the clusters or externally by using additional information that was not used in the clustering process itself [27]. The latter is an attempt to find functional relationships among the genes in a module to better elucidate the underlying biology [33].

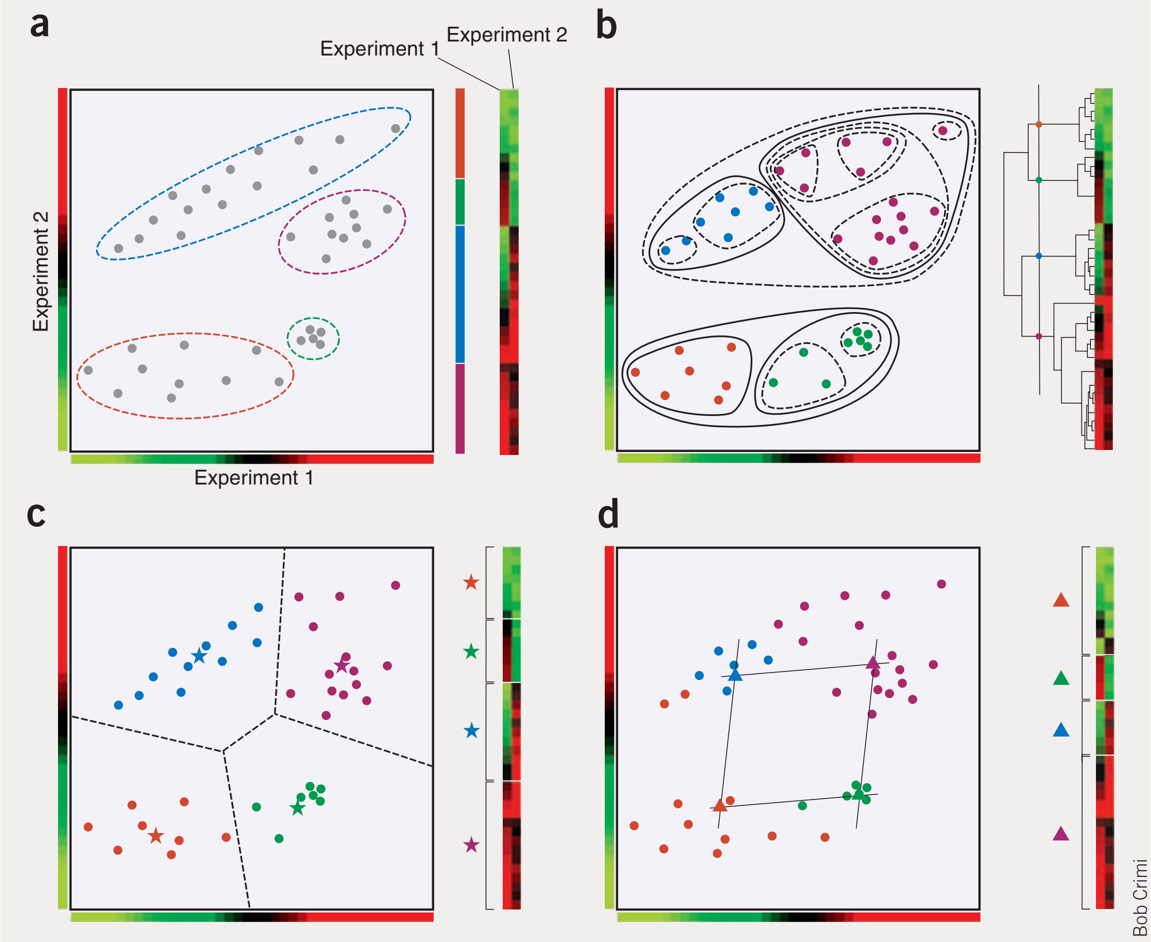


Figure 1.7 Result of clustering algorithms on a dataset of 40 genes with two expression values, shown by Experiment 1 and Experiment 2. (a) Shows the original genes with their corresponding correct labels. x-axis and y-axis show the expression intensities, Experiment 1 and Experiment 2, colorfully. The panels on the right show the two expression values and their corresponding label. (b) Result for hierarchical clustering. Dendogram cut produces 4 clusters. (c) Result for kmeans clustering with 4 predefined number of clusters. Stars show the cluster centroid. (d) Result for SOM algorithm. The panels on the right show the expression value with corresponding cluster label [27].

Internal evaluation methods use cluster geometry and statistical information. There are measures that compare the variance within the cluster with between the clusters, some check the stability of the clusters with respect to noise. More detail is available in [27, 38]. The unsupervised clustering method also may try to optimize this internal method. kmeans clustering, for instance, optimizes the variance within the clusters whereas complete linkage hierarchical clustering minimizes the radius within the clusters [27, 38].

In GCNs, the quality of modules is assessed by considering information beyond what was used in the clustering process. ***The most well-grounded quality measure for a module is the biological relevance of the module***. The commonly used approach to evaluate the module quality is gene ontology (GO) enrichment analysis. This includes the enrichment of genes in a module for known pathways or interactions. GO enrichment uses external information such as gene ontology (GO). GO has two components; GO ontologies which define GO terms and their relationships (graph structure) and GO annotation which defines the associations between the gene products and the terms [120]. Typically, GO enrichment analysis can be utilized to identify which biological processes, functions, or locations are significantly over-or-under-represented. This analysis can also provide insights into new functions that can be inferred from the data and how the genes of interest are distributed across predefined biological categories.

GO is a hierarchical classification of genes and gene products organized in a directed acyclic graph (DAG) structure similar to a tree but unlike the tree, a node may have multiple parents. Nodes are the GO terms and edges that determine the relationships between the nodes which are either "is_a" or "part_of", or "regulate" [120, 50, 124] and all the nodes and edges along their relationships are well-defined [7]. Each GO term has a unique identifier and has a specific biological function [7]. GO terms are categorized into three ontologies; cellular component ontologies (CCO), molecular function ontologies (MFO), and biological process ontologies (BPO). CCO refers to the place in the cell and extracellular in which gene product is active. MFO is defined as the biochemical activity of a gene product at the molecular level. This definition also applies to the capability that a gene product (or gene product complex) carries as a potential. BPO refers to a biological objective to which the gene or gene product contributes. A process is accomplished via one or more ordered assemblies of molecular functions [7, 124]. Each ontology consists of a set of GO terms that are organized in a hierarchy order in the GO graph [124]. The relationships between a gene product (or gene product group) to cellular component, molecular function, and biological process ontologies are one-to-many, reflecting the biological reality that a particular protein may function in several processes in multiple alternatives interactions with other proteins, organelles, or locations in the cell [7]. In other words, depending on the functional characteristics of the particular gene, the gene is assigned to a set of predefined GO terms and therefore each gene is assigned to at least three GO terms with respect to the three categories. A gene can be described with multiple GO terms and additionally, several genes may be assigned to a particular GO term. In the hierarchical structure of the GO graph, the GO terms parent nodes are more general than its children and therefore more genes are assigned to them. Additionally, a GO term in the graph inherits all the properties of its ancestors which are located on every path from the GO terms back to the root. It should be noted that in the GO graph, a term may have multiple parent nodes. Having a GO graph, it is possible to retrieve all the GO terms associated with a particular gene, or conversely, retrieve all the genes annotated to a particular GO term [120, 50]. The characteristics of the GO structure enable powerful grouping, searching, and analysis of genes [120, 7].

Figure 1.8 illustrates a snapshot of the GO. In this figure, the GO term identifiers have been listed at the right. For example, identifiers "GO:0043473" and "GO:0048066" denote the "pigmentation" and "developmental pigmentation", respectively. "Pigmentation" is more general than "developmental pigmentation" and there are more genes assigned to it, and "developmental pigmentation" inherits all the properties of the "pigmentation". Additionally, these two terms are connected by the edge "is_a" which indicates that "developmental pigmentation" is_a "pigmentation" or equivalently "developmental pigmentation" is a subtype of "pigmentation" [124]. GO has three roots; "GO:0005575" for CCO, "GO:0003674" for MFO, and " GO:0008150" for BPO.

Two types of questions can be addressed using GO annotation; if a specific GO term is enriched or depleted in a module (hypothesis-generating-query) or which GO terms are significant in a module (hypothesis-driven-query) [120]. Here we emphasize the latter question and this technique is called GO enrichment. GO enrichment

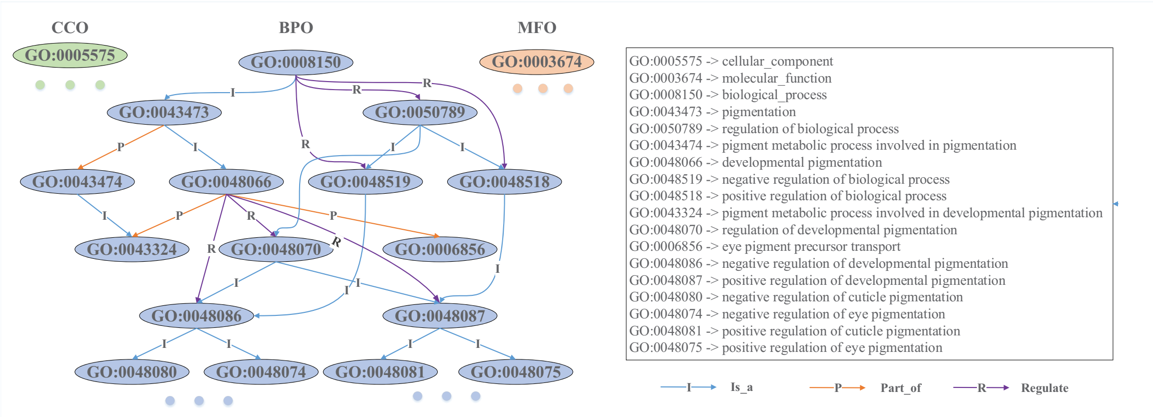

CCO
BPO
MFO
GO:0005575
GO:0008150
GO:0003674
GO:0043473
GO:0050789
GO:0043474
GO:0048066
GO:0048519
GO:0048518
GO:0043324
GO:0048070
GO:0006856
GO:0048086
GO:0048087
GO:0048080
GO:0048074
GO:0048081
GO:0048075
GO:0005575 -> cellular_component
GO:0003674 -> molecular_function
GO:0008150 -> biological_process
GO:0043473 -> pigmentation
GO:0050789 -> regulation of biological process
GO:0043474 -> pigment metabolic process involved in pigmentation
GO:0048066 -> developmental pigmentation
GO:0048519 -> negative regulation of biological process
GO:0048518 -> positive regulation of biological process
GO:0043324 -> pigment metabolic process involved in developmental pigmentation
GO:0048070 -> regulation of developmental pigmentation
GO:0006856 -> eye pigment precursor transport
GO:0048086 -> negative regulation of developmental pigmentation
GO:0048087 -> positive regulation of developmental pigmentation
GO:0048080 -> negative regulation of cuticle pigmentation
GO:0048074 -> negative regulation of eye pigmentation
GO:0048081 -> positive regulation of cuticle pigmentation
GO:0048075 -> positive regulation of eye pigmentation
I Is_a
P Part_of
R Regulate

Figure 1.8 Snapshot of Gene ontology graph. CCO, BPO, and MFO denote cellular component ontology, biological process ontology, and molecular function ontology, respectively. In GO unlike the tree structure, a node may have multiple parents and a parent node is more general than its children therefore more genes have been assigned gene [124].

methods take the entire genes in the gene expression data, called gene universe, and a module as the input. Then, for each GO term in the gene universe, it performs a hypergeometric test to see if the term is significant in the module or not. Finally, it returns the significant GO terms along with their corresponding hypergeometric test. The *p-value* associated with the GO term determines how much the GO term is significant in the module. As usual, the smaller the *p-value* the more significant the GO term is. The null hypothesis is that there is no relationship between the GO term and the module. Rejecting null means that there is an association between the module and the GO term [50]. Let N and M be the number of genes in the universe and modules, respectively. Let n be the number of genes associated with a GO term $GO1$, and m of these genes are found in the module. The *p-value* associated with the $GO1$ is calculated using Equation (1.7).

$$p(X = m) = \frac{\binom{n}{m}\binom{M-n}{N-m}}{\binom{M}{N}} \tag{1.7}$$

The main idea of calculating the *p-values* is to assign significance to the GO terms by comparing the number of observed genes in a specific category with the number of genes that might appear in the same category if a selection performed from the same pool were completely random [50]. In other words, GO enrichment determines among the list of GO terms, which of them appear more frequently than would be expected by chance when examining the set of terms annotated to the input module genes [50]. To this end, an unbiased search for significant GO associations can be done with a bottom-up approach as follows; for every leaf term, calculate the *p-value* with the genes directly associated with it. If any term is significant, do not propagate its gene above. This would provide the most specific node that is significant in that particular branch. If a term is not significant, propagate the annotations to its parents and recalculate with the parent terms. The genes will propagate upwards until a significant node is found or until the root is reached [120]. Correction for multiple experiments is a crucial factor in GO enrichment since performing multiple tests in parallel may greatly increase the false positive [50, 120]. There are several methods for correction and among them, Bonferroni is the simplest such that it multiplies the *p-values* of all terms with the number of parallel tests performed [50, 120], the more detail of the correction can be found in [50]. If genes are propagated all the way up to the root node, the number of tests is equal to the number of terms in the GO hierarchy. In practice, a term would need to have a raw *p-value* less than $4 * 10^{-7}$ for it to be significant at the 1% significance level. Hence, as a general rule, one can increase the power of the statistical analysis by performing the fewest possible number of tests [120].

1.4 Popular Frameworks

Several frameworks and algorithms have been developed for GCNs construction and analysis such as [121, 56, 86, 37, 119, 16, 96]. Among them, Weighted Correlation Network Analysis (WGCNA) [121, 56], is still the most widely accepted and used framework for module detection in GCNs [3, 16, 96, 65, 42]. In recent years, there has been a growing interest to enhance WGCNA and multiple frameworks have been proposed as a modification of this framework. These pipelines mainly utilize an additional step in the form of either pre-processing or post-processing to WGCNA. Among them, CoExpNets [16], K-Module [42], and Co-Expression Modules identification Tool (CEMiTool) [20], have paid the most attention [3, 4].

1.4.1 Weighted gene co-expression network analysis (WGCNA)

Weighted Gene Co-expression Network Analysis (WGCNA) initially was proposed in 2005 [121]. In 2008, authors developed a R [90] package for this pipeline. It consists of the following steps [56].

i Calculates Pearson correlation over the input gene expression $G_{m \times n}$, and produces the similarity matrix $S_{m \times m}$.

ii Converts the negative coefficient into positive using Equation (1.5).

iii Raises the matrix S so that the underlying network confirms to be scale-free.

iv Adds the second-order neighborhood information of the node to the network in the form of topological overlap measure [121] using Equation (1.8).

v Uses hierarchical clustering to produce the final module.

A network is scale-free if the degree of its nodes fall off as a power law $p(k) \sim k^{-\gamma}$ where k is a non-negative real number indicating the weights of the network. The scale-freeness criteria of a network can be measured using the R^2 fitting index of the linear model of $\log(p(k))$ that regresses on $\log(k)$. If R^2 approaches 1, then the scale-freeness criteria is held for the network [121].

TOM formula is as follows.

$$\omega_{i,j} = \frac{l_{i,j} + s_{i,j}}{\min\left(k_i, k_j\right) + 1 - s_{i,j}} \tag{1.8}$$

where $l_{i,j} = \sum_u a_{i,u} a_{u,j}$, and $s_{i,j}$ is the similarity value between gene i and j from previous step, and $k_i = \sum_u a_{iu}$ is the degree of node i. In fact, for every pair of genes i and j, TOM adds the information about the second-order neighborhood of those genes to the network.

WGCNA has proposed the “Dynamic Tree Cut” [58] package for hierarchical clustering that dynamically cuts the dendrogram depending on its shape. The authors have suggested that this results in more flexibility in cluster identification. And, their method is capable of identifying nested clusters, and also in addition to hierarchical clustering it can utilize the advantages of the partitioning method to give better detection of outliers. It is also claimed that the resulting clusters are more enriched with known gene ontologies in comparison with standard hierarchical clustering. Dynamic Tree Cut has two functions. “Dynamic Tree” cut, the first function, is based on divisive hierarchical clustering that relies solely on a dendrogram. It is shown that this version is capable of producing enriched clusters in terms of gene ontologies. This algorithm iteratively divides and combines the clusters that have been returned by the static tree cut and stops whenever the number of clusters becomes fixed. "Dynamic Hybrid", the second function, is based on agglomerative hierarchical clustering and performs in two steps; (i) it finds preliminary clusters (ii) it assigns unassigned objects to the closest preliminary cluster if the distance is small enough [58].

1.4.2 CoExpNets and K-Modules

CoExpNets have utilized kmeans clustering [39] as a post-processing step to the output of WGCNA to produce the final modules.

i Performs WGCNA on the gene expression $G_{m\times n}$ and produces final modules $m_1, m_2, \cdots, m_k$, where k is the number of clusters.

ii Performs kmeans clustering [39] with k number clusters, and set eigengene of the clusters as the initial centroids.

Eigengene of a cluster C is the first component of the singular value decomposition (SVD) factorization over the expression matrix of the genes that belong to C. The authors have shown that their pipeline results in higher module enrichment [16]. Similar to CoExpNets, K-Module also performs an additional kmeans clustering [39] as an extra step to WGCNA output [42].

1.4.3 Co-Expression modules identification tool (CEMiTool)

Co-Expression Modules Identification Tool (CEMiTool) is a pipeline that incorporates an extra pre-processing step to filter the genes using the inverse gamma distribution [20]. It then performs WGCN to produce the final modules.

CHAPTER 2

A NOVEL PREPROCESSING STEP TOWARD THE GENE CO-EXPRESSION NETWORK CONSTRUCTION

High-throughput technologies such as DNA microarrays and RNA-sequencing are used to measure the expression levels of large numbers of genes simultaneously. To support the extraction of biological knowledge, individual gene expression levels are transformed into Gene Co-expression Networks (GCNs). In a GCN, nodes correspond to genes, and the weight of the connection between two nodes is a measure of similarity in the expression behavior of the two genes. In general, GCN construction and analysis includes three steps; (i) Network Construction, (ii) Network Clustering, (iii) Module evaluation. The specific implementation of these three steps can significantly impact the final output and the downstream biological analysis. GCN construction is a well-studied topic. But given its widespread use and applicability, the possibility of improving existing frameworks is tantalizing and motivates further research. Currently, the software package WGCNA (see Subsection 1.4.1) appears to be the most widely accepted standard. Existing algorithms rely on relatively simple statistical and mathematical tools to implement these steps. We hypothesize that the raw features provided by sequencing data can be leveraged to extract modules of higher quality. A novel preprocessing step of the gene expression data set is introduced that in effect calibrates the expression levels of individual genes, before computing pairwise similarities. Further, the similarity is computed as an inner product of positive vectors. In experiments, this provides a significant improvement over WGCNA, as measured by aggregate *p-values* of the gene ontology term enrichment of the computed modules.

Manuscript for this section is published in [3].

2.1 Introduction

The availability of high-throughput technologies like DNA microarrays [93] or RNA-sequencing [43] (RNA-seq) has motivated several approaches for developing a computational understanding of genes and their functionalities. A prominent example is gene co-expression networks (GCNs) that are used to perform tasks such as functional annotations [101, 72], biological process [30], pathway analysis [72, 116], and disease mechanism understanding [84]. In a GCN, nodes correspond to genes, and the weight of the connection between two nodes is a measure of similarity in the expression behavior of the two genes [111].

In general, given a gene expression data set (provided by DNA microarray or RNA-seq) a GCN pipeline includes the following steps; *1-Network Construction*: Computation of the adjacency matrix of the GCN. This includes two sub-steps of *Similarity*

Calculation of a similarity value for each pair of genes, *2-Adjacency*: Further processing of these similarity values to construct a network encoded by its adjacency matrix, *3-Clustering*: Computation of clusters of genes in the network, commonly called *modules* [100, 115], and *4-Evaluation*: Evaluation of the modules based on measuring their *enrichment* with Gene Ontology (GO) terms [50]. Modules can later divulge significant biological intuition.

The specific implementation of these steps can significantly impact the final output and the downstream biological analysis. In particular, the similarity and adjacency steps can be implemented in various ways. For example, framework Petal [86] instantiates them as follows: (i) *Similarity*: Computation of the Spearman correlation, (ii) *Adjacency*: Construction of an initial network using the signum function and further modification so that it follows certain scale-free and small-world criteria [8]. On the other hand, WeiGhted Correlation Network Analysis (WGCNA) which is the most widely accepted framework for GCN construction takes the following steps: (i) *Similarity*: Computation of the Pearson correlation, (ii) *Adjacency*: Conversion of the negative correlation values into positive, further powering the coefficients so that the resulting network follows the scale-free criteria and adding information about second-order neighborhoods of the network, in the form of what is called the Topological Overlap Measure (TOM) of the network [121, 56].

GCN construction and analysis is well studied, for over a decade. But given its widespread use and applicability, the possibility of improving existing frameworks is tantalizing and motivates further research. We hypothesize that the raw features provided by sequencing data can be leveraged to extract modules of higher quality. To this end, we introduce a novel step that precedes the steps of the standard pipeline and is performed directly on the gene expression data set. This is further processing of the level of the expression provided by the DNA microarrays: this in effect calibrates the expression levels of individual genes, before computing pairwise similarities. Further, we deviate from standard frameworks that use statistical measures for the similarity computation [66], and instead use a geometric measure, cosine similarity. Specifically, we compute similarity as a simple inner product of vectors of positive numbers. This is appropriate for our context, since expression levels are positive numbers, and avoids complications related to the interpretation of negative coefficients that are artificially inserted in the analysis via correlation measures. While simple, these steps have not been considered in earlier literature, to the best of our knowledge. As WGCNA appears to be the most widely accepted standard, we implement the proposed steps as modifications to the WGCNA framework, so that they can be easily incorporated into the current GCN construction and analysis workflow. The rest of the process for network construction is the same as WGCNA, to make things comparable. In multiple experiments, our modifications seem to provide an overall **significant improvement** over WGCNA on real data, as measured by aggregate *p-values* of the gene ontology (GO) term enrichment of the computed modules. Specifically, we ran a set of experiments on six different data sets with sample sizes between 44 up to 438 and we found that in all but one case, calibration combined with geometric similarity resulted in more enriched modules.

2.2 Methods

2.2.1 Proposed steps

We describe the two novel steps that constitute our proposed modification to the standard pipeline.

Calibration step Let G be an $m \times n$ gene expression matrix where m and n are the number of samples and genes, respectively, and the entry $g_{i,j}$ is the value of the expression gene j in sample i, as shown in Section 2.2.1.

$$G = \begin{pmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,n} \\ g_{2,1}, & g_{2,2} & \cdots & g_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{m,1} & g_{m,2} & \cdots & g_{m,n} \end{pmatrix} \tag{2.1}$$

In the calibration step, we filter the raw level of the expressions provided in G. Concretely, let G_j denote the j^{th} column of G that contains the expression of gene j. Also, define μ_j and σ_j^2 as the mean and variance of gene vector G_j. Then for every gene j and sample i we calculate a calibrated expression $s_{i,j}$ as follows:

$$s_{i,j} = \frac{1}{1 + \exp\left(-\frac{1}{\sigma_j^2}(g_{i,j} - \mu_j)\right)} \tag{2.2}$$

It should be noted that $s_{i,j} \geq 0$. In the sequel, we denote by $S = [s_{i,j}]$ the gene expression matrix after the calibration step, and S_j the j^{th} column of S.

Similarity We consider two variants of a similarity measure based on computing simple inner products between positive vectors.

In the first variant, we initially set $S' = S^T S$. Note that $s'_{i,j}$ is the inner product between the calibrated expression levels of genes i and j. These similarity values $s'_{i,j}$ may not be in the interval $(0, 1)$. Therefore, in order to compute similarity values in the range $(0, 1)$ we compute the final similarities $m_{i,j}$ via the following normalization:

$$m_{i,j} = \frac{s'_{i,j} - \min_{i,j}}{\max_{i,j} - \min_{i,j}} \tag{2.3}$$

where $\min_{i,j}$ and $\max_{i,j}$ denote the minimum and maximum entry over row i and column j of S'.

In the second variant, we let

$$m_{i,j} = \frac{S_i^T S_j}{||S_i||_2 ||S_j||_2} \tag{2.4}$$

where S_i denotes the i^{th} column of S, and $|| \cdot ||_2$ denotes the Euclidean norm of a vector. This is precisely the **cosine similarity** between the two vectors S_i and S_j.

In both variants we have $m_{i,j} = m_{j,i}$ and $0 \leq m_{i,j} \leq 1$.

2.2.2 Adjacency

As we discussed earlier, the main goal of this study is to compare the effectiveness of the proposed steps with WGCNA. Let us summarize the WGCNA pipeline. In Subsection 1.4.1 gives more detail of WGCNA:

WGCNA

i Calculate the Pearson correlation on gene expression.

ii Convert the negative values to positive using Equation (2.3).

iii Power the similarity matrix (element-wise) so that the network becomes scale-free.

iv Add topological information (TOM) to the network using Equation (2.5).

Two remarks are due here.

• A network is scale-free if the degree of its nodes follow a power law $p(k) \sim k^{-\Gamma}$ where k is a non-negative real number. The scale-freeness criteria of a network can be measured using the R^2 fitting index of the linear model of $\log(p(k))$ that regresses on $\log(k)$. If R^2 approaches 1, then the scale-freeness criteria hold for the network.

• The topological overlap measure (TOM) calculates the weight $\omega_{i,j}$ between genes i and j in the adjacency matrix by including second-order neighborhood information in gene interactions. For instance, if for two genes i and j there are multiple genes k showing a strong interaction with both i and j, then that adds extra strength in the weight $\omega_{i,j}$. More formally the weight is given in Equation (2.5) [121].

$$\omega_{i,j} = \frac{l_{i,j} + a_{i,j}}{\min(k_i, k_j) + 1 - a_{i,j}} \tag{2.5}$$

where $l_{i,j} = \sum_u a_{i,u} a_{u,j}$, and $a_{i,j}$ is the similarity value between gene i and j from previous step, and $k_i = \sum_u a_{iu}$ is the degree of node i.

2.2.3 Calibration-based pipeline variants

We now describe three pipelines for constructing a network from the raw expression data. They all use steps described in subsections Subsection 2.2.1 and Subsection 2.2.2. We name the variants and specify them as follows:

Alpha

i Apply the calibration step and calculate matrix S according to Equation (2.2).

ii Compute similarities according to Equation (2.3).

iii Power the similarity matrix so that the network becomes scale-free.

Beta

i Apply the calibration step and calculate matrix S according to Equation (2.2).

ii Compute similarities according to Equation (2.4).

iii Power the similarity matrix so that the network becomes scale-free.

Gamma

i Follow steps 1-3 of Beta.

ii Add TOM to the network, according to Equation (2.5).

All three variants include the calibration step and will be compared against the standard pipeline of WGCNA. We include Alpha to contrast it with the pure cosine similarity measure used in Beta and Gamma. Gamma includes TOM and its comparison with Beta shows that including second-order neighborhood information remains an effective tool in synergy with our proposed steps.

2.2.4 Clustering

Several algorithms for detecting modules in the network have been proposed; among them, hierarchical clustering, partitioning, and neural networks have received the most attention [115]. In this study we used the ‘Dynamic Tree Cut’ [58] package in R [90], which is the de facto standard and used with WGCNA. Dynamic Tree Cut is a version of hierarchical clustering that dynamically cuts the dendrogram depending on its shape which results in more flexibility in cluster identification. The authors have suggested that their method is capable of identifying nested clusters, and the resulting modules are more enriched with known GO [58].

2.3 Data, Evaluation, and Results

In this section we discuss the evaluation of our three calibration-based pipelines and their comparison against WGCNA. We use 6 real datasets. For each dataset, we compute modules with the four different pipelines and then compare their quality. The only differentiation in these four different computations is in the construction of the network, as described in the previous section, and all other steps remain the same as in WGCNA.

2.3.1 Data sets

The gene expression data sets have been downloaded from NCBI Gene Expression Omnibus GEO [9]. They are distinguished by their unique GEO Series (GSE) number. The first data set is the gene expression data of Drosophila melanogaster GSE34400 [71], and it contains 44 samples. The second data set is the gene expression data of kidney transplantation in human being patients GSE129166 [117], and it contains 212 samples. The third data set is the gene expression data of transcriptional consequences of pharmacologic PPAR a, d, & g agonist administration in the murine liver, heart, kidney, and skeletal muscle in mus musculus organism GSE27948[1], and it contains 300 samples. The fourth data set is the gene expression data of PAXgene allergic asthma patients at baseline GSE13739 [21] and it contains 309 samples. The fifth data set is the gene expression data of livers of F2 mice (C57BL/6 X DBA/2) deficient in leptin receptor (db/db) of mus musculus GSE30140 [25] and it contains 435 samples. The last (sixth) data set is the gene expression data of changes in HK-2 cells following exposure to nephrotoxic compounds of homo sapiens GSE27211[2] and it contains 438 samples. The data sets are ordered by size, and their results are presented accordingly.

2.3.2 GO enrichment and module quality

The quality of the computed modules is evaluated by measuring their *enrichment* with respect to GO annotation, following a methodology that was established and used in previous works, among others [106, 44].

[1] https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE27948, (2022)

[2] https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE27211, (2022)

Concretely, for each computed module we perform a number of non-conditional hypergeometric tests using function `hyperGTest` of GOstats [32]. To be more specific, we note that GOstats provides an option to choose among three GO ontologies ("Biological Process", "Cellular Component", "Molecular Function"), and also an option to choose a 'test direction', i.e., checking for overrepresented or underrepresented terms. Collectively, there are six different ways of calling the non-conditional `hyperGTest`. We perform all these six tests on each module.

These tests return a set of terms and corresponding *p-values* for each module. As usual, smaller *p-values* indicate a higher statistical significance. Following previous works [106, 44], we keep the five smaller *p-values* for each module, and their geometric mean is viewed as a measure of module quality.

More precisely, let $p_{i,j}$ be the i^{th}-order p-value calculated for module j. We define the quality of module j to be the negative logarithm of the geometric mean of the 5 best *p-values* for module j:

$$Q_j = -(\sum_{j=1}^{5} \log_{10} p_{i,j})/5 \tag{2.6}$$

2.3.3 Pipeline evaluation and comparison

Average cluster quality. Following previous conventions and methodology [106, 44], we evaluate the performance of each pipeline by calculating an average module quality over all modules computed by the pipeline. More precisely, suppose that pipeline x outputs a number n_x of different modules. Then, given definition Equation (2.6) the average module quality is defined as

$$\bar{Q} = (\sum_{j=1}^{n_x} Q_j)/n_x \tag{2.7}$$

Figure 2.1 depicts in bars the average quality $\bar{Q}$ (Equation (2.7)) for each pipeline. It can be seen that Gamma yields better average module quality in all 6 data sets, with the exception of GSE30140. In the same Figure, we also observe that

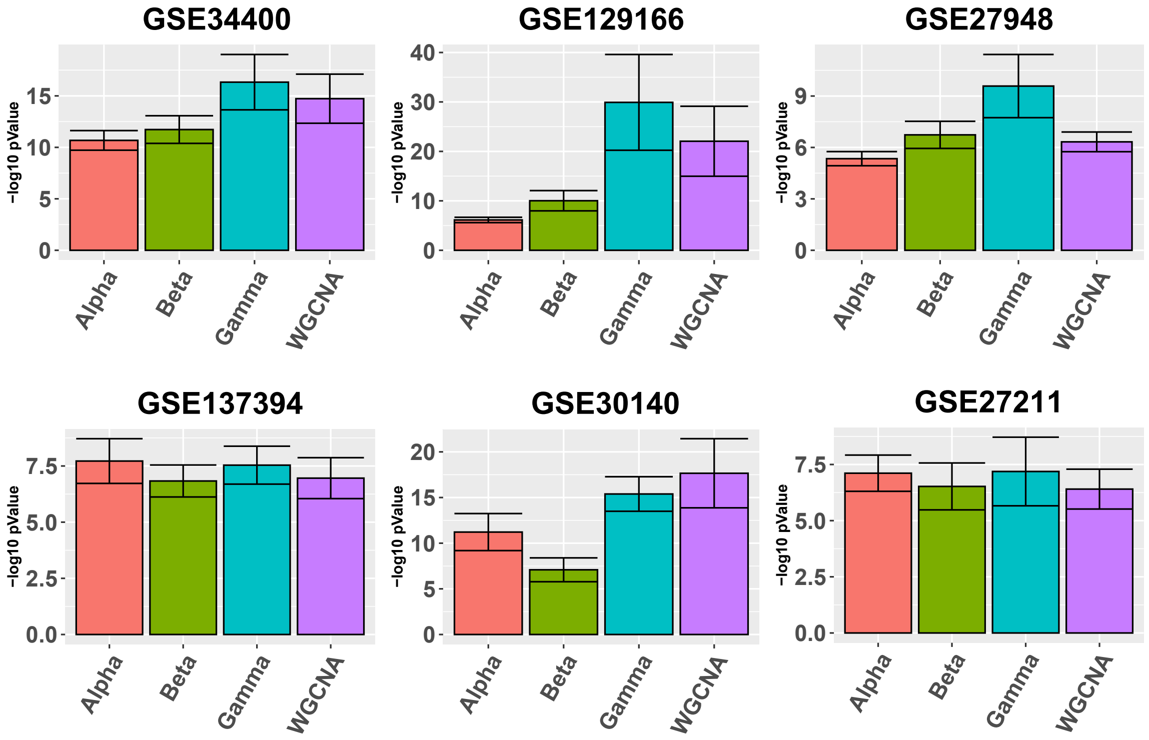


Figure 2.1 Gene ontology enrichment analysis comparing Alpha, Beta, Gamma with WGCNA in six real data sets. The five best GO enrichment *p-values* from all modules are log-transformed, averaged, and shown as bar plots. Higher is better. Error bars indicate the 95% confidence intervals that have been calculated based on the standard deviation of the *p-values*.

in half of the data sets Alpha outperforms Beta, and Alpha performs better on data sets with larger sample sizes.

Ordered Cluster Quality. It has been observed in [35] that expression-based clustering methods produce multiple clusters of relatively low enrichment. In view of this, we take a more detailed look at the *p-values* for each module individually. To this end, we calculate the quality (as defined in Subsection 2.3.2) for each module, then sort the modules according to their quality and plot up to 20 corresponding values, whenever available. As shown in Figure 2.2, the difference between the four pipelines is more pronounced for the higher-quality modules and it becomes less clear for the lower-quality modules that are presumably less important from a biological point of view due to their lower quality.

Figure 2.3 is similar to Figure 2.2, except that it focuses on the 3 best modules for each pipeline, for reading clarity. It can be seen that, in all data sets, Gamma returns the module with the highest enrichment, with the exception of GSE34400 and GSE30140. We note that GSE34400 has the least number of samples which is 44. In GSE30140, as discussed earlier, WGCNA is better on average (Figure 2.1) but even in this, case Beta produces a module of higher quality relative to WGCNA. Notably, in GSE30140, Beta's top cluster is by far better than those of Gamma and WGCNA. This demonstrates a case where TOM leads to lower quality in the top module. The dominance of calibration-based methods, in general, extends to the order-2 module and, while still present in some data sets, diminishes in the order-3 module.

Other comparisons. Besides comparing the enrichment of the computed clusters, multiple other related questions can be considered. Here we perform two additional types of comparisons in order to demonstrate that the modules computed by the calibration-based methods can be significantly different than those computed by WGCNA.

The clustering algorithm used in WGCNA has a number of parameters that can affect the clustering outcome, but in this work, we use the default settings for all four pipelines. With these default settings, the algorithm rejects a number of trivial clusters of small size, and the corresponding genes do not appear in the clustering output. In Table 2.1, we wish to highlight the percentage of such genes that are not assigned to any module. In general, WGCNA leaves unassigned nodes relative to calibration methods, and in particular Gamma. For example, in GSE30140, WGCNA

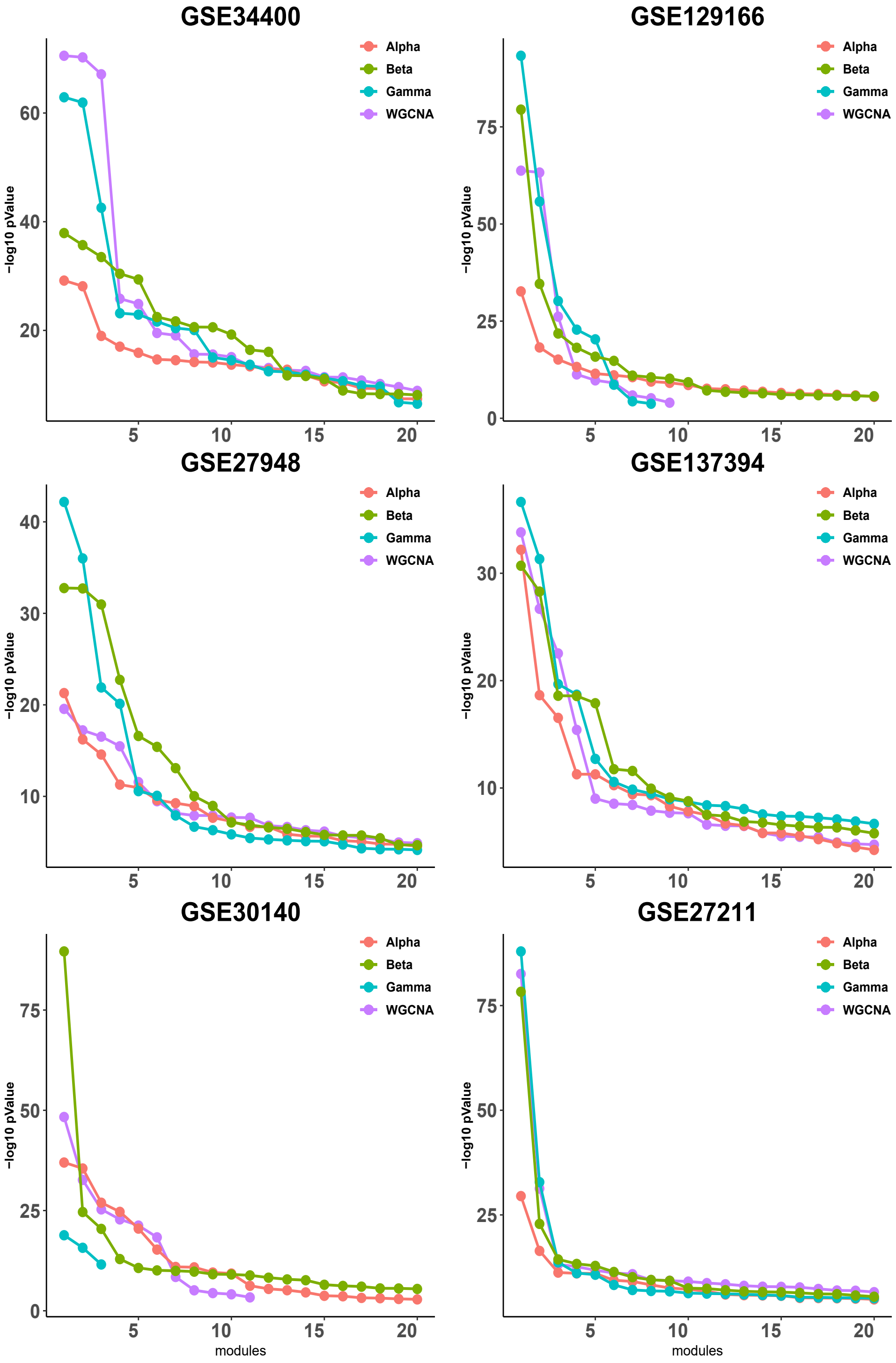


Figure 2.2 Gene ontology enrichment analysis of clusters produced by Alpha, Beta, Gamma with WGCNA in six empirical data sets. For each data set the sorted quality values of the modules are plotted. The x-axis and y-axis indicate the module indices and the module quality, respectively.

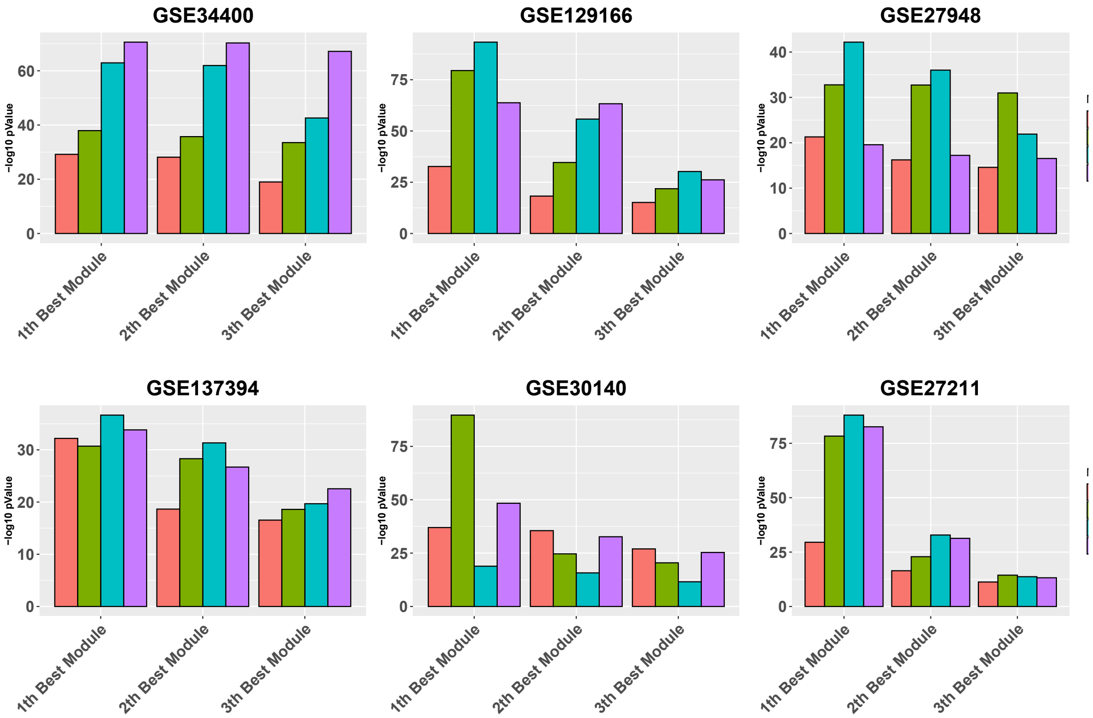


Figure 2.3 Gene ontology enrichment analysis of 10 best modules produced by Alpha, Beta, Gamma, and WGCNA in 6 real data sets. The mean over the 5 best GO enrichment *p-values* from the 10 top modules of each pipeline is compared. The x-axis and y-axis indicate the 10 best modules and the module performance, respectively.

Table 2.1 A Clustering Summary for six datasets on the four pipelines; Alpha (A), Beta (B), Gamma (Γ), WGCNA (W).

Data Set	Pipeline	# clusters	% of unassigned genes
GSE34400	A B Γ W :	34 42 27 39	0.58 32.0 13.0 22.0
GSE129166	A B Γ W :	60 35 8 9	0.01 15.5 2.13 6.50
GSE27948	A B Γ W :	58 60 24 41	2.4 4.5 0.0 29.2
GSE137394	A B Γ W :	29 53 48 40	0.8 62.3 56.00 74.1
GSE30140	A B Γ W :	22 60 3 11	1.3 0.45 18.0 91.1
GSE27211	A B Γ W :	31 66 51 84	0.56 50.0 32.0 38.13

Note: The number of modules and percentages of unassigned genes for the four pipelines Alpha (A), Beta (B), Gamma (Γ), WGCNA (W).

ignores over 90% of the genes for the clustering, i.e., these genes are not included in any module; in comparison, Gamma assigns 82% of the genes to modules. We also observe that there is a significant variance in the number of clusters computed by the four pipelines and that WGCNA has a tendency to produce more clusters than Gamma (with the 'slight' exception of GSE137394). These two facts combined imply that the sizes of the clusters computed by our pipeline are on average bigger than the standard WGCNA pipeline. The precise cluster sizes can be found along with the code and data in the public code repository. We also note that the very recent work in [42] has also identified the issue with unassigned genes in WGCNA, and introduced an additional clustering step that assigns all genes to an appropriately selected module, claiming higher module enrichment. The tendency of our pipeline to automatically do much of what [42] does in a 'forced' way, is an interesting feature of our pipeline.

Recall that in the computation of the quality measures, we kept the 5 GO terms with the smallest *p-values* for each module. In Table 2.2 we focus on the top module, and report how many of these 5 GO terms are shared between each pair of methods. We see that in two datasets (GSE129166 and GSE27211) the overlap between Gamma and WGCNA is significant (five and four, respectively). In other datasets, it can be as low as zero. This indicates that the computed clusters are potentially different (relative to WGCNA) in terms of their biological meaning and significance.

Table 2.2 Overlapping In The Five GO Terms of The Top Module For Each Pair Of Pipelines

Data Set		Alpha	Beta	Gamma	WGCNA
GSE34400	Alpha		3	1	0
	Beta	2		0	0
GSE129166	Gamma	4	3		0
	WGCNA	4	3	5	
GSE27948	Alpha		2	0	0
	Beta	0		0	0
GSE137394	Gamma	0	2		0
	WGCNA	0	0	2	
GSE30140	Alpha		5	0	0
	Beta	5		0	0
GSE27211	Gamma	0	5		4
	WGCNA	0	4	4	

Note: Each table contains two data sets: the first data set is shown in the upper-triangular part of the table, and the second in the lower-triangular part. For instance, the number of GO terms shared between WGCNA and Gamma in GSE27211 can be found in the corresponding cell of the lower part of the third table (=4 in this case).

2.4 Discussion

WGCNA is a widely used software package for identifying biologically meaningful clusters of genes. As highlighted in the title of the original work [121], WGCNA is in fact a versatile *general framework* that can be instantiated in multiple ways into concrete data-processing pipelines. The research community has adopted the GO enrichment of the computed modules as a proxy of the biological utility of a pipeline [106, 44]. Indeed, several research articles have been devoted to studying individual algorithmic components of WGCNA and their impact on GO enrichment, up until recently [42]. For example, the current practice of using Pearson correlation as a similarity measure for pairs of genes has been influenced by the outcome of an extensive study that considered various other similarity measures [106].

In this work, we go beyond modifying the existing WGCNA components and propose an 'architectural' change with the inclusion of a novel calibration layer that precedes the computation of pairwise similarities between the genes. The proposed calibration is a sigmoid transformation of the raw gene expressions that are applied separately to each gene. In addition, we replace Pearson correlation as a similarity measure with an even simpler geometric measure (cosine similarity) that –somewhat curiously– has not been considered before, possibly due to “cultural” reasons related to the background of the research groups that undertook earlier efforts [106]. As discussed in Section 2.3, calibration appears to help the clustering algorithm capture modules with a higher average enrichment in Gene Ontology terms, with the effect being more pronounced for the modules of highest enrichment. It also appears to result in modules that can be qualitatively quite different than those computed by WGCNA.

Ultimately the biological utility of a specific pipeline can only be confirmed by applied biological discovery. While we are encouraged by our results in terms of the GO enrichment, we do not regard our methods as antagonistic to WGCNA but rather as alternatives that can be easily incorporated into existing WGCNA-based pipelines and hopefully provide an additional tool to biologists. For that reason, we provide code that can work directly with the existing WGCNA codebase.

2.4.1 Future considerations

We wish to highlight an additional interesting fact. Topological Overlap (TOM), i.e., the formation of the final network based not on just pairwise similarities but also on second-order neighborhoods of the genes, appears to yield more enriched modules in our calibrated setting, as it has also been observed for other pipelines that are markedly different. This independent confirmation leads to the natural question of whether higher-order neighborhoods can enhance cluster quality as it has been observed recently in other types of datasets (e.g., see [88]); we feel that this is a topic worthy of more exploration. We have also found (although not reported in this paper) that dropping the scale-freeness step from our pipeline reduces module quality, as it does in the standard WGCNA pipeline. Interestingly, the single dataset (GSE30140) where TOM leads to a deterioration in module enrichment for the top module is also the only dataset where powering the network does not yield in practice a good fit to the scale-freeness criterion used by WGCNA. The notion of scale-freeness in biological networks has received significant criticism (e.g., see [17]) and indeed the existence of datasets where scale-freeness is not present may provide a very interesting lead for further research on graph-theoretic alternatives to scale-freeness, especially in terms of its synergy with topological overlap. We leave these questions open for future research.

CHAPTER 3

SGCP: A SEMI-SUPERVISED PIPELINE FOR GENE CLUSTERING USING THE SELF-TRAINING APPROACH IN GENE CO-EXPRESSION NETWORKS

A widely used approach for extracting information from gene expression data employs the construction of a *gene co-expression network* and the subsequent application of algorithms that discover network structure. In particular, a common goal is the computational discovery of gene clusters, commonly called *modules*. When applied to a novel gene expression dataset, the quality of the computed modules can be evaluated automatically, using *Gene Ontology enrichment*, a method that measures the frequencies of Gene Ontology terms in the computed modules and evaluates their statistical likelihood. In this work, we propose SGCP as a novel pipeline for gene clustering based on relatively recent seminal work in the mathematics of spectral network theory. SGCP consists of multiple novel steps that enable the computation of highly enriched modules in an unsupervised manner. But unlike all existing frameworks, it further incorporates a novel step that leverages Gene Ontology information in a *semi-supervised* clustering method that further improves the quality of the computed modules. Compared with already well-known existing frameworks, we show that SGCP results in higher enrichment in real data. In particular, in 12 real gene expression datasets, SGCP outperforms all except one.

3.1 Introduction

High throughput gene expression data enables gene functionality understanding in fully systematic frameworks. Gene module detection in Gene Co-expression Networks (GCNs) is a prominent such framework that has generated multiple insights, from unraveling the biological process of plant organisms [30] and essential genes in microalgae [82], to assigning unknown genes to biological functions [72] and recognizing disease mechanisms [84], e.g., for coronary artery disease [65].

GCNs are graph-based models where nodes correspond to genes and the strength of the link between each pair of nodes is a measure of similarity in the expression behavior of the two genes [111]. The goal is to group the genes in a way that those with similar expression pattern fall within the same network cluster, commonly called *module* [34, 115]. GCNs are constructed by applying a similarity measure on the expression measurements of gene pairs. Genes are then clustered using unsupervised graph clustering algorithms. Finally, the modules are analyzed and interpreted for gene functionality [3].

The de facto standard automatic technique for module quality analysis is Gene Ontology (GO) enrichment. This method reveals if a module of co-expressed genes is enriched for genes that belong to known pathways or functions. Enrichment is a measure of module quality and the module-enriching GO terms can be used to discover biological meaning [50, 3, 16, 96]. Statistically, in a given module, this method determines the significance of the GO terms for a test query by associating *p-values* . The query includes the test direction, either "underrepresented" (under) or "overrepresented" (over), and three ontologies; "biological process" (BP), "cellular component" (CC), and "molecular function" (MF). *p-values* are derived based on the number of observed genes in a specific query with the number of genes that might appear in the same query if a selection performed from the same pool was completely random. In effect, these values identify if the GO terms appear more frequently than would be expected by chance [50]. As usual the smaller the *p-value* the more significant the GO term.

3.1.1 Background on existing GCN frameworks

Several frameworks and algorithms have been developed for GCNs construction and analysis such as [121, 56, 86, 37, 119, 16, 96]. Among them, Weighted Correlation Network Analysis (WGCNA) [56], is still the most widely accepted and used framework for module detection in GCNs [3, 16, 96, 65, 42]. WGCNA uses the Pearson correlation of gene expressions to form a 'provisional' network and then powers the strength values on its links so that the network conforms with a "scale-freeness" criterion. The final network is constructed by adding to the provisional network additional second-order neighborhood information, in the form of what is called topological overlap measure (TOM). Finally, WGCNA uses a standard hierarchical clustering (HC) algorithm to produce modules [58].

In recent years, there has been a growing interest to enhance WGCNA and multiple frameworks have been proposed as a modification of this framework. These pipelines mainly utilize an additional step in the form of either pre-processing or post-processing to WGCNA. Co-Expression Modules Identification Tool (CEMiTool) is a pipeline that incorporates an extra pre-processing step to filter the genes using the inverse gamma distribution [96]. Another study shows that a calibration pre-processing step in raw gene expression data results in increased GO enrichment [3]. Two other frameworks, the popular CoExpNets [16] and K-Module [42] have utilized k-means clustering [39] as a post-processing step to the output of WGCNA. Finally, in a comparative study, it is revealed that CEMiTool has advantages over WGCNA [20].

3.1.2 Our framework: self-trained gene clustering

We have developed Self-training Gene Clustering Pipeline (SGCP), a user-friendly R package (see `https://github.com/na396/SGCP` for the latest version) for GCNs construction and analysis. Its integration with Bioconductor makes it easy to incorporate into existing workflows. SGCP differentiates itself from WGCNA and other pipelines in three key ways: (i) It constructs a network without relying on the scale-freeness criterion which has been controversial [100, 49, 64, 17, 22]. (ii) It clusters the network using a variant of spectral graph clustering that has been proposed relatively recently in seminal work [59]. (iii) It incorporates 'self-training', a supervised clustering step that GO enrichment information obtained from the previous step and further enhances the quality of the modules. To our knowledge, SGCP is the first pipeline that uses GO enrichment information as supervision.

3.1.3 SGCP: the workflow

The workflow of SGCP is illustrated in Figure 3.1; in what follows we give an overview of SGCP and also point to the corresponding sections containing more details. SGCP takes as input a gene expression matrix GE with m genes and n sample and performs the following five main steps:

Network Construction. Each gene vector, i.e., each row in matrix GE is normalized to a unit vector; this results in a matrix G. Next, the Gaussian kernel function is used as the similarity metric to calculate S in which $0 \leq s_{i,j} = s_{j,i} \leq 1$ and $s_{i,j}$ shows the similarity value between gene i and j. Then, the second-order neighborhood information will be added to the network as topological overlap

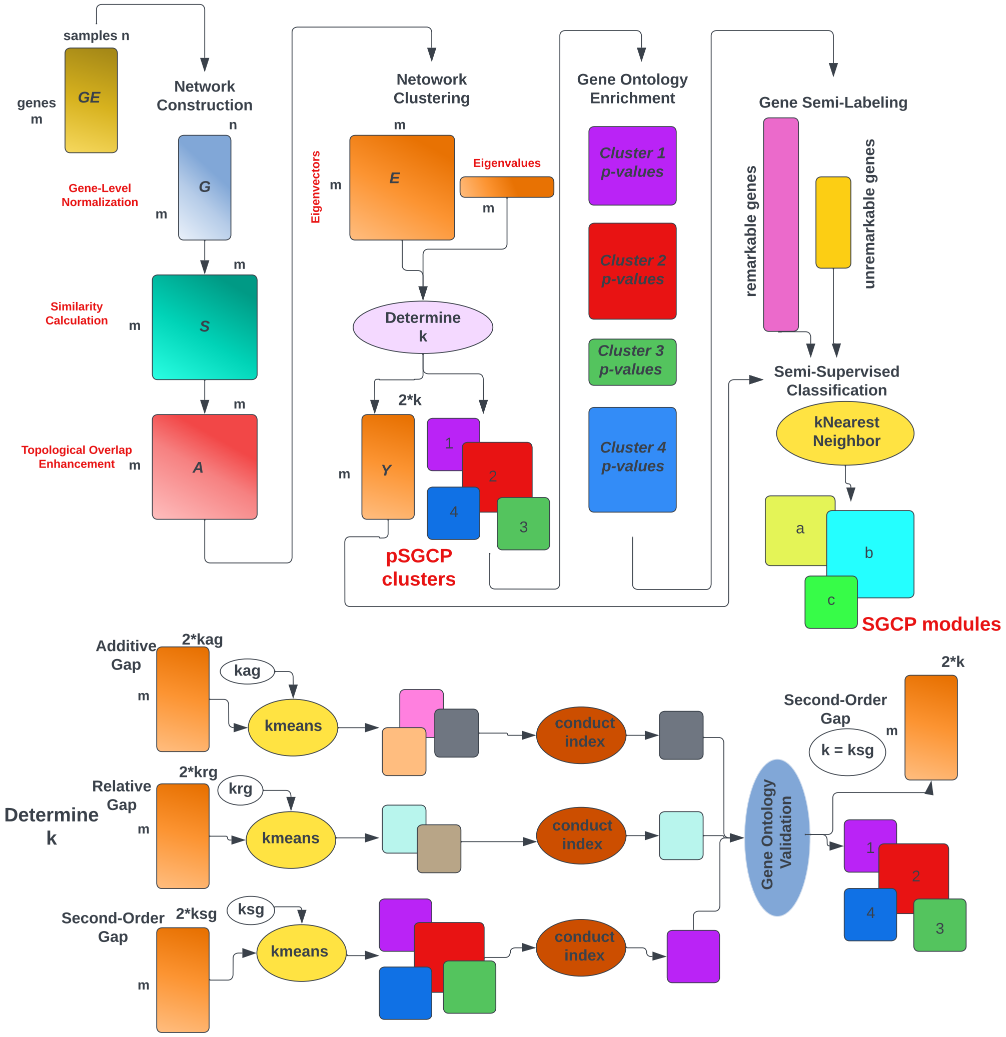


Figure 3.1 The SGCP pipeline for gene clustering in gene co-expression networks. SGCP takes the gene expression matrix GE and outputs clusters and their refinements to modules after the semi-supervised classification steps. The steps for determining the number of clusters k is drawn below the main pipeline.

measure (TOM) [121]. The result of this step is an $m \times m$ symmetric adjacency matrix A (see Subsection 3.4.1).

Network Clustering. Matrix A is used to define and solve an appropriate eigenvalue problem. The eigenvalues are used to determine three potential values (k_{ag}, k_{rg}, k_{sg}) for the number of clusters k (see Subsection 3.4.2). For each such value of k, SGCP computes a clustering of the network, by applying the kmeans algorithm on an embedding matrix Y generated from $2k$ eigenvectors. In each clustering, it finds a *test cluster*, defined as the cluster with the smallest conductance index. The three test clusters are evaluated for GO enrichment, and SGCP picks the clustering that yielded the test cluster with the highest GO enrichment. This clustering is the output of the Network Clustering step, and its clusters are the *initial clusters* (see Subsection 3.4.2).

Gene Ontology Enrichment. GO enrichment analysis is carried out on the initial clusters individually (see Subsection 3.4.3).

Gene Semi-Labeling . Genes are categorized into *remarkable genes* and *unremarkable genes* using information derived from the GO enrichment step. For each cluster, remarkable genes are those that have contributed to GO terms that are more significant relative to a baseline. Remarkable genes are labeled according to their corresponding cluster label. Not all clusters contain remarkable genes, and thus a new number $k' \leq k$ of clusters is determined, and accordingly, k' labels are assigned to the remarkable genes and to the corresponding geometric points in the embedding matrix Y computed in the Network Clustering step. This defines a supervised classification problem.

Semi-Supervised Classification. The supervised classification problem is solved with an appropriately selected and configured machine learning algorithm (either k-nearest neighbors [14], or one-vs-rest logistic regression [14]) with the remarkable genes as the training set. The supervised classification algorithm assigns labels to unremarkable genes. At the end of this step, all the genes are fully labeled, and the final clusters called ***modules*** are produced. SGCP returns two sets of modules, those obtained by the unsupervised Network Clustering step, and those produced by the Semi-supervised classification step. For clarity, in this study, the former and the latter are called ***clusters*** and ***modules*** and we denote the corresponding methods with pSGCP (prior to semi-supervised classification) and SGCP, respectively (see Subsection 3.4.5).

Remark. Computing the Gene Ontology Enrichment is a computationally time-expensive task. The process for selecting k, described in the Network Clustering step, is meant to reduce the amount of computation for the GO enrichment. However, whenever the amount and time of computation are not of concern, multiple other values of k can be evaluated (whenever possible independently, by parallelly running computing processes). This has the potential to produce even better modules. Indeed, in the single case when our method does not outperform the baselines (see Section 3.2), a different choice of k does produce a 'winning' output for our framework.

3.1.4 A comparison of SGCP with existing frameworks

SGCP deviates from commonly used existing pipelines for GCNs in three key ways:
(i) *Network Construction:* While existing pipelines employ a procedure that relies on a controversial scale-freeness criterion, SGCP employs a Gaussian kernel whose computation relies on simple statistics of the dataset that are not related to scale-freeness considerations. To the extent that SGCP is effective in practice reveals that scale-freeness is **not** fundamental in GCNs, affirming the findings of multiple other works on biological networks.
(ii) *Unsupervised Clustering:* Most existing pipelines employ hierarchical clustering algorithms as the main tool for the unsupervised learning step. SGCP first computes a spectral embedding of the GCN and then applies kmeans clustering on it. Crucially, the embedding algorithm is based on a recent breakthrough in the understanding of spectral embeddings of networks.
(iii) *GO-based supervised improvement:* Existing frameworks do not make any use of GO information, except for providing it in the output. This includes methods that work on improving the quality of a first set of 'raw' clusters. SGCP is the first framework that explicitly uses GO information to define a semi-supervised problem which in turn is used to find more enriched modules.

3.2 Results

We present experiments that demonstrate that SGCP outperforms three competing state-of-the-art methods on a wide variety of datasets.

Experimental Setting. We compare pSGCP (i.e., SGCP without semi-supervised cluster improvement) and SGCP with three pipelines (WGCNA, CoExpNets, and CEMiTool) on 12 gene expression datasets: four DNA-microarray datasets [41, 110, 107, 12] and eight RNA-sequencing datasets [95, 60, 51, 19, 87, 77, 78, 92]. These include expression arrays with a wide range of samples from five to 511, various organisms, along with different units [1].[1] The datasets were downloaded from the NCBI Gene Expression Omnibus (GEO) database [10]. Details on the datasets are available in Table 3.1. We note that raw DNA-microarray datasets are normalized using robust multiarray analysis (RMA) [45] which is the most popular preprocessing step for Affymetrix [67] expression arrays data [76]. The Principal component analysis (PCA) of the 12 dataset is available in Appendix B.

[1] Expression units provide a digital measure of the abundance of genes or transcripts.

Table 3.1 Benchmark Datasets And Summary Statistics Of Applying Pipelines WGCNA, CoExpNets, CEMiTool, PSGCP, SGCP On them

						WGCNA		CoExpNets		CEMiTool		pSGCP		SGCP				
Data	Type	Organism	#Samples	Unit	sft	k	#GO Terms	k	#GO Terms	k	#GO Terms	k	#GO Terms	k	#GO Terms	mth	% UNR Genes	% CH Label
GSE181225 [95]	RNA	Hs	5	RLE	26	48	7462	75	9027	32	6252	2	2598	2	2598	ag,rg	1%	0%
GSE33779 [41]	DNA	Dm	90	probes	14	22	5631	19	6213	17	5299	10	4144	7	3821	ag	56%	47.1%
GSE44903 [110]	DNA	Rn	142	probes	30	18	3298	27	4705	14	3303	4	987	4	1059	rg	29%	5%
GSE54456 [60]	RNA	Hs	174	RPKM	30	31	9386	46	14473	22	11056	3	6004	3	600	ag,rg	1%	1%
GSE57148 [51]	RNA	Hs	189	FPKM	14	45	13296	36	14110	33	12027	9	2833	5	2383	sg	46%	24%
GSE60571 [19]	RNA	Dm	235	FPKM	9	21	7107	19	8622	16	5564	2	2969	2	2971	ag,rg	21%	2%
GSE107559 [87]	RNA	Hs	270	FPKM	3	26	10915	20	12499	80	15952	14	5257	12	4913	sg	6%	48.4%
GSE28435 [107]	DNA	Rn	335	probes	22	51	7331	47	7769	31	6566	2	2052	2	2053	sg	12%	0%
GSE104687 [77]	RNA	Hs	377	FPKM	18	31	10339	28	1193	23	11369	2	6426	2	6426	ag,rg	0%	0%
GSE150961 [78]	RNA	Hs	418	TMM	5	9	3619	18	606	17	4856	2	2111	2	2111	ag,rg	9%	0%
GSE115828 [92]	RNA	Hs	453	CPM	12	51	12611	10	7693	10	5231	3	1934	3	1926	sg	33%	0%
GSE38705 [12]	DNA	Mm	511	probes	16	8	3320	12	4123	7	3308	6	2824	4	2610	sg	39%	62%

The first 6 columns contain dataset information. Possible dataset types are DNA-microarray (DNA) or RNA-seq (RNA). Datasets come from the following organisms: Homo sapiens (Hs), Drosophila melanogaster (Dm), Rattus norvegicus (Rn), Mus musculus (Mm). Units are Relative Log Expression (RLE), Reads Per Kilobase of transcript per Million mapped reads (RPM), Fragments Per Kilobase of exon per Million mapped fragments (FKPM), Trimmed Mean of M-values (TMM). The "sft" column indicates softpower used by tested benchmarks to enforce the network to be scale-free. For each of the 5 pipelines, k is the number of clusters and #GO Terms indicated the number of gene ontology terms found in all modules collectively by the pipeline; in particular, a single #GO term will appear once for each cluster where its presence exceeds a threshold of significance in term of its *p-value*. In the case of SGCP, "mth" denotes the method ultimately used for selecting k, "ag": additive gap, "rg": relative gap, and sg: second-order gap. %UNR Genes show the percentage of the entire genes at are unremarkable. %CH Label shows the percentage of the unremarkable genes whose labels have changed after the semi-labeling step.

We look at the following metrics of quality. The log-transformed *p-values* of GO terms identified in modules for each pipeline are explained in Appendix C.

Average Cluster Quality. We follow the previous convention and methodology [106, 44], and evaluate performance by comparing the *p-values* returned by pipelines. Let $p_{i,j}$ be the ith order *p-value* calculated for module j. Then, the quality of module j is defined as $q_j = -(\sum_{j=1} \log_{10} p_{i,j})/n_j$ where n_j is the number of GO terms found in module j. Finally, the quality of framework f is defined as $Q_f = (\sum_{i=1}^{k} q_i)/k$ where k is the number of modules in f. The results are shown in Figure 3.2. SGCP outperforms the three baselines on all datasets, and the same is true for pSGCP, with the exception of GSE38705. We can also see that SGCP is at least as good as pSGCP on all datasets, and in six out of 12 of the datasets, it improves the module quality.

Most significant GO terms. The summary evaluation includes all *p-values* for the GO terms, as reported by GOstats, but here we focus on the top 100 *p-values* for each pipeline. Figure 3.3 reports these *p-values* in the form of 'violin' plots. The y-axis indicates the significance of each GO term in terms of the *p-value*. The top GO terms in pSGCP and SGCP have a higher *p-value* than the corresponding top terms of the other frameworks except for datasets GSE44903 and GSE57148; in GSE57148 only CEMiTool does better than SGCP. It can be also observed that in five datasets (GSE181225, GSE54456, GSE107559, GSE28435, GSE104687), SGCP is *dominant* to other frameworks, as the *least significant* GO term found by SGCP is more significant than the majority of GO terms founds by the other frameworks. In datasets (GSE150961, GSE11582, GSE60571, and GSE38705) the 'violin' for pSGCP and SGCP tends to be higher relative to the other frameworks. In two datasets (GSE33779, GSE38705) the three pipelines have similar performance.

GO terms of most significant module. We consider as most significant or *prominent*, the module that contains the GO term containing the highest *p-value*. We then consider the ten most significant GO terms in the prominent module and we show their *p-values* in Figure 3.4. We observe that, even when restricted to the prominent module, pSGCP, and SGCP report more significant terms than other methods, on all datasets except GSE44903 and GSE57148; in GSE57148 only CEMiTool is better

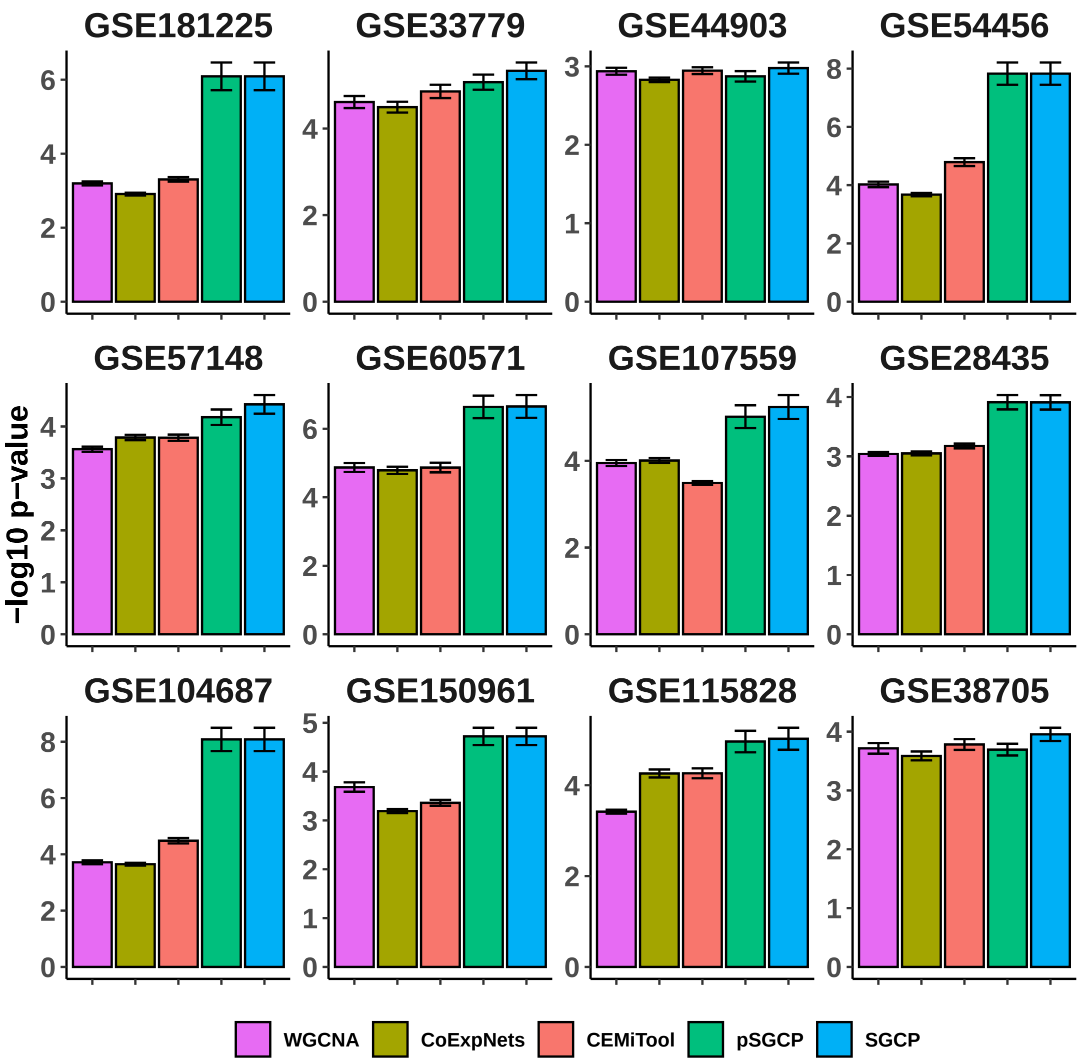


Figure 3.2 Gene ontology enrichment analysis comparing WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP in 12 real datasets. *p-values* are log-transformed. The order of the pipelines from left to right is WGCNA (purple), CoExpNets(yellow), CEMiTool (orange), pSGCP(green), and SGCP (blue). All *p-values* from all modules are pooled, averaged, and shown as a barplot. Error bars indicated the 95% confidence intervals that have been calculated based on the standard deviation of the *p-values*.

than SGCP. In six of the datasets, pSGCP and SGCP are astonishingly better than the other frameworks.

3.2.1 Observations

Overlap in significant GO terms. It is interesting to investigate the overlap of GO terms reported by the different frameworks. To this end, we report two

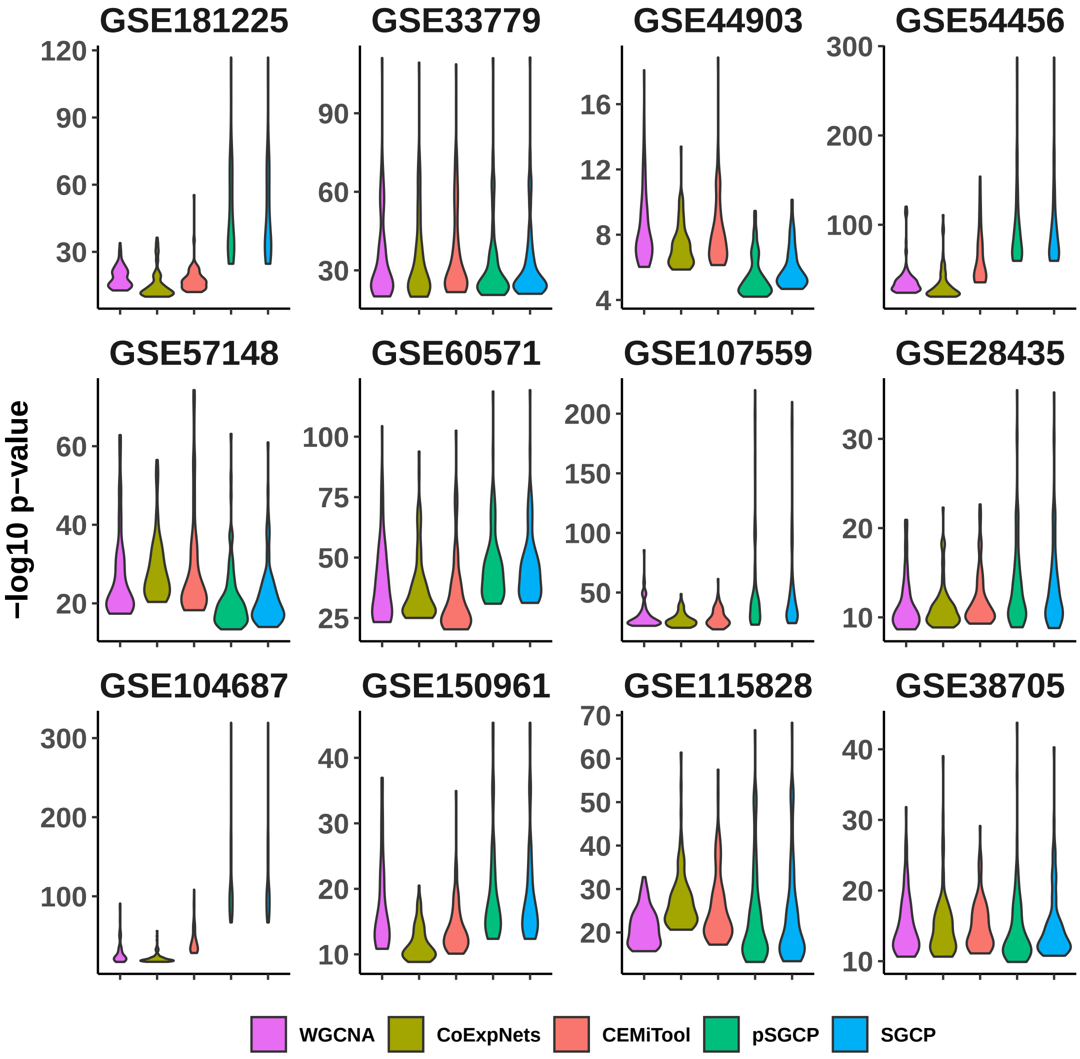


Figure 3.3 Gene ontology enrichment analysis comparing WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP in 12 real datasets. *p-values* are log-transformed. The order of the pipelines from left to right is WGCNA (purple), CoExpNets(yellow), CEMiTool (orange), pSGCP(green), and SGCP (blue). Top 100 most significant *p-values* from all modules are shown as a violin plot.

different measures for overlap, in Figure 3.5. Not surprisingly, pSGCP and SGCP show significant overlaps with each other, as is the case with WGCNA, CEMiTool and CoExpNets, which also share algorithmic components. The overlaps between SGCP and the other three frameworks are smaller, indicating that SGCP reports GO terms that are not reported by the other frameworks. This is even more prominent when focusing on the most significant GO terms, reported in Figure 3.5.

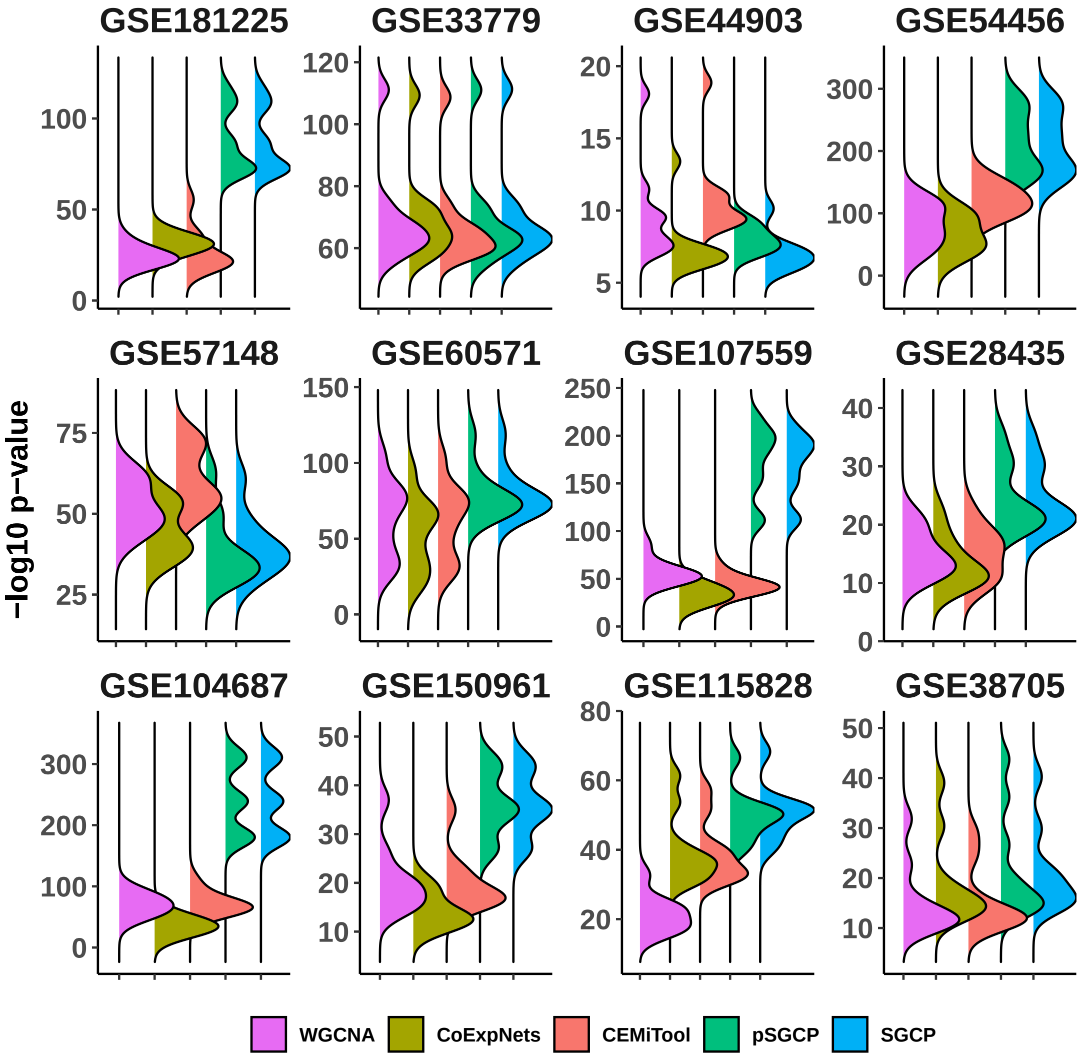
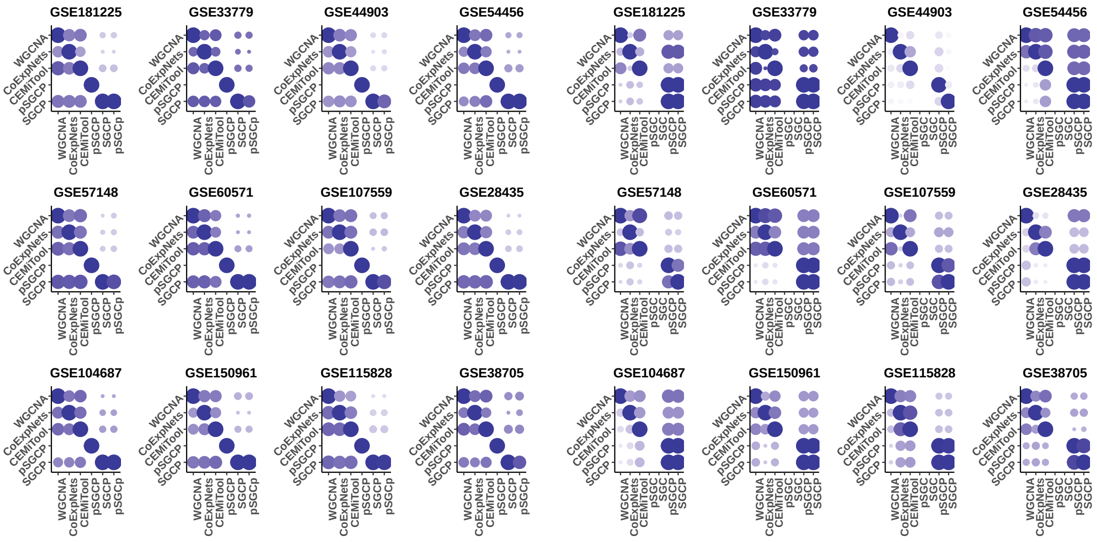


Figure 3.4 Gene ontology enrichment analysis comparing WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP in 12 real datasets. *p-values* are log-transformed. The order of the pipelines from left to right is WGCNA (purple), CoExpNets(yellow), CEMiTool (orange), pSGCP(green), and SGCP (blue). Top ten most significant *p-values* for the prominent module for each pipeline

It is also interesting to consider the information that is implied by the different sizes of the circles in symmetric positions (p, q) and (q, p). In Figure 3.5a, the circles in the lower-triangular part of the 12 arrays are larger than their symmetric counterparts. This shows that when considering all modules the number of GO terms reported by SGCP is smaller. On the other hand, when considering the prominent module (defined as the module containing the GO term with the most significant *p-value*), Figure 3.5b shows that the prominent module for SGCP contains a higher number of statistically significant GO terms, relative to the prominent module of other methods. Overall, these two observations show that SGCP produces a prominent module with a higher density of statistically significant GO terms.

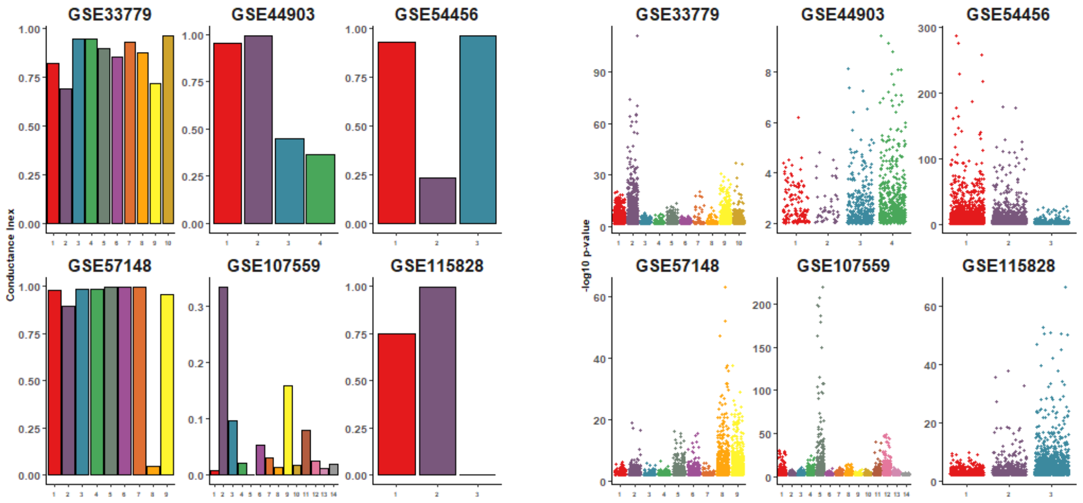


(a) **(b)**

Figure 3.5 Overlaps GO terms reported by WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP in 12 real datasets. In both Figures (a) and (b), for pipeline p in the x-axis and pipeline q in the y-axis, position (p, q) shows the number of unique GO terms reported by both p and q, divided by the number of terms reported by q. Figure (a) accounts for GO terms reported in all modules, while Figure (b) accounts for GO terms reported in the prominent (most significant) modules. The bigger and darker a circle the higher the percentage.

The conductance measure. Spectral clustering targets the computation of clusters with a small conductance index as defined in Section 3.4.2 [59]. Thus, when optimizing for conductance, we implicitly hypothesize that smaller conductance should correspond to higher module enrichment. Figure 3.6 shows this correspondence.

We indeed have observed that there is correspondence between the cluster conductance index and cluster enrichment. Figure 3.6 shows the conductance index of the modules computed by SGC, along with their corresponding enrichment; here we focus on the cases when $k > 2$. It can be seen that in all six data except GSE54456, clusters with smaller conductance indices have higher enrichment. In particular, in GSE107559, the modules with smaller conductance indexes were in order the clusters with label $5, 1, 13, 8, 10, 14, 4, 13$ (see Figure 3.6a). Interestingly, from Figure 3.6b, it can be seen that these clusters have higher enrichment.

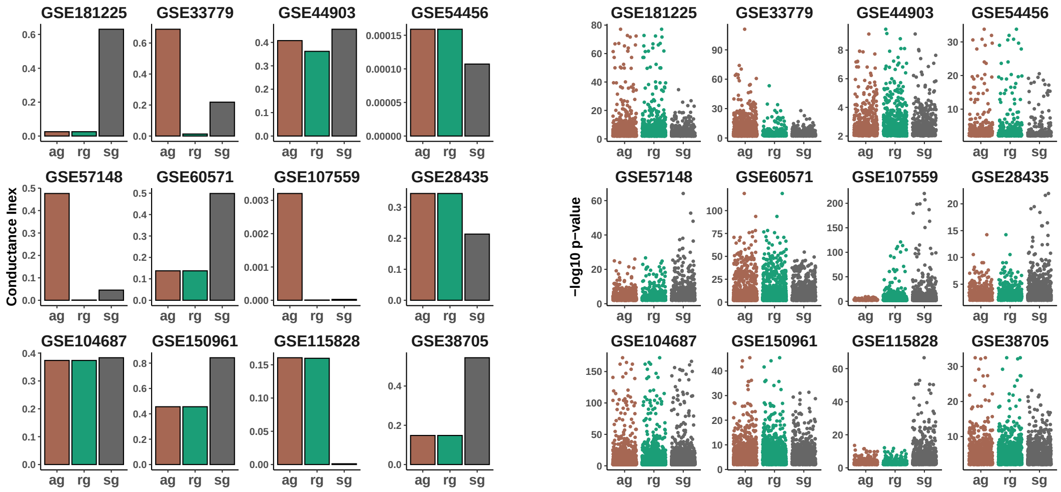


(a) **(b)**

Figure 3.6 Conductance index and log-transformed *p-values* analysis in six real datasets. For each data, the conductance index for the clusters (on the left) along with its corresponding log-transformed *p-values* distribution (on the right) is depicted. (a) Conductance index for each module per data. The smaller the bar, the better the cluster. (b) log-transformed *p-values* for each module per data. The higher the point, the more enriched the GO term.

As discussed in Section 3.1, our framework relies on this connection of cluster conductance with enrichment to automatically compute a value of k before computing the final clustering and the GO enrichment for the modules. In particular, the method computes the enrichment of three test clusters that were picked based on their conductance. These clusters' conductance and enrichment are reported in Figure 3.7, where the general correlation between conductance and enrichment is evident.

Semi-supervised re-classification. Once initial clusters by kmeans are produced, SGCP carries out an additional semi-supervised re-classification of genes to return final modules, as described in Section 3.1. A summary of the impact of this final step is given in Table 3.1 in the SGCP column. "%UNR Genes" indicates the percentage of the total genes that are *unremarkable*, and "% CH Label" specifies the percentage of unremarkable genes whose label changed after the re-classification.



(a)

(b)

Figure 3.7 Conductance index and log-transformed *p-values* analysis for additive gap ("ag"), relative gap ("rg"), and second-order gap ("sg") clusters in 12 real datasets.
(a) Conductance index for the best cluster of each method on the 12 datasets.
(b) log-transformed *p-values* of the selected clusters for "ag", "rg", and "sg" are shown. The higher the point, the more significant the GO term.

Generally, when the percentage of unremarkable genes is small, the final modules agree with pSGCP clusters; this happens in GSE104687, GSE181225, GSE54456, GSE107559, and GSE150961. In contrast, for a higher percentage of unremarkable genes, SGCP assigns new labels to unremarkable genes and changes significantly the clusters' shape and size. The highest unremarkable gene percentages occurred in GSE33779, GSE57148, and GSE38705. The difference in enrichment between the clusters (pSGCP) and modules (SGCP) for these data is shown in Figure 3.8. It can be seen that, in all cases, the number of clusters gets reduced and the overall enrichment of the modules increases. In GSE107559 the percentage of unremarkable genes is relatively low, but re-classification has wiped out two clusters. In general, if there are clusters that are not enriched the re-classification step eliminates these clusters

3.3 Discussion

We have proposed a novel framework for gene co-expression network analysis. Our pipeline integrates multiple novel elements that deviate from existing frameworks in various ways. The GCN construction relies on a similarity measure that unlike

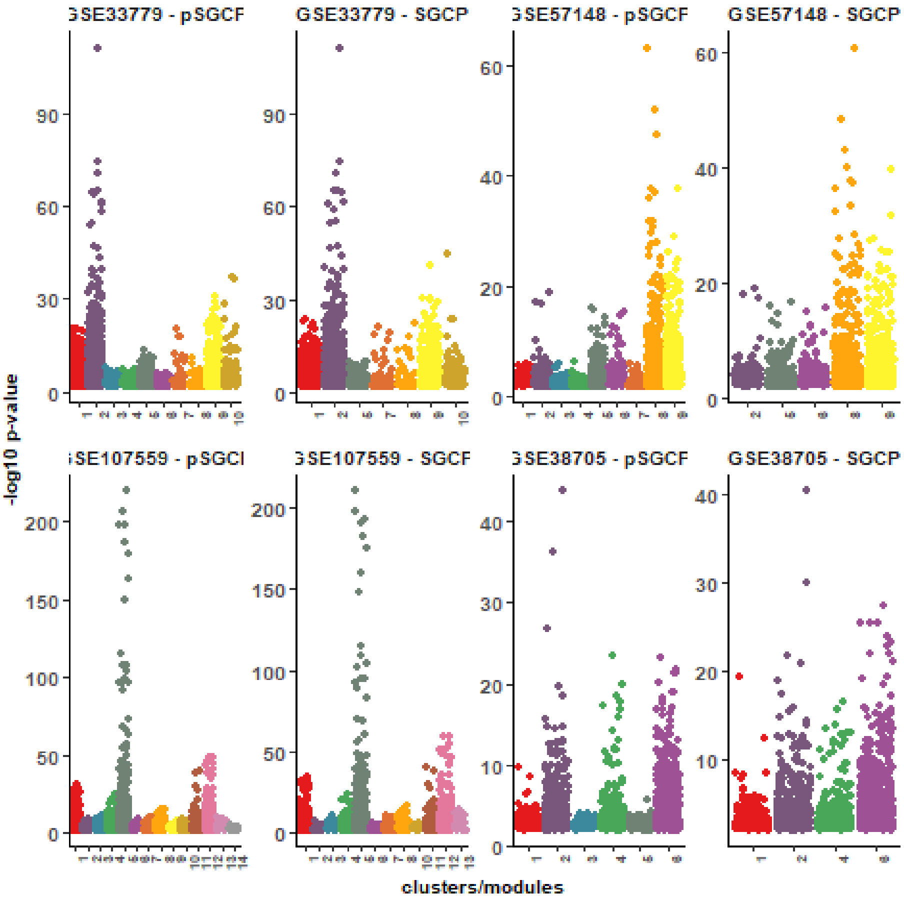


Figure 3.8 Comparing pSGCP clusters and SGCP modules in four real datasets. In all cases, re-classification has resulted in a smaller number of modules relative to clusters. The labels of the eliminated clusters are $3, 4, 6$, in GSE33779, $1, 3, 4, 10$ in GSE57148, $9, 14$ in GSE107559, and $3, 5$ in GSE38705.

previous works does not take into account network scale-freeness. The clustering algorithm departs from the currently used paradigm of Hierarchical Clustering and makes use of recent progress in spectral clustering. The framework also includes a novel semi-supervised re-clustering step that takes into account gene ontology information, unlike previous frameworks that include it only as an end result for module evaluation. We also make the observation that the conductance index, i.e., the optimization objective of spectral clustering is empirically correlated with high module enrichment, providing experimental support to our choice of spectral clustering. Our framework produces modules that have been found to be of superior quality on a comprehensive set of experiments with real datasets.

3.4 Methods

The input is a matrix $GE_{m\times n}$ containing the gene expressions. In GE, rows and columns correspond to genes and samples, respectively. Each entry $ge_{i,j}$ is an expression value for gene i in sample j. SGCP does not perform any normalization or correction for batch effects and it is assumed that these preprocessing steps have been already performed. SGCP is based on 5 main steps. Each step offers parameters that can be adjusted by the user.

3.4.1 Step I: Network construction

Gene-level Normalization. In this step, each gene expression vector, i.e., each row of the matrix $GE_{m\times n}$ is divided by its Euclidean norm which is calculated as

$$\|GE_{i,.}\|_2 = \sqrt{ge_{i,1}^2, \ldots, ge_{i,n}^2}, \tag{3.1}$$

where $GE_{i,.} =< ge_{i,1}, \ldots, ge_{i,n} >$ is the expression vector of gene i. The result of this step is matrix $G_{m\times n}$.

Similarity Calculation. We calculate the variance γ^2 over all $m^2/2$ pairwise Euclidean distances $\|g_i - g_j\|_2^2$. We then calculate the following type of Gaussian kernel for each pair of genes.

$$s_{i,j} = k(g_i, g_j) = \exp(\frac{-\|g_i - g_j\|_2^2}{2\gamma^2}). \tag{3.2}$$

The result is a similarity matrix $S_{m\times m}$ where m is the number of the genes. Note that S is a symmetric square matrix that ranges from 0 for the most dissimilar to 1 for the most similar genes.

Topological Overlap Enhancement. The adjacency of the network is derived by adding second-order neighborhood information to $S_{m\times m}$ in the form of the topological overlap measure (TOM) [121, 56]. The adjacency strength between gene i and j is calculated by the following formula:

$$a_{i,j} = \frac{l_{i,j} + s_{i,j}}{\min(k_i, k_j) + 1 - s_{i,j}}, \tag{3.3}$$

where $l_{i,j} = \sum_u s_{i,u}s_{u,j}$, and $s_{i,j}$ is the similarity coefficient between gene i and j from matrix S of the previous step, and $k_i = \sum_j s_{ij}$ is the degree of node i. The output is a symmetric adjacency matrix $A_{m\times m}$ with values in $[0,1]$ where m is the number of genes. Note that the diagonal elements of A are zero.

3.4.2 Step II: Network clustering

Eigenvalues and Eigenvectors. Let A be the adjacency matrix from the previous step. Let D be the diagonal matrix containing the degrees of the nodes in the similarity matrix, i.e., $d_{ii} = \sum_j a_{ij}$. We perform the following steps:

- Compute the eigenvalues and the corresponding eigenvectors of $D^{-1}A$. Let $\lambda_1, \ldots, \lambda_m$ be the eigenvalues, and $Y_1, \ldots, Y_m$ be the corresponding eigenvectors [2].
- For eigenvector Y_i defines the scalar $a_i = \mathbf{1}^T DY_i/m$, where $\mathbf{1}$ is the all-ones vector. Then subtract t_i from each entry of Y_i.
- Let $Y_i := Y_i/(Y_i^T DY_i)$.
- Drop the first column of Y.

The output of this step consists of the eigenvalues $\lambda_1, \ldots, \lambda_m$, and of the matrix of eigenvectors $Y_{m-1\times m}$, where eigenvector Y_i is the i^{th} column of Y.

Determining the Number of Clusters. Three potentially different values for the number of clusters are calculated, kag, krg, ksg using, respectively what we call the *additive gap*, *relative gap*, and the *second-order gap* methods. These are calculated as follows:

$$kag = \arg\max_i \left(\lambda_{i+1} - \lambda_i\right) \qquad \text{for} \quad i = 2, \ldots, m-1 \tag{3.4}$$

$$krg = \arg\max_i \left(\frac{1-\lambda_{i+1}}{1-\lambda_i}\right) \qquad \text{for} \quad i = 2, \ldots, m-2 \tag{3.5}$$

$$ksg = \arg\max_i \left(\frac{1-\lambda_{i+1}}{1-\lambda_i} - \frac{1-\lambda_{i+2}}{1-\lambda_{i+1}}\right) \qquad \text{for} \quad i = 2, \ldots, m \tag{3.6}$$

[2] The number of eigenvectors that are practically needed for the rest of the pipeline never exceeds $m' = 50$. With an appropriate method, one can calculate at most m' eigenvectors, resulting in a faster method.

Calculation of Conductance Index. For each of the three possible values of k (i.e., kag, krg, ksg), We set Y' to consist of the $2k$ columns (i.e., eigenvectors) of Y. Each row in Y' is then divided by its Euclidean norm so that length of each row becomes 1. Next, the kmeans clustering algorithm [39] is applied on Y' to find k clusters using the default kmeans() R function. By default, the maximum number of iterations is set to 10^8 and the number of starts is set to 1000. Then, for each cluster, the conductance index is computed. Let C_i be one of the clusters. The conductance index for cluster C_i is defined in Equation (3.7).

$$conduct(C_i) = \frac{\sum_{u \in C_i, v \notin C_i} a_{u,v}}{\sum_{u \in C_i} deg(u)} \tag{3.7}$$

where $deg(u) = \sum_j A_{u,j}$ which indicates the degree node u (sum of all the weights associated to node u), and $a_{u,v}$ is the pairwise association between node u and v in adjacency matrix A. For each method, the cluster that has the minimum conductance index is chosen and passed to the next level. Let c_{ag}, c_{rg}, and c_{sg} denote the clusters with minimum conductance index for the three aforementioned methods, respectively.

Gene Ontology Validation. In this step, the enrichment of clusters c_{ag}, c_{rg}, and c_{sg} are calculated using the GOstats [32] R package individually for all six possible queries (“underBP”, “overBP”, “underCC”, “overCC”, “underMF”, “overMF‘”) combined. To this end, a conditional “hyperGTest” test is performed and the entire set of genes in the data is considered for the “universeGeneIds”. For each cluster $c \in \{c_{ag}, c_{rg}, c_{sg}\}$, GOstats returns the GO terms found in c along with a *p-value* for each term. Let P_i denote the *p-value* associated with a GO term i found in c. Then the quality of a cluster c is determined by:

$$\sum_{j \in c} -\log_{10}(P_j). \tag{3.8}$$

This measure is then used to pick the cluster of best quality among $\{c_{ag}, c_{rg}, c_{sg}\}$. Each of these three clusters was produced by kmeans with a specific choice of k: kag, krg, ksg, respectively. Then the cluster of best quality directly determines what value of k will be used. For example, if c_{ag} is the best cluster, then $k = kag$. After determining k, the clusters computed earlier by kmeans for that value of k are returned as output, along with embedding matrix $Y'_{m\times 2k}$.

3.4.3 Gene ontology enrichment

The GOstats R package [32] is applied to each cluster returned in the GO Validation step. The settings of GOstats are the same as in the GO validation step. GOstats reports answers on user-specified queries including “id”, “term”, “p-value”, “odds” “ratio”, “expected count”, “count”, and “size”. SGCP reports this information for each cluster separately. Additionally, for each cluster SGCP reports the GO terms that have been found in the cluster.

3.4.4 Gene Semi-labeling

In the default setting, SGCP picks the top 10% GO terms according to their associated *p-values*, and consider their corresponding genes as *remarkable*. All other genes are considered *unremarkable*. That percentage is user-adjustable.

With this definition, some clusters may not contain any remarkable genes. Then, each remarkable gene inherits the label of its parent cluster. The unremarkable genes remain unlabeled.

3.4.5 Semi-supervised Classification

Labeled and unlabeled gene sets along with their corresponding $2k$-dimensional points given by the rows of Y' (obtained in the Network clustering step) define a semi-supervised classification problem. We adopt a simple solution that uses the embeddings of the labeled genes as training points, and we train a simple classifier such as *k-nearest neighbors* (kNN) [40] or logistic regression [85]. Then the trained classifier is used to classify the unlabeled points and their corresponding genes. Note that k is the number of clusters determined in the Network Clustering step, but the actual number of clusters returned in this step is equal to the number of clusters found to contain remarkable genes in the *Gene Semi-labeling* step. The default model is kNN and the number of neighbors is ranging from $20 : (20 + 2 * k)$ if $2 * k \leq 30$ otherwise $20 : 30$ depending on accuracy metric using [53] R-package.

3.4.6 Final remark

The proposed pipeline is flexible. In particular, SGCP enables the user to define their preferred number f clusters k. For example, we have found that in the case of dataset GSE44903, SGCP outperforms the baselines significantly with a different value of k that is not automatically produced by our pipeline. SGCP also includes a user-defined threshold about the percentage of GO terms used for finding remarkable genes and clusters.

3.4.7 Settings in baseline pipelines

As discussed earlier, all the baseline pipelines use soft-powering (sft) to make the GCNs scale-free. We use the same soft-power methods across all pipelines and the specific powers used for each dataset are reported in Table 3.1. The functions that are used for GCN construction and analysis in WGCNA, CoExpNets, and CEMiToo, are “blockwiseModules”, “getDownstreamNetwork” and “cemitool”, respectively.

APPENDIX A

SIMILARITY FUNCTIONS

In this appendix, you find mathematical and statistical information on the similarity functions used in the network construction phase of gene co-expression networks, as described in Subsection 1.3.1. The similarity functions for two vectors of $X =< x_1, \cdots, x_n >$ and $Y =< y_1, \cdots, y_n >$ are described in this section. Let $R =< r_1, \cdots, r_n >$ and $Q =< q_1, \cdots, q_n >$ are the rank vectors of X and Y respectively.

A.1 Assumptions

Let A and B be vectors of random variables $a_1, \cdots, a_n$ and $b_1, \cdots, b_n$. Throughout this appendix, we make the following assumptions.

- $var(A)$ is the variance of random variable A.
- $cov(A, B)$ indicates the covariance value between random variable A and B.
- $cor(A, B)$ indicates the Pearson correlation value between random variable A and B.
- $\bar{A}$ and $\hat{\sigma}_A$ denote the sample mean and sample standard deviation of A.
- The rank of A is a random variable C where $c_1, \cdots, c_n$ is the non-decreasing permutation of the $a_1, \cdots, a_n$.
- $\mathbb{E}[A]$ is the expectation value of A.
- $med(A)$ denote the median of A.
- $mad(A)$ is the median absolute deviation of A.
- $< A, B >$ is the scalar product between A and B.
- $I(b)$ is the indicator that takes 1 value if $b > 0$ and 0 otherwise.
- $Entropy(A)$ is the entropy of A.
- $det(M)$ denotes the determinant of matrix M.
- $\Sigma_{A,B} = \begin{pmatrix} \Sigma_{1,1} & \Sigma_{1,2} \\ \Sigma_{2,1} & \Sigma_{2,2} \end{pmatrix}$ is the covariance matrix between variables A and B such that $\Sigma_{1,1} = var(A), \Sigma_{2,2} = var(B), \Sigma_{1,2} = \Sigma_{2,1} = cov(A, B)$.

A.2 Pearson Correlation

Equation (A.1) denotes the Pearson Correlation and its coefficients are always $-1 \leq \rho_{X,Y} \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. This is a parametric association method and it captures the linear dependencies.

$$\rho_{X,Y} = \frac{cov(X,Y)}{\hat{\sigma}_X \hat{\sigma}_Y} \tag{A.1}$$

The Pearson Correlation assumes the data is normally distributed and there is a linear relationship between the two variables. This association method is sensitive to outliers and has great power [55].

A.3 Spearman (Rank) Correlation

Equation (A.2) denotes the Spearman (Rank) Correlation and its coefficient is always $-1 \leq \rho_{R,Q} \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. This is a non-parametric association method and it captures the non-linear monotonic dependencies.

$$\begin{aligned} \rho_{R,Q} &= \frac{cov(R,Q)}{\sigma_R \sigma_Q} \\ &= \frac{6\sum_{i=1}^{n} d^2}{n(n^2-1)} \end{aligned} \tag{A.2}$$

Where d_i is the difference between ranks of the i^{th} observations of the two variables. Spearman (Rank) Correlation does not require the data to be measured on interval or ratio scale and it has less power than Pearson Correlation [55, 66].

A.4 Kendall (Rank) Correlation

Equation (A.3) denotes the Kendall (Rank) Correlation and its coefficient is always $-1 \leq \tau \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. This is a non-parametric association method and it captures non-linear monotonic dependencies. Kendall (Rank) Correlation has less power than Pearson Correlation.

$$\tau = \frac{n_c - n_d}{\binom{n}{2}} \tag{A.3}$$

Where n_c and n_d indicate the number of concordant and discordant pairs, respectively. For any pair of (x_i, y_i) and (x_j, y_j) where $1 \leq i, j \leq n$ are to be said concordant if both $x_i > x_j$ and $y_i > y_j$ or if both $x_i < x_j$ and $y_i < y_j$ and they are said to be discordant of $x_i > x_j$ and $y_i < y_j$ or $x_i < x_j$ and $y_i > y_j$ [55, 66].

A.5 Biweight Midcorrelation

Equation (A.4) denotes the biweight Midcorrelation and its coefficient $-1 \leq bicor_{X,Y} \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. This is a parametric association method and it captures the linear dependencies.

$$bicor_{X,Y} = \frac{\sum_{i=1}^{n}(x_i - med(X))w_i{}^{(X)}(y_i - med(Y))w_i{}^{(Y)}}{\sqrt{\sum_{j=1}^{n}[(x_j - med(X))w_j{}^{(X)}]^2}\sqrt{\sum_{k=1}^{n}[(y_k - med(Y))w_k{}^{(Y)}]^2}}$$

$$\begin{aligned}
&\text{where}\\
&w_i{}^{(X)} = (1 - u_i{}^2)^2 I(1 - \mid u_i \mid)\\
&w_i{}^{(Y)} = (1 - v_i{}^2)^2 I(1 - \mid v_i \mid)\\
&\text{where}\\
&u_i = \frac{x_i - med(X)}{9mad(X)}\\
&v_i = \frac{y_i - med(X)}{9mad(Y)}
\end{aligned} \tag{A.4}$$

Biweight Midcorrelation is more robust to outlier and its power is higher than Spearman and Kendall Correlation [106, 55, 66]

A.6 Weighted Rank Correlation

Equation (A.5) denotes the Weighted Rank Correlation and its coefficient is bounded by $-1 \leq r_w \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. This method is non-parametric and captures the non-linear monotonic dependencies [55].

$$r_w = 1 - \frac{6\sum_{i=1}^{n}(r_i - q_i)^2((n - r_i + 1) + (n - q_i + 1))}{n^4 + n^3 - n^2 - n} \tag{A.5}$$

A.7 Renyi Correlation

Equation (A.6) denotes the Rényi Correlation and its coefficient $0 \leq \rho_{max\{X,Y\}} \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. It captures the non-linear monotonic dependencies

$$\rho_{max\{X,Y\}} \triangleq \sup_{\substack{\mathbb{E}[f(X)]=\mathbb{E}[g(Y)]=0 \\ \mathbb{E}[f^2(X)]=\mathbb{E}[g^2(Y)]=1}} \mathbb{E}[f(X)g(Y)] \tag{A.6}$$

If $\rho_{max\{X,Y\}} = +1$, then there exists functions such that $f(X) = g(Y)$. If X and Y are jointly Gaussian, then $\rho_{max\{X,Y\}} =\mid \rho_{X,Y} \mid$ where $\rho_{X,Y}$ is the Pearson Correlation Coefficient [97] (https://web.mit.edu/18.338/www/2016s/projects/makur_slides.pdf).

A.8 Partial Correlation

Equation (A.7) denotes the Partial Correlation and its coefficient $-1 \leq \rho_{XY.Z} \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. Formally, the Partial Correlation between X and Y given a set of n controlling variables $Z = \{z_1, z_2, \cdots, z_n\}$, written $\rho_{XY.Z}$ is the correlation between the residuals ϵ_X and ϵ_Y resulting from the linear regression of X with Z and Y with Z, respectively. The first-order Partial Correlation (i.e., when $n = 1$) is the difference between a correlation and the product of the removable correlations divided by the product of the coefficients of alienation of the removable correlations https://en.wikipedia.org/wiki/Partial_correlation.

$$
\begin{aligned}
&\rho_{XY.Z} = \frac{n\sum_{i=1}^{n} \epsilon_{x_i} \cdot \epsilon_{y_i}}{\sqrt{n\sum_{i=1}^{n} \epsilon_{x_i}^2}\sqrt{n\sum_{i=1}^{n} \epsilon_{y_i}^2}} \\
&\text{where} \\
&\quad \epsilon_{x_i} = x_i - < w_X, z_i > \\
&\quad \epsilon_{y_i} = y_i - < w_Y, z_i > \\
&\text{where} \\
&\quad w_X = \underset{w}{\arg\min} \ \sum_{i=1}^{n} (x_i - < w, z_i >)^2 \\
&\quad w_Y = \underset{w}{\arg\min} \ \sum_{i=1}^{n} (y_i - < w, z_i >)^2
\end{aligned}
\tag{A.7}
$$

Where ϵ's are the residuals after calculating the linear regression of X with Z and Y with Z.

A.9 CCor Correlation

Equation (A.8) denotes the "CCor" Correlation and its coefficient $-1 \leq CCor_{X,Y} \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. $CCor$ between genes i and j represented by X_i and Y_j, respectively is defined as the Pearson Correlation (see Appendix A.2) between correlation coefficients X_i versus genes except i, j and X_j versus genes except i, j [44].

$$
\begin{aligned}
&CCor_{X_i,Y_j} \triangleq cor(U_i, V_j) \\
&\quad \text{where} \\
&\quad\quad u_i = (x_1, x_2, \cdots, x_{i-1}, x_{i+1}, \cdots, x_{j-1}, x_{j+1}, \cdots, x_n) \\
&\quad\quad v_j = (y_1, y_2, \cdots, y_{i-1}, y_{i+1}, \cdots, y_{j-1}, y_{j+1}, \cdots, y_n)
\end{aligned}
\tag{A.8}
$$

A.10 Blomqvist Correlation

Equation (A.9) denotes the Blomqvist Correlation and its coefficient $-1 \leq \beta_{X,Y} \leq +1$ where -1 and $+1$ indicate the perfect negative and positive correlation, respectively. This is a non-parametric association method and it captures the non-linear monotonic dependencies.

Let "$x - y$"-plane be divided into four regions by the median lines of $\tilde{X}$ and $\tilde{Y}$. The relationship of X and Y can be obtained from the number of sample points in the four quadrants.

$$\beta_{X,Y} = \frac{n_1 - n_2}{n_1 + n_2} \tag{A.9}$$

Where n_1 and n_2 denote the number of data in the first or third and second or fourth quadrant, respectively [66].

A.11 Kernel Canonical Correlation Analysis

Canonical Correlation Analysis (CCA) and Kernel Canonical Correlation Analysis (KCCA) are statistical techniques used to find linear and nonlinear relationships between two sets of variables. CCA finds the linear combination of variables from each set that has the highest correlation with the linear combination of variables from the other set. KCCA extends CCA to non-linear relationships using kernel functions.

In Equation (A.10), CCA finds the maximum correlation between linear combinations of X and Y by solving for the optimal weight vectors a and b. The correlation coefficient is the correlation between the linear combinations of variables.

$$\begin{aligned} CCA_{X,Y} &= \sup_{a,b} \qquad cor(U,V) \\ &\text{where} \\ cor(U,V) &= \frac{a^T \Sigma_{1,2} b}{\sqrt{a^T \Sigma_{1,1} a}\sqrt{b^T \Sigma_{2,2} b}} \end{aligned} \tag{A.10}$$

In Equation (A.11), KCCA uses a kernel function Φ to map the variables X and Y into a higher-dimensional space before computing the linear combinations. KCCA then finds the optimal weight vectors α and β to maximize the correlation between the linear combinations of variables in the higher-dimensional space.

$$\begin{aligned} K_X &= \sum_{i=1}^{n} \Phi(x_i)^T \Phi(x_i) \\ K_Y &= \sum_{i=1}^{n} \Phi(y_i)^T \Phi(y_i) \end{aligned} \tag{A.11}$$

The Kernel Canonical Correlation Analysis (KCCA) is Equation (A.12) [66].

$$\begin{aligned} kcca_{X,Y} &= \sup_{\alpha,\beta} \qquad cor(U,V) \\ &\text{where} \\ cor(U,V) &= \frac{\alpha^T K_X K_Y \beta}{\sqrt{\alpha^T K_X K_Y \alpha}\sqrt{\alpha^T K_X K_Y \beta}} \end{aligned} \tag{A.12}$$

Both CCA and KCCA can be used for dimensionality reduction, feature extraction, and pattern recognition. CCA has been used in many fields, such as biology, finance, and image analysis, while KCCA has been used in natural language processing, speech recognition, and bioinformatics.

A.12 Distance Covariance and Correlation

Equation (A.13) denotes the squared Distance Covariance of X and Y and its coefficient is $0 \leq dCov^2_{X,Y} \leq 1$. It captures non-linear dependencies.

Let A be the pairwise Euclidean distance matrices of X with $a_{i,j} = \mid x_i - x_j \mid$ as the $(i,j)^{\text{th}}$ entry and similarly let B the pairwise Euclidean distance matrices of Y with $b_{i,j} = \mid y_i - y_j \mid$ as the $(i,j)^{\text{th}}$ entry for $i, j = 1, 2, \cdots, n$. Then A^c and B^c are derived by centralizing the matrices A and B. That is, the $(i,j)^{\text{th}}$ entry of A^c is ${a_{i,j}}^c = a_{i,j} - \bar{a_{i.}} - \bar{a_{.j}} + \bar{a_{..}}$ where $\bar{a_{i.}}$ is the i^{th} row mean, $\bar{a_{.j}}$ is the j^{th} column mean, and $\bar{a_{..}}$ is the grand mean of A. Similarly the $(i,j)^{\text{th}}$ entry of B^c is ${b_{i,j}}^c = b_{i,j} - \bar{b_{i.}} - \bar{b_{.j}} + \bar{b_{..}}$ where $\bar{b_{i.}}$ is the i^{th} row mean, $\bar{b_{.j}}$ is the j^{th} column mean, and $\bar{b_{..}}$ is the grand mean of B [55, 66].

$$dCov^2_{X,Y} = \frac{1}{n^2} \sum_{i,j} {A_{i,j}}^c {B_{i,j}}^c \tag{A.13}$$

Equation (A.14) denotes the squared Distance Correlation of X and Y and its coefficient is $0 \leq dCor^2_{X,Y} \leq 1$. It captures non-linear dependencies.

$$\begin{aligned} dCor^2_{X,Y} &= R^2 \\ &= \frac{dCov^2_{X,Y}}{dCov_X \; dCov_Y} \end{aligned} \tag{A.14}$$

A.13 Wilks' Statistic

Equation (A.15) denotes the Wilks' Statistics and its coefficient $0 \leq W_{X,Y} \leq 1$. It captures linear bivariate associations [66].

$$W_{X,Y} = 1 - \frac{det(\Sigma)}{det(\Sigma_{11})\ \ det(\Sigma_{22})} \tag{A.15}$$

A.14 Hoeffding's Dependence

Equation (A.16) denotes the Hoeffding's Dependence and its coefficient $-1 \leq D_{X,Y} \leq +1$. This is a non-parametric association method and it captures non-linear dependencies.

$$\begin{aligned} D_{X,Y} &= \frac{(n-2)(n-3)D_1 + D_2 - 2(n-2)D_3}{n(n-1)(n-2)(n-3)(n-4)} \\ &\text{where} \\ D_1 &= \sum_{i=1}^{n} s_i(s_i - 1) \\ D_2 &= \sum_{i=1}^{n}(r_i - 1)(r_i - 2)(q_i - 1)(q_i - 2) \\ D_3 &= \sum_{i=1}^{n}(r_i - 2)(q_i - 2)S \end{aligned} \tag{A.16}$$

Where S is the bivariate rank which is the number of both X and Y values less than the i^{th} point and can be calculated as $s_i = \sum_{j=1}^{n} \varnothing(x_j, x_i)\varnothing(y_j, y_i)$, where $\varnothing(a, b) = 1$ if $a < b$ and 0 otherwise. So it returns the number of bivariate observations for which $x_i < x_j$ and $y_j < y_i$ [55, 66].

A.15 Goodman and Kruskal Dependence

Equation (A.17) denotes the Goodman and Kruskal Dependence measure and its coefficient $-1 \leq \gamma_{X,Y} \leq +1$. This is a non-parametric association method and it captures the non-linear monotonic dependencies.

$$\gamma_{X,Y} = \frac{p_s - p_d}{p_s + p_d} \tag{A.17}$$

Where p_s and p_d are the probabilities that a randomly selected pair of observations will relocate in the same opposite order, respectively, when ranked by both variable [66].

A.16 Copula-Based Maximum Mean Discrepancy

A copula is a multivariate probability distribution function defined on the unit hypercube with known uniform marginals. It is popular in high-dimensional statistics for describing the relationships between variables. Specifically, the copula of two random gene variables X and Y is defined as a function Equation (A.18).

$$copula(U, V) = copula(F_X(x), F_Y(y)) \tag{A.18}$$

Where $F_X(x) = P(X \leq x), F_Y(y) = P(Y \leq y)$, and $F_{XY}(x, y) = P(X \leq x, Y \leq y)$ are the two marginal distributions and the joint distribution.

Copula-Based Maximum Mean Discrepancy (CMMD) is a copula-based kernel association measure between random variables. It extends the maximum mean discrepancy (MMD) method of measuring dependence to the copula of the joint distribution. Suppose two copula transformations have been implemented on the original variables, i.e., $U = F_1(X)$ and $V = F_2(Y)$. F_1 and F_2 are the empirical cumulative distribution functions for X and Y, respectively. CMMD defines the relationship between X and Y as

$$\begin{aligned} cmmd_{X,Y} &= mmd[F_1(X), F_2(Y)] \\ &= \frac{1}{n(n-1)} \sum_{i \neq j}^{n} K(u_i, v_j) \end{aligned} \tag{A.19}$$

Where $K(u_i, v_j) = \Phi(u_i, u_j) + \Phi(v_i, v_j) - \Phi(u_i, v_j) - \Phi(u_j, v_i)$, and Φ s a specific kernel function, e.g., Gaussian kernel.

A.17 Theil-Sen Estimator

The coefficient $\hat{\beta}_1$ of the linear regression of form $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i,\ i = 1, 2, \cdots, n$ demonstrates the relationship between X and Y. Equation (A.20) denotes the Theil-Sen Estimator which is a median of the slopes determined by all pairs of (x_i, y_i) $i = 1, 2, \cdots, n$ [55].

$$\hat{\beta}_1 = median\{m_{i,j} = \frac{y_i - y_j}{x_i - x_j} :\ x_i \neq x_j,\ 1 \leq i \leq j \leq n\} \tag{A.20}$$

A.18 Rank Theil-Sen Estimator

It is the same as the Theil-sen (see section A.17) but it uses R and Q instead of X and Y [55],.

A.19 W1 and W2 measure of count statistics

The Equation (A.21) notes measures of co-expression, $W1$ and $W2$, btained by aggregating interactions across all sub-samples of $k \leq n$. The coefficients $W1$ and $W2$ are bounded 0 and ∞, and the method is non-parametric, capturing the strength of the non-parametric association between two variables.

$$\begin{aligned} W1 = &\sum_{i=1}^{n-k+1} \{I(\varnothing(x_i, \cdots, x_{i+k-1}) = \varnothing(y_i, \cdots, y_{i+k-1})) \\ &+ I(\varnothing(x_i, \cdots, x_{i+k-1}) = \varnothing(-y_i, \cdots, -y_{i+k-1}))\} \\ W2 = &\sum_{1 \leq i_1 < \cdots < i_k \leq n} \{I(\varnothing(x_{i1}, \cdots, x_{ik}) = \varnothing(y_{i1}, \cdots, y_{ik})) \\ &+ I(\varnothing(x_{i1}, \cdots, x_{ik}) = \varnothing(-y_{i1}, \cdots, -y_{ik}))\} \end{aligned} \tag{A.21}$$

Here $\varnothing$ is the rank function, which returns a vector of indices of the sorted elements. $W1$ counts the number of contiguous subsequences of length k with matching and reverse rank patterns, indicating positive and negative associations, respectively. $W2$ is equal to the number of increasing (and decreasing) subsequences of length k in a suitably permuted sequence [118, 66].

A.20 Randomized Dependence Coefficient

Based on the previous kernel CCA and copulas, (see section A.11 and section A.16), the randomized dependence coefficient (RDC) provides computationally efficient association measures between multivariate random variables. In detail, it is defined as

$$rdc(X, Y, k, s) = \sup_{\alpha,\beta} \quad cor\{\alpha^T \Phi[F_1(X); k, s], \beta^T \Phi[F_2(Y); k, s]\} \tag{A.22}$$

Where the functions are the same as the former ones, $k \in \mathbb{N}^+$ and $s \in \mathbb{R}^+$ are the parameters which are often set as 20 and .6, respectively. RDC is proven to be capable of discovering a wide range f functional association patterns in multiple data sets [66].

A.21 Mutual Information Association

Let X and Y correspond to a discrete or categorical random variables. Each entry x_i $i = 1, 2, \cdots, n$ equals to one of the predefined bin_X. The mutual information (MI) coefficient is defined as Equation (A.23) [106].

$$MI_{X,Y} = \sum_{i=1}^{bin_X} \sum_{j=1}^{bin_Y} p(NX_i, NY_j) \log(\frac{p(NX_i, NY_j)}{p(NX_i)p(NY_j)}) \tag{A.23}$$

Where $p(NX_i)$ and $p(NX_j)$ are the frequency of level i and j of X and Y, respectively, and log is the natural logarithm. $MI_{X,Y}$ is always non-negative and it captures non-linear dependencies. it has an upper bound given by Equation (A.24). MI is the same as the Kullback-Leibler divergence [54, 106].

$$\begin{aligned} MI_{X,Y} &\leq \max\{Entropy(X), Entropy(Y)\} \\ MI_{X,Y} &\leq \frac{Entropy(X) = Entropy(Y)}{2} \\ MI_{X,Y} &\leq \min\{Entropy(X), Entropy(Y)\} \end{aligned} \tag{A.24}$$

A.22 CLR Coefficient

Equation (A.25) denotes the CLR coefficient which is a modification of MI [31].

$$
\begin{aligned}
z_{i,j} &= \sqrt{z_i^2 + z_j^2} \\
&\text{where} \\
z_i &= \max(0, \frac{MI(x_i, y_j) - \bar{X}_i}{\hat{\sigma}_{X_i}})
\end{aligned}
\tag{A.25}
$$

Where $MI(x_i, y_j)$ is the MI coefficient (see Section A.21).

A.23 ARACNE Algorithm

Equation (A.26) hows how the ARACNE algorithm utilizes the data processing inequality from theoretical information theory. For any three random variables X, Y, and Z, if X and Y interact only through Z, and if the Equation (A.26) holds for these three random variables, then the association between X and Y will be 0. In other words, if for the three vectors of gene expression X, Y, and Z the Equation (A.26) holds, then the edge between X and Y will be removed [74].

$$
MI(X, Y) \leq \min(MI(X, Z), MI(Z, Y)) - \epsilon \tag{A.26}
$$

A.24 Maximal Information Coefficient

Equation (A.27) denotes the Maximal Information Coefficient and its coefficient is given by $0 \leq MIC(D) \leq 1$. It can capture non-linear dependencies. Let D be a set of ordered pairs of X and Y. The x_i's and the y_i's of D can be partitioned into x and y bins, respectively. Such a pair of partitions is called an *x-by-y gird*. Given a gird G, let $D \mid G$ be the distribution induced by the points in D on the cells of G; that is, the distribution on the cells of G obtained by letting the probability mass in each cell be the fraction of points in D falling in that cell.

$$
\begin{aligned}
MIC(D) &= \max_{xy<B(n)} \{M(D)_{x,y}\} \\
&\text{where} \\
M(D)_{x,y} &= \frac{I^*(D, x, y)}{\log \min\{x, y\}} \\
&\text{where} \\
I^*(D, x, y) &= \max I(D \mid G)
\end{aligned}
\tag{A.27}
$$

Where $\omega(1) < B(n) < O(n^{1-\epsilon})$ for some $0 < \epsilon < 1$. In practice, $B(n) = n^{0.6}$ [93, 66].

APPENDIX B

PRINCIPAL COMPONENT ANALYSIS OVER THE 12 DATASETS

This appendix provides information on PCA applied to the 12 datasets discussed in Section 3.2. Each section is named after a dataset, and every figure displays the PCA of the data. The color-coding of points in the plots reflects the cluster assignment produced by the pipelines in the benchmark (as described in Section 3.2), with the pipeline name serving as the title of the plot. The SGCP clusters and modules denote the initial clusters and final modules, respectively. The two plots on the bottom row depict the PCA over the spectral embedding of the data. The table in the lower right corner presents the pipelines and their corresponding number of clusters.

B.1 PCA of GSE181225

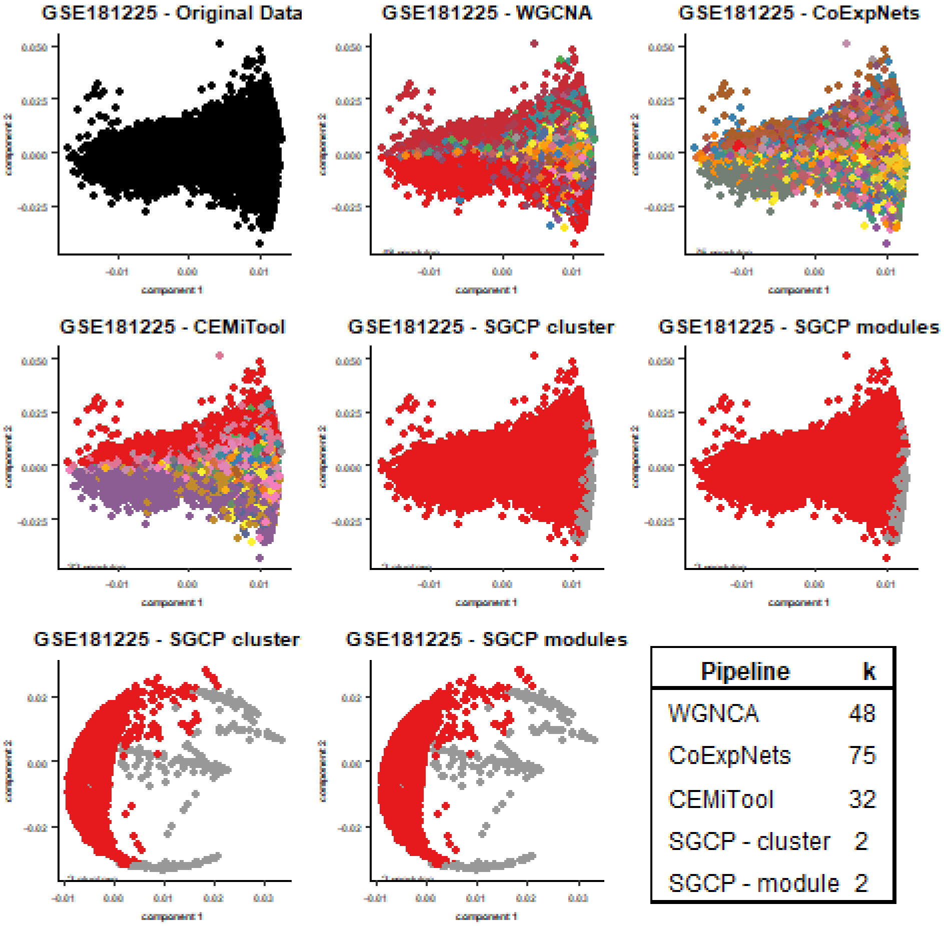
Figure B.1 displays the results of applying PCA to GSE181225 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

B.2 PCA of GSE33779

Figure B.2 displays the results of applying PCA to GSE33779 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

B.3 PCA of GSE44903

Figure B.3 displays the results of applying PCA to GSE44903 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is



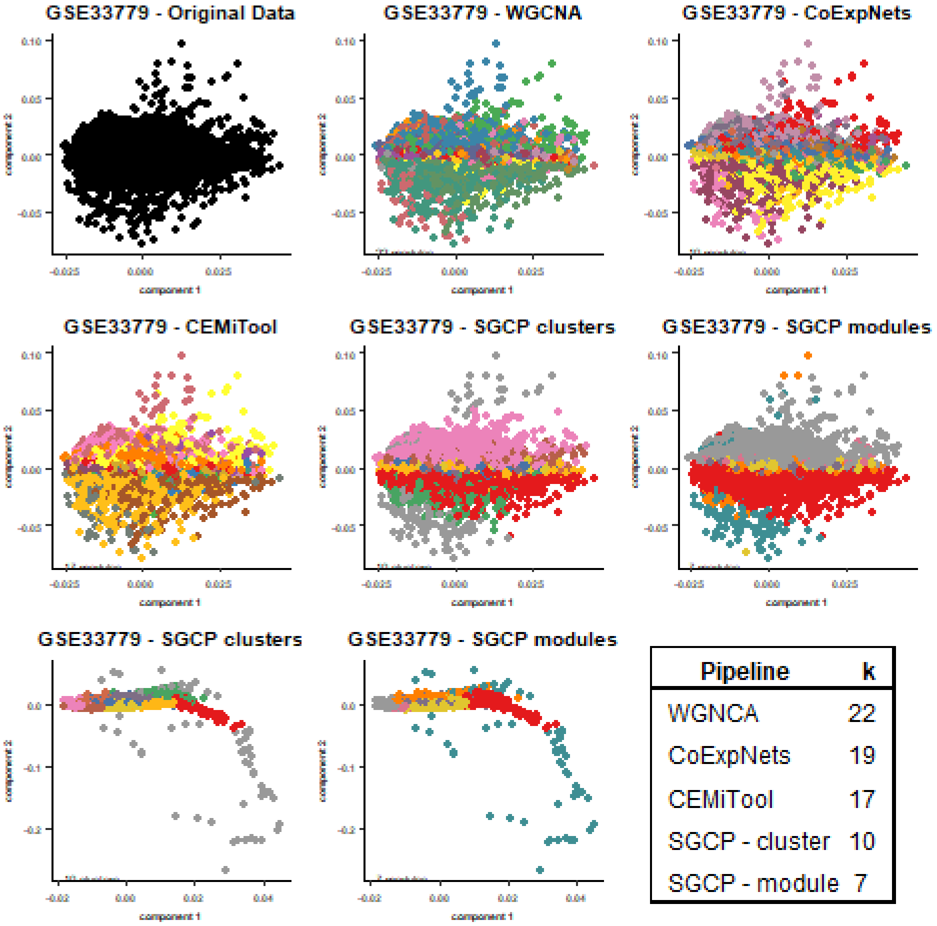
Pipeline	k
WGNCA	48
CoExpNets	75
CEMiTool	32
SGCP - cluster	2
SGCP - module	2

Figure B.1 PCA of GSE181225 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

B.4 PCA of GSE54456

Figure B.4 displays the results of applying PCA to GSE54456 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA



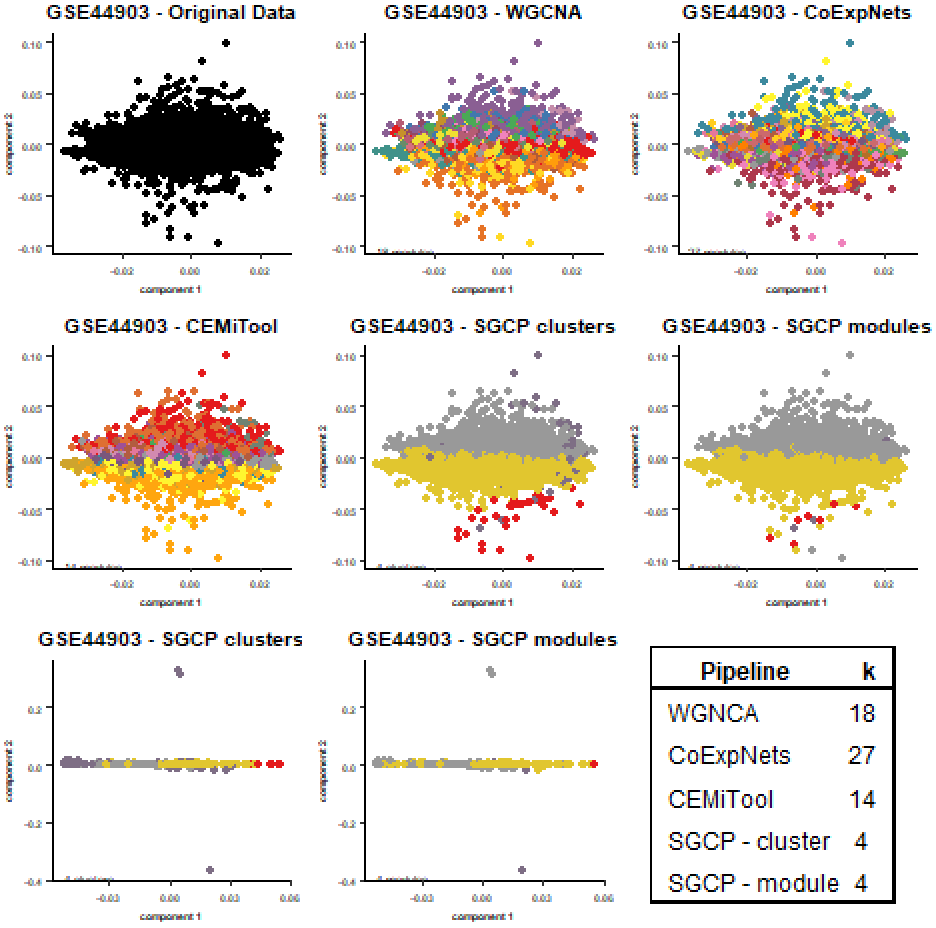
Pipeline	k
WGNCA	22
CoExpNets	19
CEMiTool	17
SGCP - cluster	10
SGCP - module	7

Figure B.2 PCA of GSE33779 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

B.5 PCA of GSE57148

Figure B.5 displays the results of applying PCA to GSE57148 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

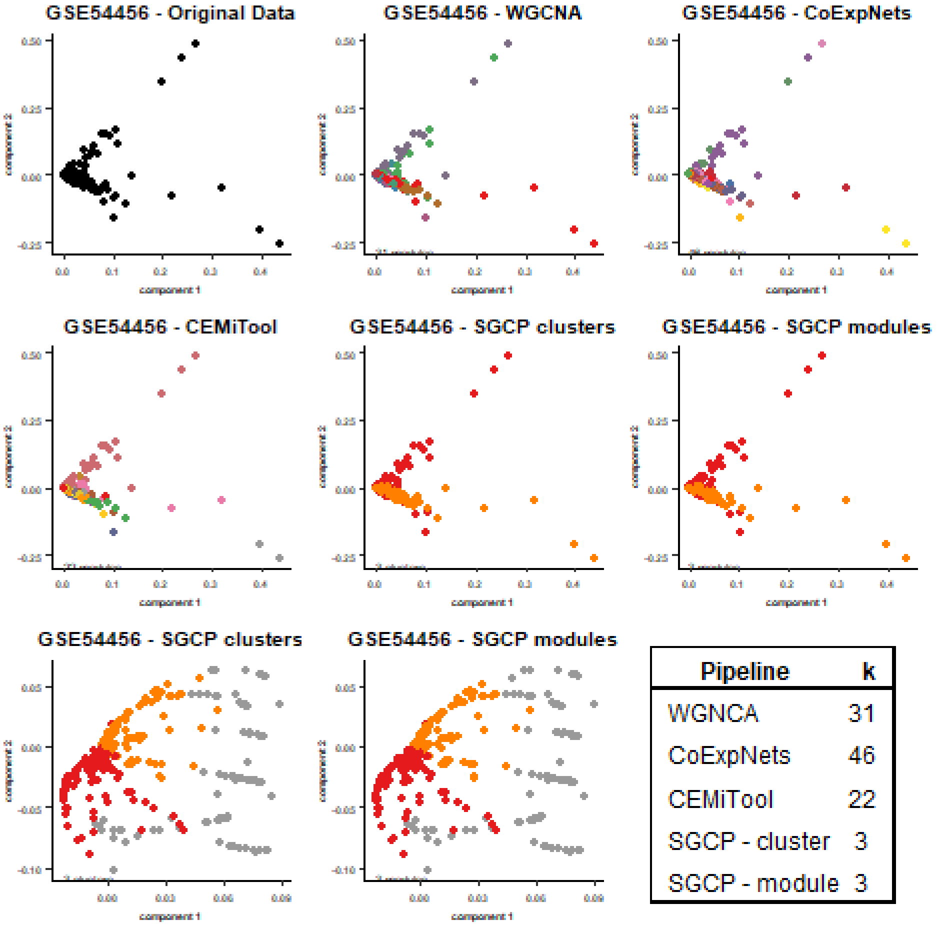


Pipeline	k
WGNCA	18
CoExpNets	27
CEMiTool	14
SGCP - cluster	4
SGCP - module	4

Figure B.3 PCA of GSE44903 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

B.6 PCA of GSE60571

Figure B.6 displays the results of applying PCA to GSE60571 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

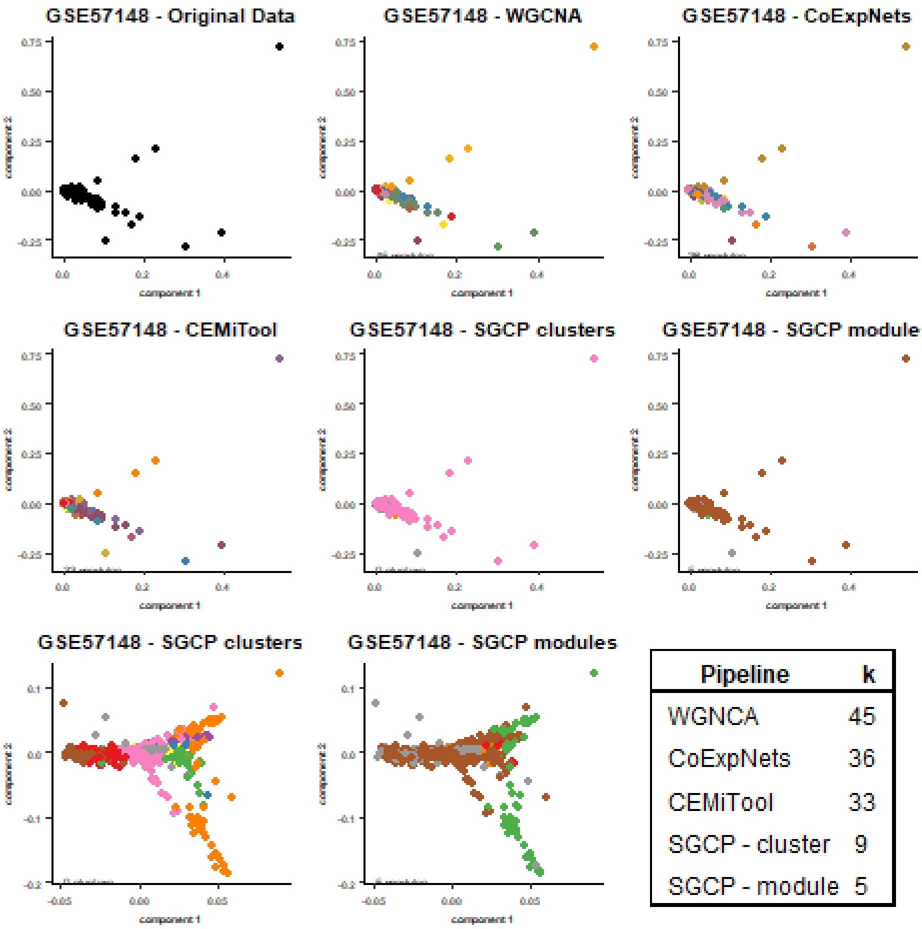


Pipeline	k
WGNCA	31
CoExpNets	46
CEMiTool	22
SGCP - cluster	3
SGCP - module	3

Figure B.4 PCA of GSE54456 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

B.7 PCA of GSE107559

Figure B.7 displays the results of applying PCA to GSE107559 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

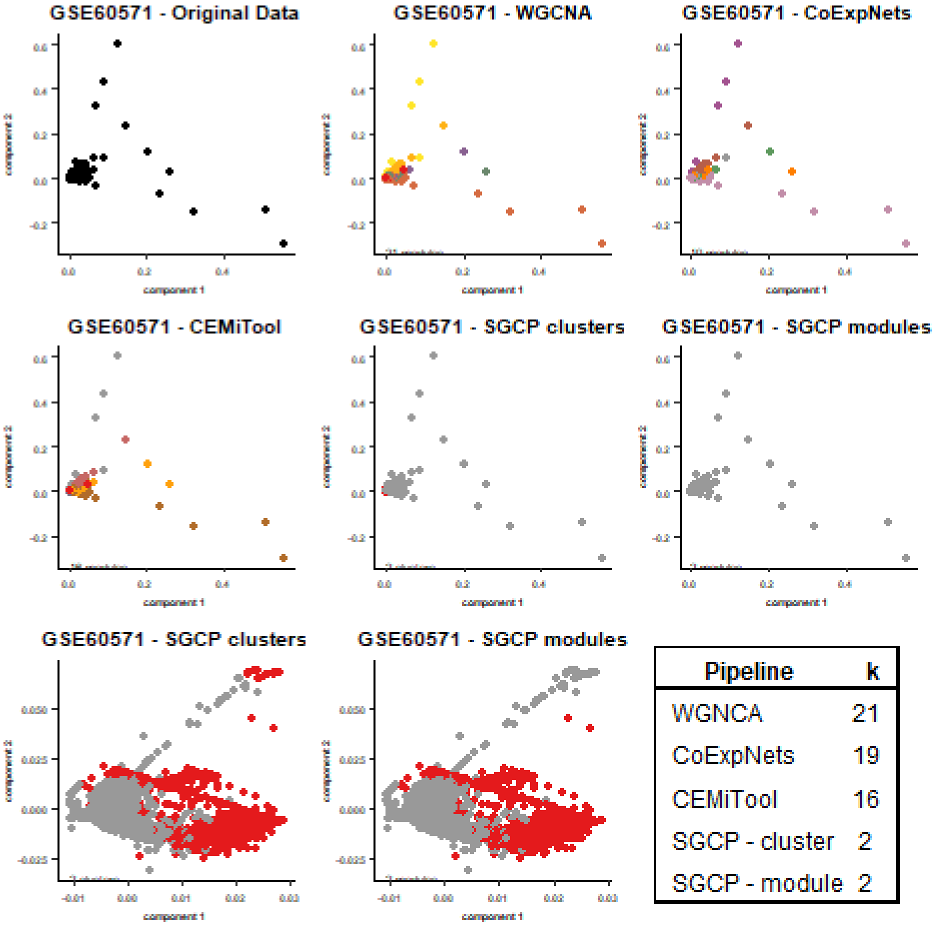


Pipeline	k
WGNCA	45
CoExpNets	36
CEMiTool	33
SGCP - cluster	9
SGCP - module	5

Figure B.5 PCA of GSE57148 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

B.8 PCA of GSE28435

Figure B.8 displays the results of applying PCA to GSE28435 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

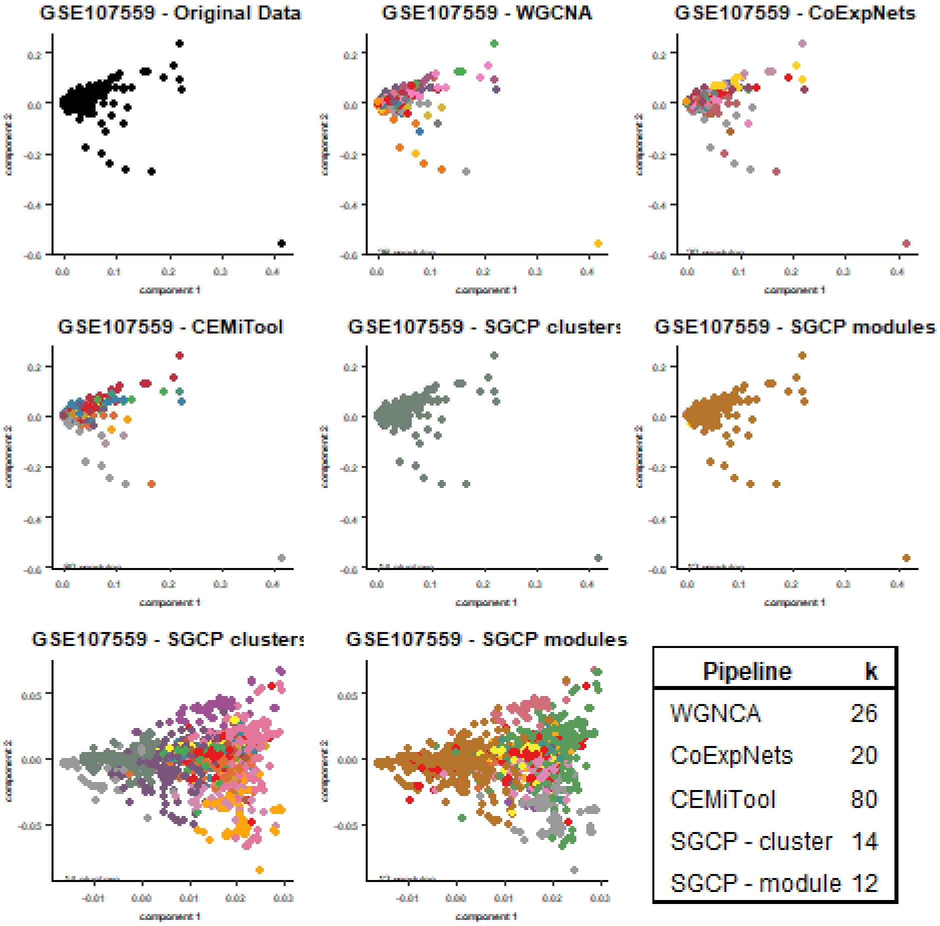


Pipeline	k
WGNCA	21
CoExpNets	19
CEMiTool	16
SGCP - cluster	2
SGCP - module	2

Figure B.6 PCA of GSE60571 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

B.9 PCA of GSE104687

Figure B.9 displays the results of applying PCA to GSE104687 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

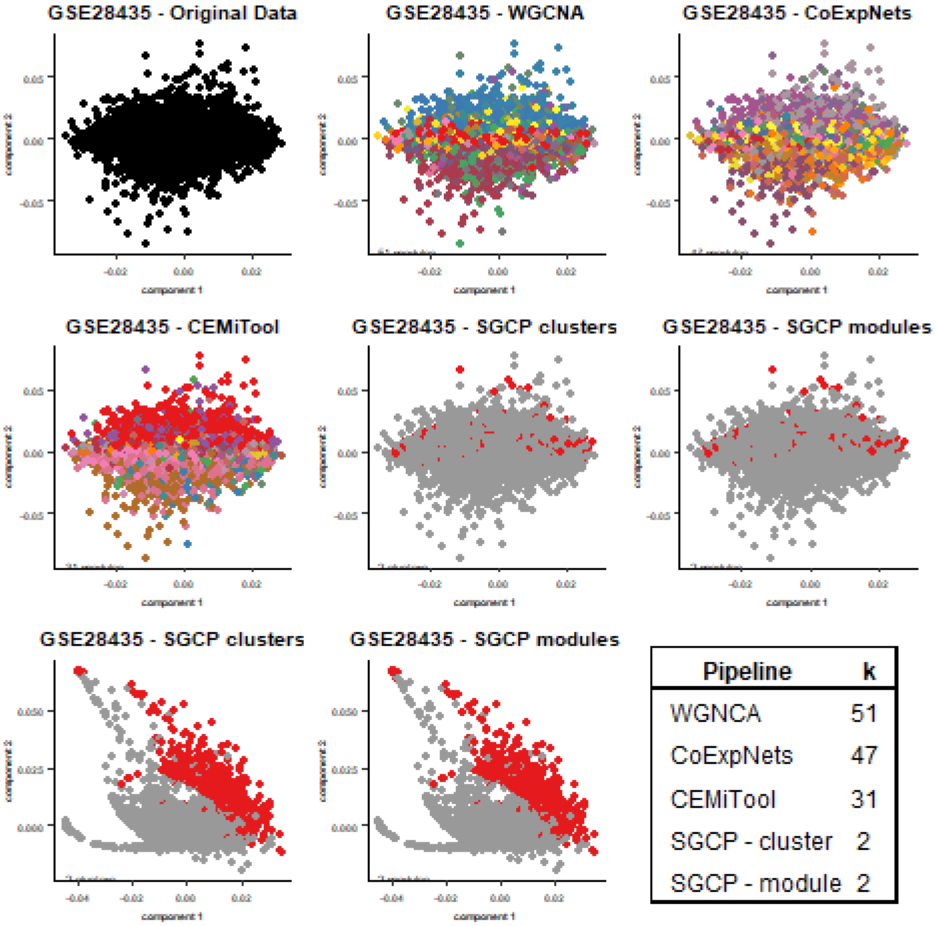


Pipeline	k
WGNCA	26
CoExpNets	20
CEMiTool	80
SGCP - cluster	14
SGCP - module	12

Figure B.7 PCA of GSE107559 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

B.10 PCA of GSE150961

Figure B.10 displays the results of applying PCA to GSE150961 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

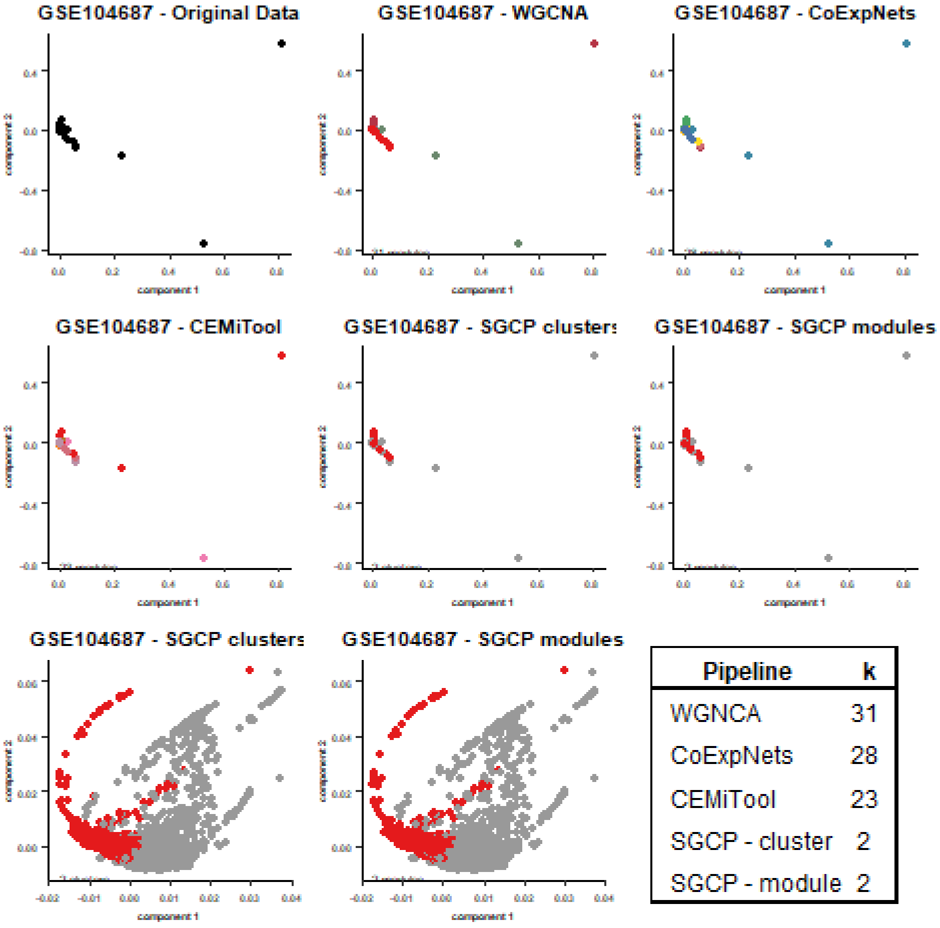


Pipeline	k
WGNCA	51
CoExpNets	47
CEMiTool	31
SGCP - cluster	2
SGCP - module	2

Figure B.8 PCA of GSE28435 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

B.11 PCA of GSE115828

Figure B.11 displays the results of applying PCA to GSE115828 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.

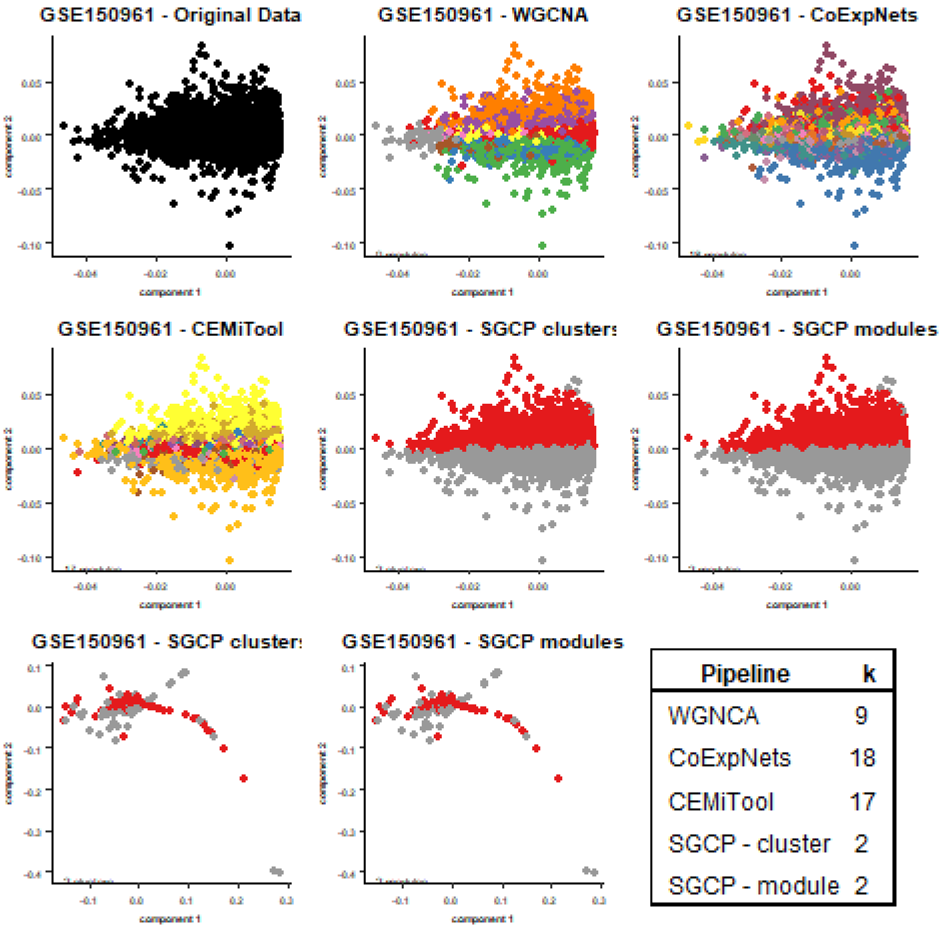


Pipeline	k
WGNCA	31
CoExpNets	28
CEMiTool	23
SGCP - cluster	2
SGCP - module	2

Figure B.9 PCA of GSE104687 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

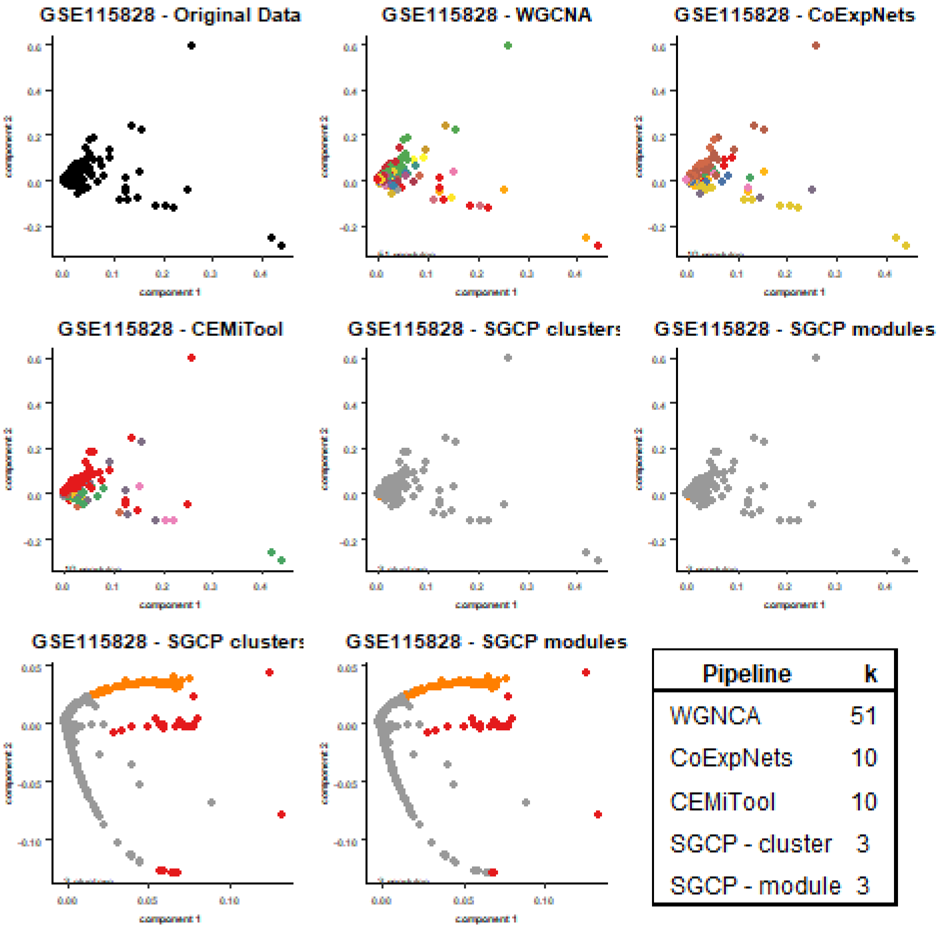
B.12 PCA of GSE38705

Figure B.12 displays the results of applying PCA to GSE38705 using four different pipelines: WGCNA, CoExpNets, CEMiTool, and SGCP. Each point in the plot is color-coded based on the module label assigned by each pipeline, and the pipeline name is displayed in the plot title. The two plots in the bottom row show the PCA over the spectral embedding of the data, and the table in the lower right corner lists the number of clusters generated by each pipeline.



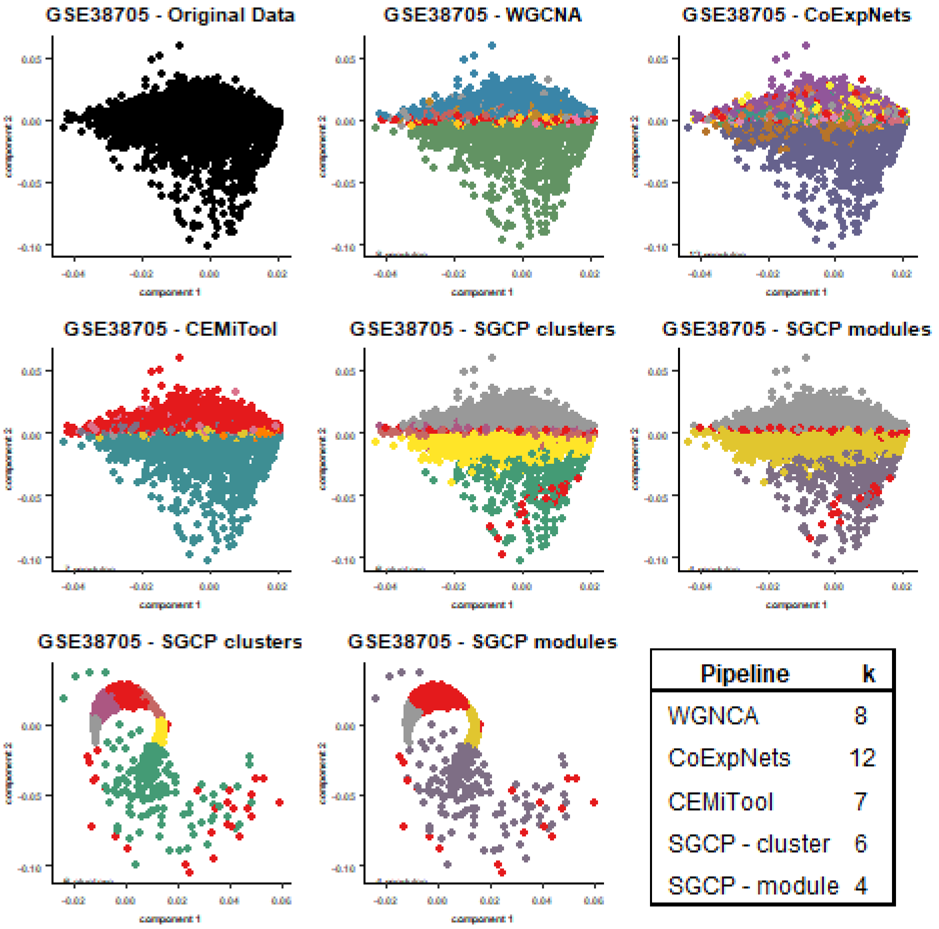
Pipeline	k
WGNCA	9
CoExpNets	18
CEMiTool	17
SGCP - cluster	2
SGCP - module	2

Figure B.10 PCA of GSE150961 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.



Pipeline	k
WGNCA	51
CoExpNets	10
CEMiTool	10
SGCP - cluster	3
SGCP - module	3

Figure B.11 PCA of GSE115828 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.



Pipeline	k
WGNCA	8
CoExpNets	12
CEMiTool	7
SGCP - cluster	6
SGCP - module	4

Figure B.12 PCA of GSE38705 for WGCNA, CoExpNets, CEMiTool, and SGCP. Colors denote the module labels defined by the pipeline. SGCP clusters and SGCP modules represent the initial clusters and final labels, respectively. The two plots on the bottom row show the PCA over the spectral embedding of the data. The table in the bottom right denotes the number of clusters per pipeline.

APPENDIX C

SIGNIFICANCE OF GENE ONTOLOGY TERM PER PIPELINE

This appendix provides information on the gene ontology (GO) terms identified by five different pipelines: WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP, as discussed in Section 3.2. The significance of the GO terms is determined by their corresponding p-value, which is log-transformed and displayed as a jitter plot. The x-axis and y-axis represent the cluster label and the log-transformed p-value, respectively. Higher values on the y-axis indicate greater significance. The points are color-coded based on their cluster assignment. Each plot corresponds to the result of a single pipeline, which is indicated by the plot title. The labels ”pSGCP” and ”SGCP” respectively denote the initial clusters and final modules.

C.1 GO terms of GSE181225

Figure C.1 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.2 GO terms of GSE33779

Figure C.2 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.3 GO terms of GSE44903

Figure C.3 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

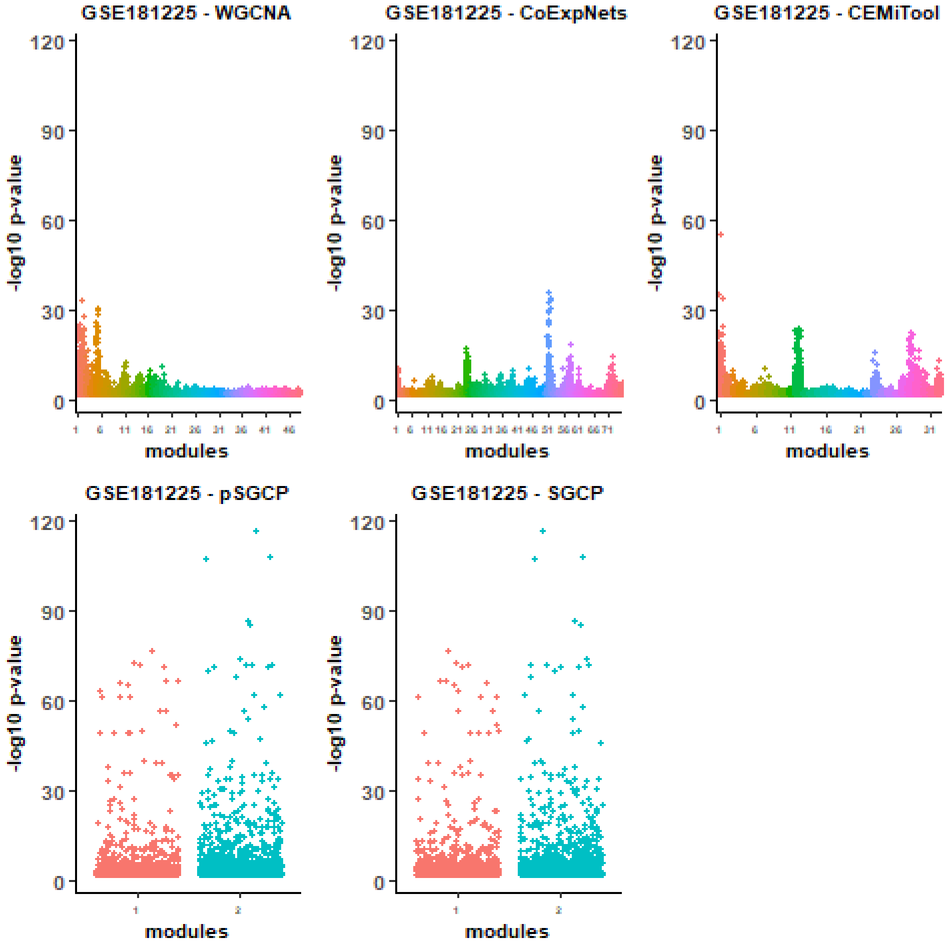


Figure C.1 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE181225 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

C.4 GO terms of GSE54456

Figure C.4 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.5 GO terms of GSE57148

Figure C.5 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by

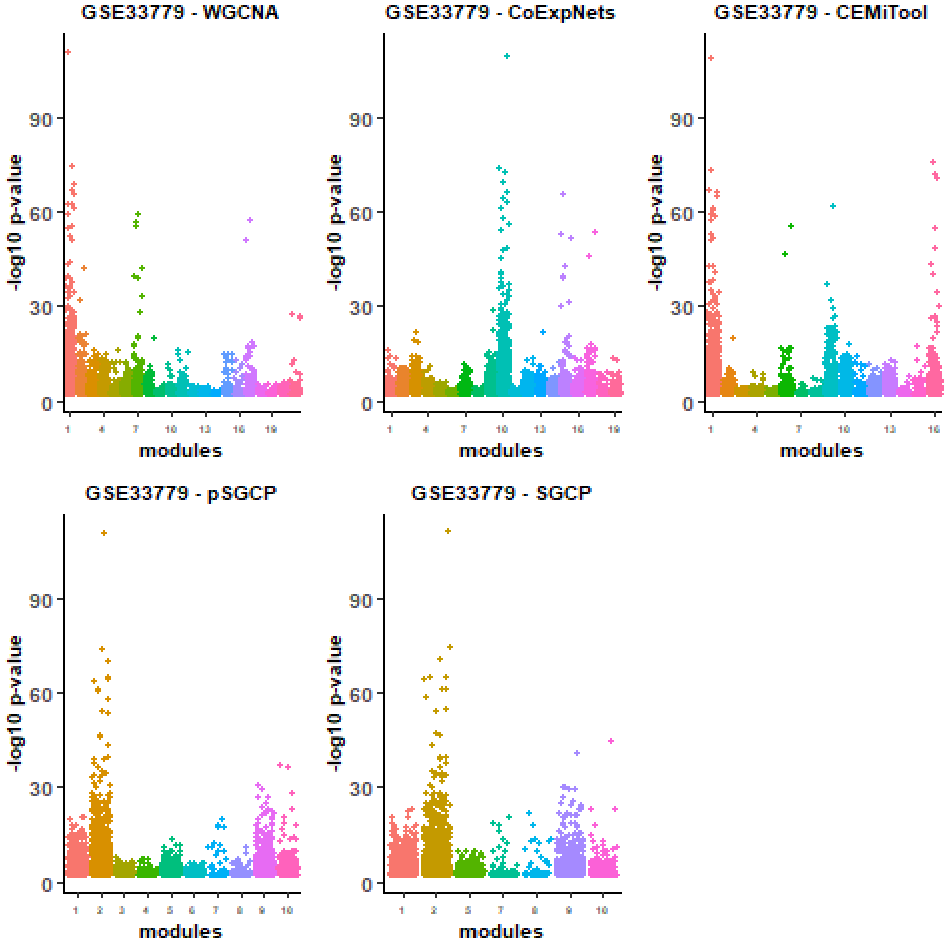


Figure C.2 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE33779 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.6 GO terms of GSE60571

Figure C.6 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

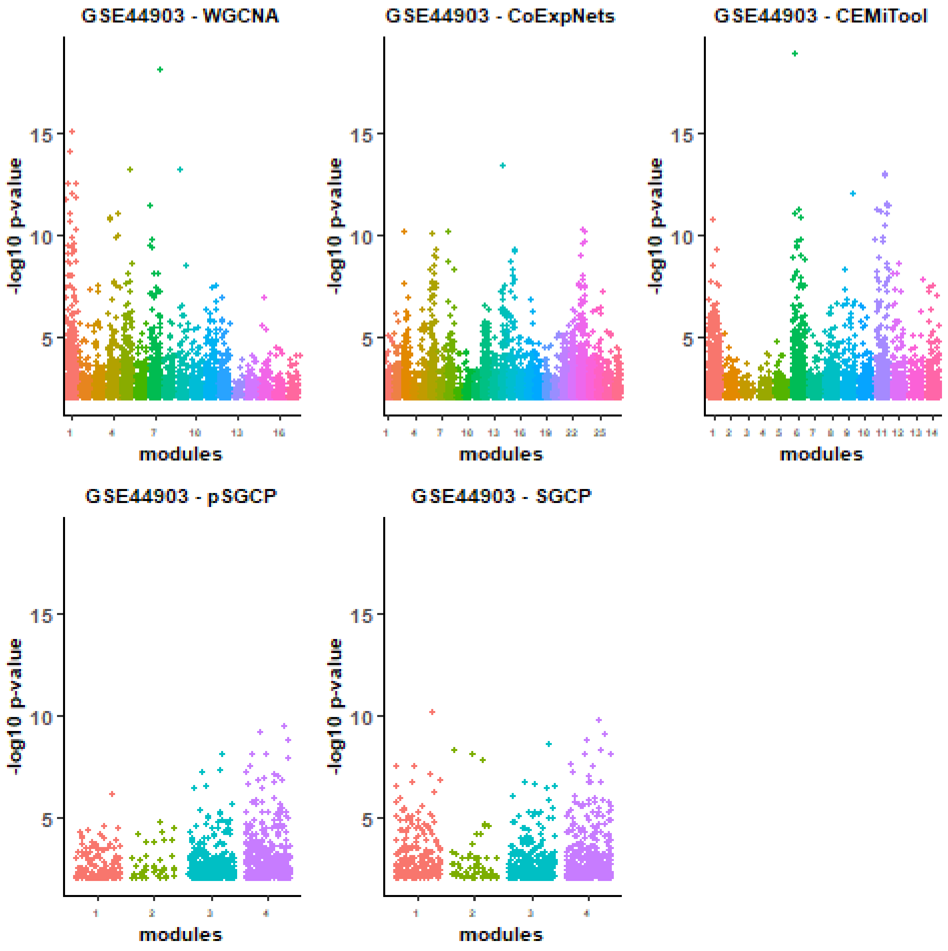


Figure C.3 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE44903 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

C.7 GO terms of GSE107559

Figure C.7 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.8 GO terms of GSE28435

Figure C.8 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by

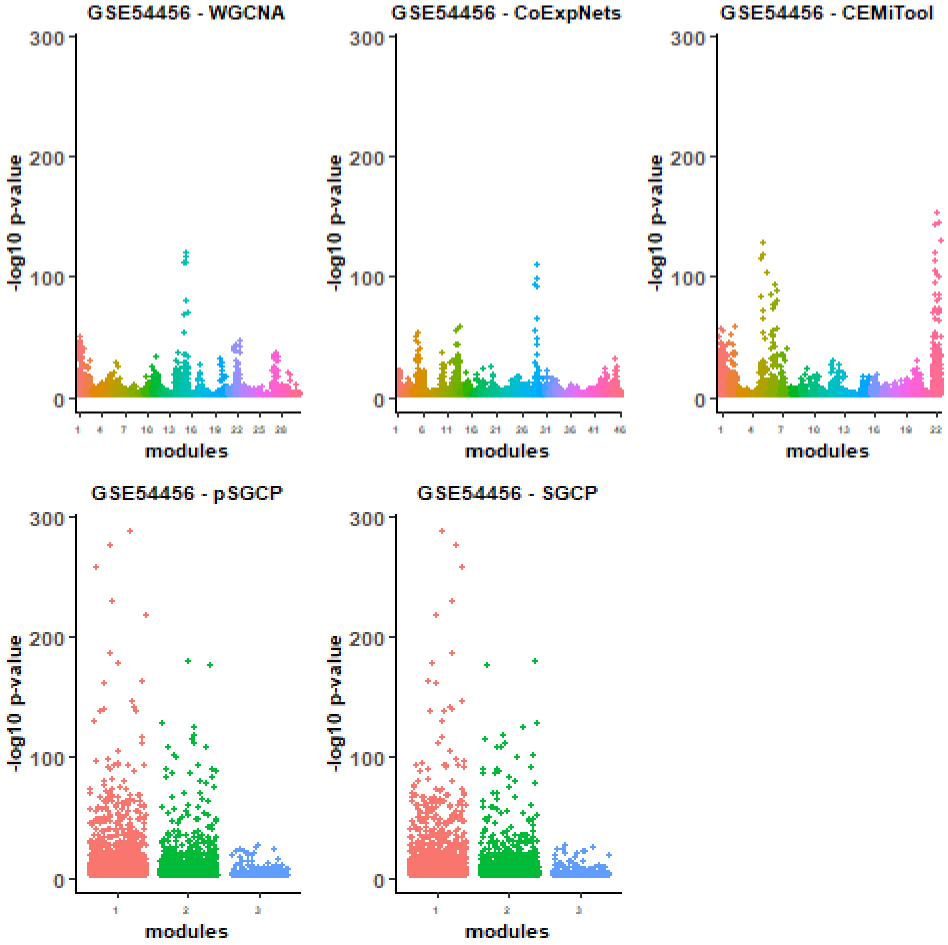


Figure C.4 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE54456 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.9 GO terms of GSE104687

Figure C.9 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

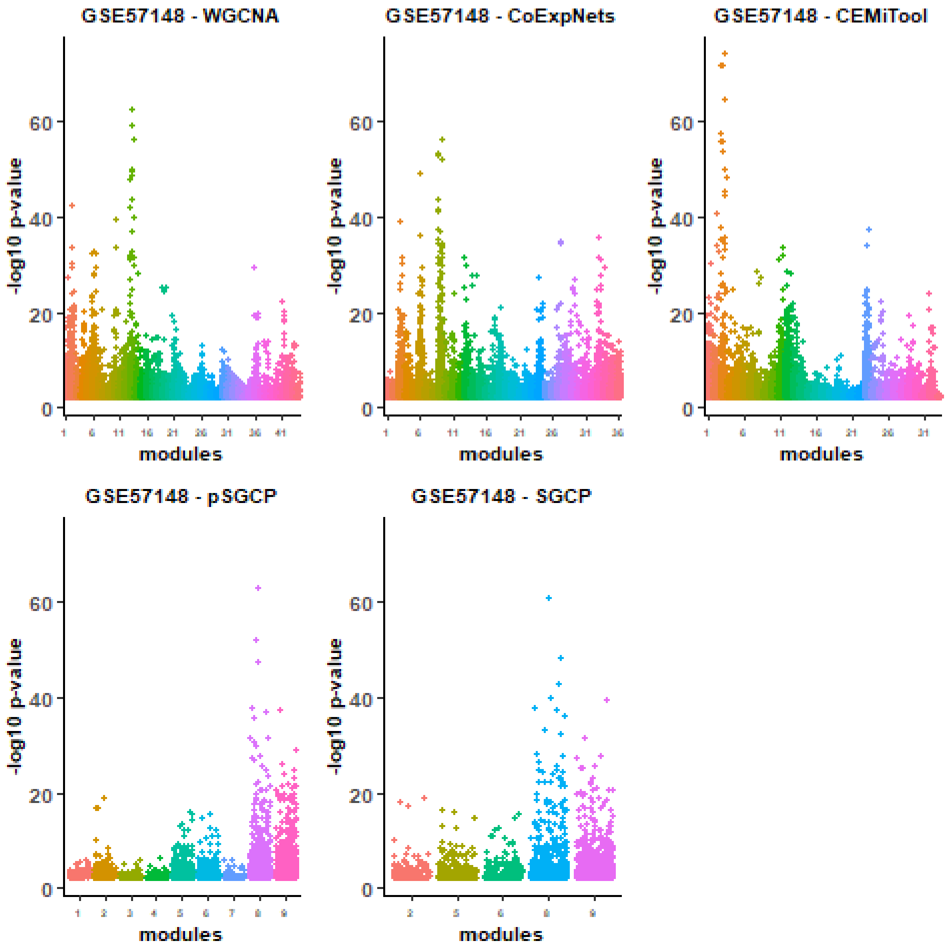


Figure C.5 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE57148 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

C.10 GO terms of GSE150961

Figure C.10 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.11 GO terms of GSE115828

Figure C.11 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by

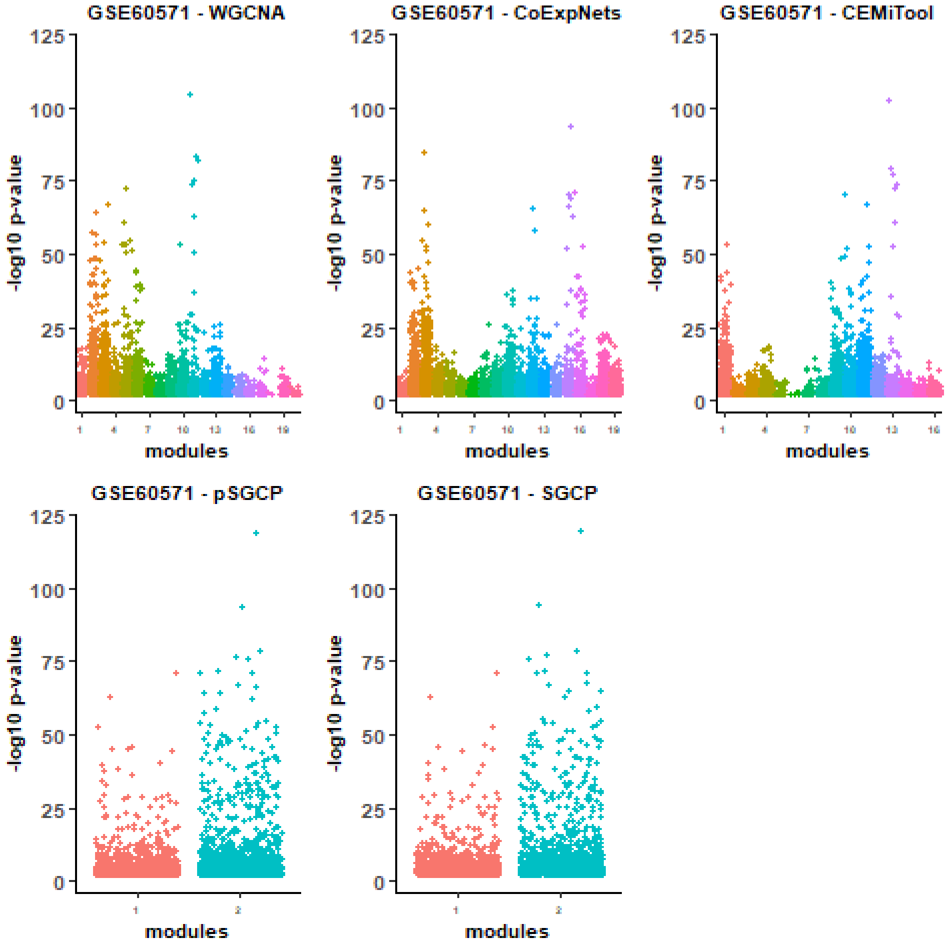


Figure C.6 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE60571 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

C.12 GO terms of GSE38705

Figure C.12 displays the significant gene ontology (GO) terms identified by each pipeline. Each point in the plot is color-coded based on the module label assigned by the corresponding pipeline, and the title of the plot identifies the pipeline name. The p-values of the GO terms are log-transformed and presented as a jitter plot.

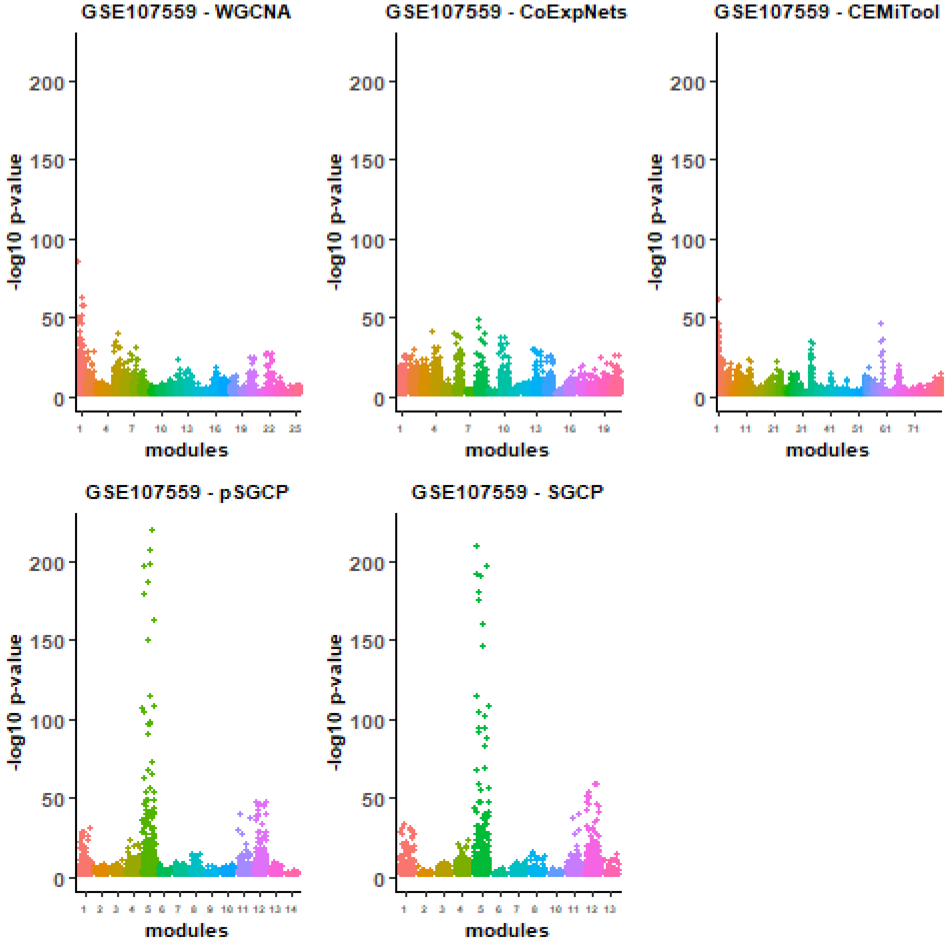


Figure C.7 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE107559 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

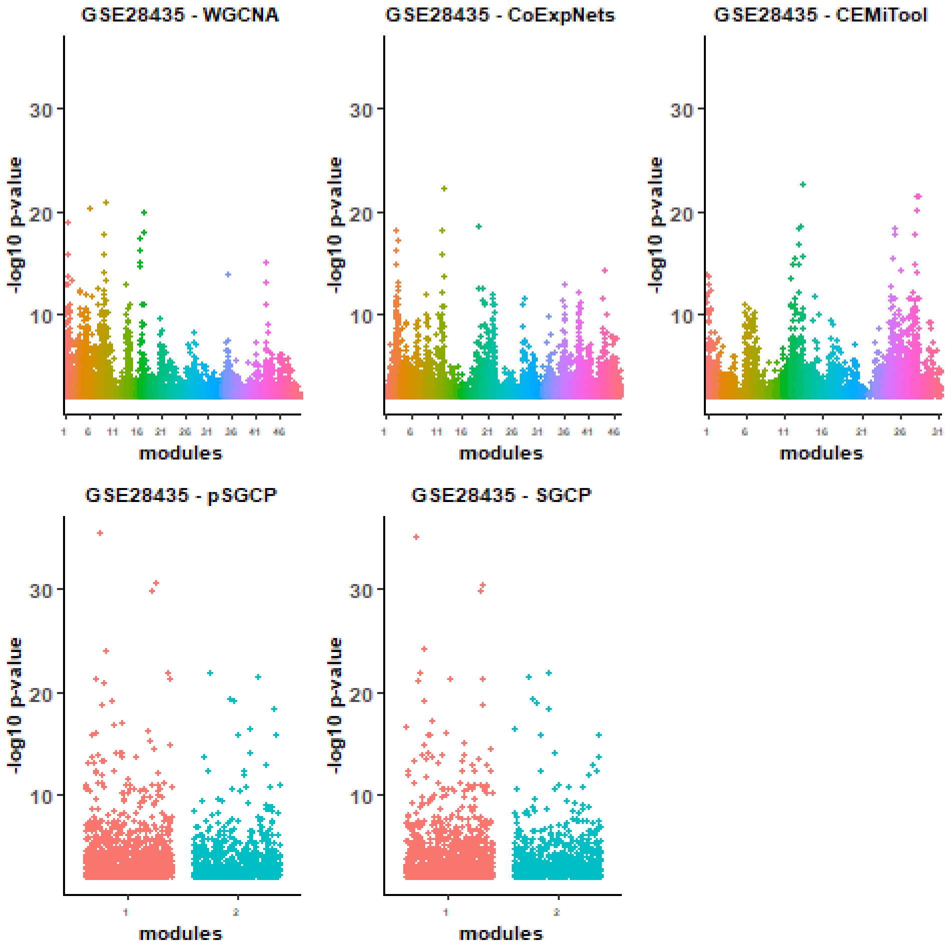


Figure C.8 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE28435 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

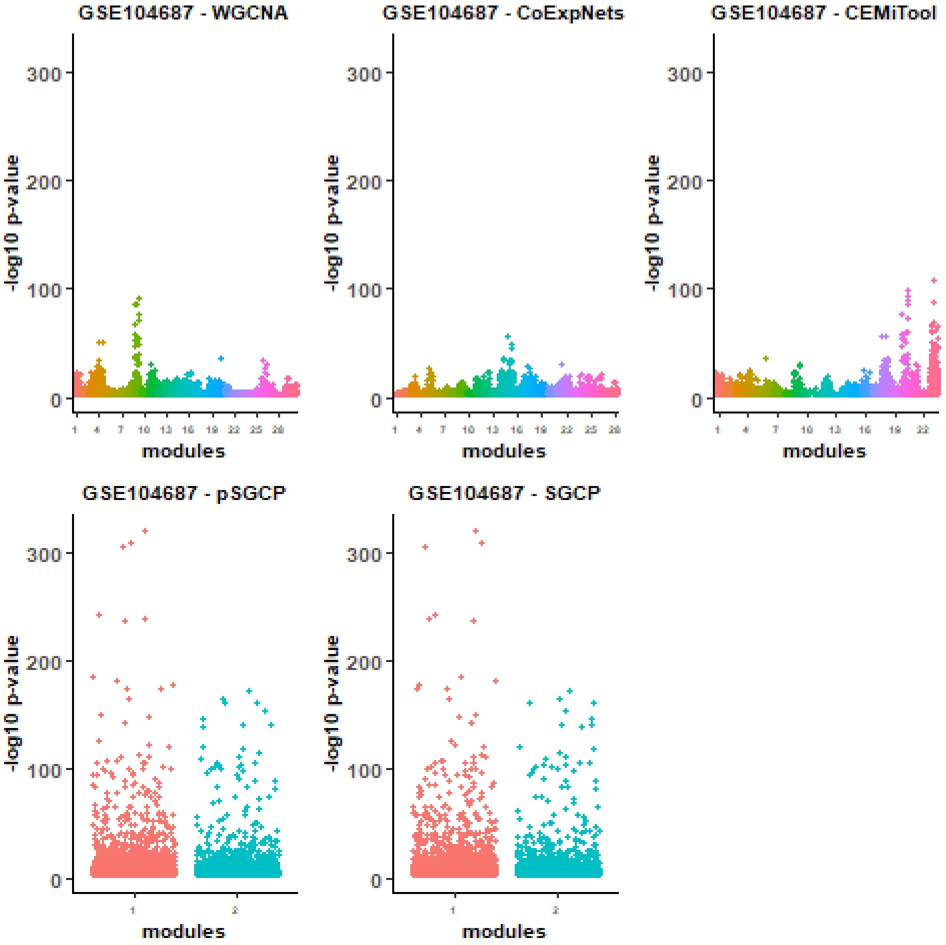


Figure C.9 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE104687 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

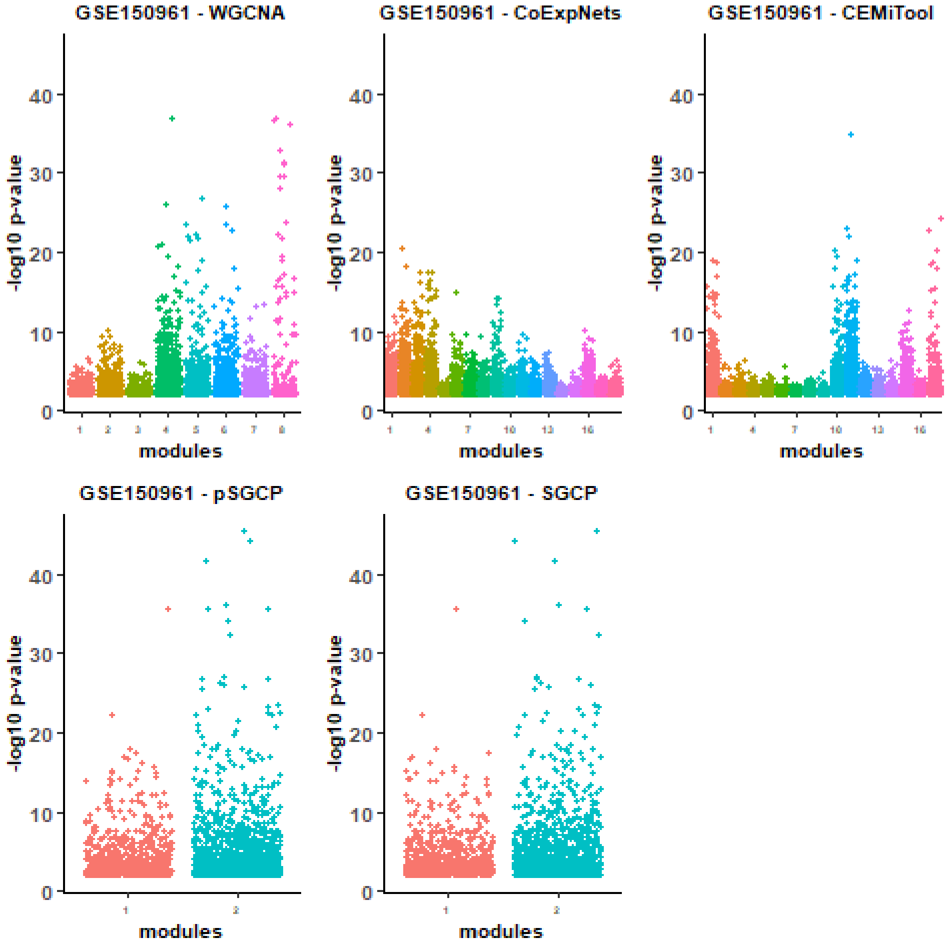


Figure C.10 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE150961 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

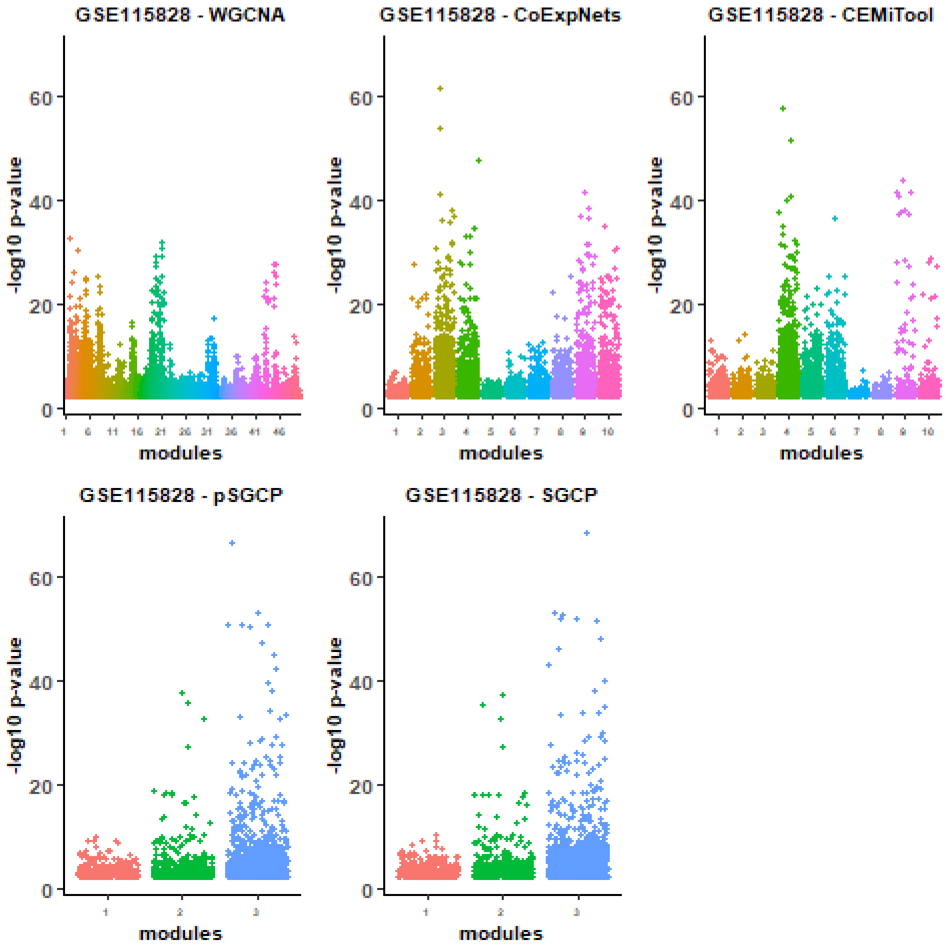


Figure C.11 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE115828 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

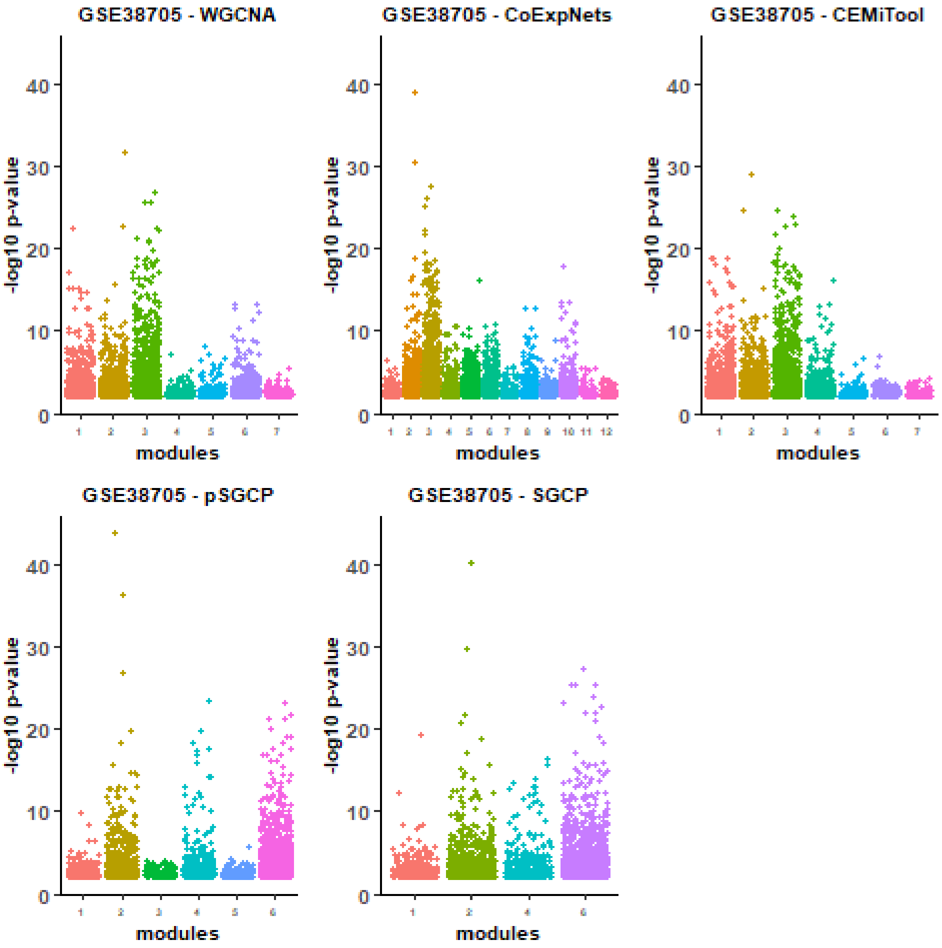


Figure C.12 Significance of gene ontology (GO) terms associated with modules identified by the WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP pipelines on the GSE38705 dataset. The *p-values* of the GO term significance for each module are log-transformed and presented as a jitter plot, with higher points indicating greater significance. The colors used in the plot correspond to the module labels. pSGCP and SGCP represent the initial clusters and final modules, respectively.

REFERENCES

[1] Farnoosh Abbas-Aghababazadeh, Qian Li, and Brooke L. Fridley. Comparison of normalization approaches for gene expression studies completed with high-throughput sequencing. *Plos One*, 13(10):e0206312–e0206312, Oct 2018.

[2] Zachary B. Abrams, Travis S. Johnson, Kun Huang, Philip R. O. Payne, and Kevin Coombes. A protocol to evaluate RNA sequencing normalization methods. *BMC Bioinformatics*, 20(24):679, Dec 2019.

[3] Niloofar Aghaieabiane and Ioannis Koutis. A novel calibration step in gene co-expression network construction. *Frontiers in Bioinformatics*, 1, 2021.

[4] Niloofar Aghaieabiane and Ioannis Koutis. SGCP: A semi-supervised pipeline for gene clustering using self-training approach in gene co-expression networks. *Biologyiv*, 2022.

[5] Jeffrey D. Allen, Yang Xie, Min Chen, Luc Girard, and Guanghua Xiao. Comparing statistical methods for constructing large scale gene networks. *Plos One*, 7(1):1–9, 01 2012.

[6] Simon Anders and Wolfgang Huber. Differential expression analysis for sequence count data. *Genome Biology*, 11(10):R10, Oct 2010.

[7] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

[8] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[9] Tanya Barrett, Stephen E. Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F. Kim, Maxim Tomashevsky, Kimberly A. Marshall, Katherine H. Phillippy, Patti M. Sherman, Michelle Holko, Andrey Yefanov, Hyeseung Lee, Naigong Zhang, Cynthia L. Robertson, Nadezhda Serova, Sean Davis, and Alexandra Soboleva. NCBI GEO: archive for functional genomics data sets—update. *Nucleic Acids Research*, 41(D1):D991–D995, 11 2012.

[10] Tanya Barrett, Stephen E. Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F. Kim, Maxim Tomashevsky, Kimberly A. Marshall, Katherine H. Phillippy, Patti M. Sherman, Michelle Holko, Andrey Yefanov, Hyeseung Lee, Naigong Zhang, Cynthia L. Robertson, Nadezhda Serova, Sean Davis, and Alexandra Soboleva. NCBI GEO: archive for functional genomics data sets–update. *Nucleic Acids Research*, 41(Database issue):D991–D995, Jan 2013.

[11] Katia Basso, Adam A Margolin, Gustavo Stolovitzky, Ulf Klein, Riccardo Dalla-Favera, and Andrea Califano. Reverse engineering of regulatory networks in human b cells. *Nature Genetics*, 37(4):382–390, 2005.

[12] Brian J Bennett, Charles R Farber, Anatole Ghazalpour, Calvin Pan, Nam Che, Pingzi Wen, Hong Xiu Qi, Adonisa Mutukulu, Nathan Siemers, Isaac Neuhaus, Roumyana Yordanova, Peter Gargalovic, Matteo Pellegrini, Todd Kirchgessner, and Aldons J Lusis. Unraveling inflammatory responses using systems genetics and gene-environment interactions in macrophages. *Cell*, 151(3):658–670, 2012.

[13] Massimo Bionaz and Juan J. Loor. Gene networks driving bovine mammary protein synthesis during the lactation cycle. *Bioinformatics and Biology Insights*, 5:BBI.S7003, 2011. PMID: 21698073.

[14] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg, Springer-Verlag, 2006.

[15] B M Bolstad, R A Irizarry, M Astrand, and T P Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, Jan 2003.

[16] Juan A. Botía, Jana Vandrovcova, Paola Forabosco, Sebastian Guelfi, Karishma D'Sa, The United Kingdom Brain Expression Consortium, John Hardy, Cathryn M. Lewis, Mina Ryten, and Michael E. Weale. An additional k-means clustering step improves the biological features of WGCNA gene co-expression networks. *BMC Systems Biology*, 11(1):47, 2017.

[17] Anna D. Broody and Aaron Clauset. Scale-free networks are rare. *Nature Communications*, 10(1):1017–1017, Mar 2019.

[18] Atul J Butte and Isaac S Kohane. Mutual information relevance networks: Functional genomic clustering using pairwise entropy measurements. *Pacific Symposium Biocomputing 2000*, pages 418–429, 2000.

[19] Zhen-Xia Chen and Brian Oliver. X chromosome and autosome dosage responses in drosophila melanogaster heads. *G3 (Bethesda, Md.)*, 5(6):1057–1063, Apr 2015.

[20] Chew Weng Cheng, David J. Beech, and Stephen B. Wheatcroft. Advantages of CEMitool for gene co-expression analysis of RNA-seq data. *Computers in Biology and Medicine*, 125:103975, 2020.

[21] David F. Choy, Guiquan Jia, Alexander R. Abbas, Katrina B. Morshead, Nicholas Lewin-Koh, Rajiv Dua, Pauline Rivera, Priscilla Moonsamy, Marcel Fontecha, Aarthi Balasubramanyam, Chris Santini, Ekaterina Bassett, Jill M. Ray, Christopher R. Cabanski, Mary S. Bradley, Romeo Maciuca, Sofia Mosesova, Heleen Scheerens, and Joseph R. Arron. Peripheral blood gene expression predicts clinical benefit from anti-IL-13 in asthma. *Journal of Allergy and Clinical Immunology*, 138(4):1230–1233.e8, 2016.

[22] Peter Clote. Are RNA networks scale-free? *Journal of Mathematical Biology*, 80(5):1291–1321, Apr 2020.

[23] Ana COnesa, Pedro Madrigal, Sonia Tarazona, David Gomez-Cabrero, Alejandra Cervera, Andrew McPherson, Michał Wojciech Szcześniak, Daniel J Gaffney, Laura L Elo, Xuegong Zhang, and Ali Mortazavi. A survey of best practices for RNA-seq data analysis. *Genome Biology*, 17:13, Jan 2016.

[24] Carsten O. Daub, Ralf Steuer, Joachim Selbig, and Sebastian Kloska. Estimating mutual information using b-spline functions – an improved similarity measure for analysing gene expression data. *BMC Bioinformatics*, 5(1):118, 2004.

[25] Richard C. Davis, Atila van Nas, Lawrence W. Castellani, Yi Zhao, Zhiqiang Zhou, Pingzi Wen, Suzanne Yu, Hongxiu Qi, Melenie Rosales, Eric E. Schadt, Karl W. Broman, Miklós Péterfy, and Aldons J. Lusis. Systems genetics of susceptibility to obesity-induced diabetes in mice. *Physiological Genomics*, 44(1):1–13, 2012. PMID: 22010005.

[26] João Pedro de Magalhães, Caleb E. Finch, and Georges Janssens. Next-generation sequencing in aging research: Emerging applications, problems, pitfalls and possible solutions. *Ageing Research Reviews*, 9(3):315 – 323, 2010.

[27] Patrik D'haeseleer. How does gene expression clustering work? *Nature Biotechnology*, 23(12):1499–1501, 2005.

[28] Marie-Agnès Dillies, Andrea Rau, Julie Aubert, Christelle Hennequet-Antier, Marine Jeanmougin, Nicolas Servant, Céline Keime, Guillemette Marot, David Castel, Jordi Estelle, Gregory Guernec, Bernd Jagla, Luc Jouneau, Denis Laloë, Caroline Le Gall, Brigitte Schaëffer, Stéphane Le Crom, Mickaël Guedj, and on behalf of The French StatOmique Consortium Jaffrézic, Florence. A comprehensive evaluation of normalization methods for illumina high-throughput RNA sequencing data analysis. *Briefings in Bioinformatics*, 14(6):671–683, 09 2012.

[29] J. C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3(3):32–57, 1973.

[30] Abbasali Emamjomeh, Elham Saboori Robat, Javad Zahiri, Mahmood Solouki, and Pegah Khosravi. Gene co-expression network reconstruction: a review on computational methods for inferring functional information from plant-based expression data. *Plant Biotechnology Reports*, 11(2):71–86, 2017.

[31] Jeremiah J Faith, Boris Hayete, Joshua T Thaden, Ilaria Mogno, Jamey Wierzbowski, Guillaume Cottarel, Simon Kasif, James J Collins, and Timothy S Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *Plos Biology*, 5(1):1–13, 01 2007.

[32] S. Falcon and R. Gentleman. Using GOstats to test gene lists for GO term association. *Bioinformatics*, 23(2):257–258, 11 2006.

[33] S. Falcon and R Gentleman. *Hypergeometric Testing Used for Gene Set Enrichment Analysis*, pages 207–220. New York, NY, Springer New York, 2008.

[34] I Gat-Viks, R Sharan, and R Shamir. Scoring clustering solutions by their biological relevance. *Bioinformatics*, 19(18):2381–9, 2003.

[35] Francis D Gibbons and Frederick P Roth. Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Research*, 12(10):1574–1581, 2002.

[36] Francis D. Gibbons and Frederick P. Roth. Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Research*, 12(10):1574–1581, Oct 2002.

[37] Antoine Godichon-Baggioni, Cathy Maugis-Rabusseau, and Andrea Rau. Clustering transformed compositional data using k-means, with applications in gene expression and bicycle sharing system data. *Applied Statistics*, 46(1):47–65, 2019.

[38] Julia Handl, Joshua Knowles, and Douglas B. Kell. Computational cluster validation in post-genomic data analysis. *Bioinformatics*, 21(15):3201–3212, 08 2005.

[39] J. A. Hartigan and M. A. Wong. Algorithm AS 136: A K-Means clustering algorithm. *Applied Statistics*, 28(1):100–108, 1979.

[40] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* New York, NY, Springer Science & Business Media, 2nd edition, 2009.

[41] Raul Herranz, Oliver J. Larkin, Richard JA Hill, Irene Lopez-Vidriero, Jack JWA van Loon, and F. Javier Medina. Suboptimal evolutionary novel environments promote singular altered gravity responses of transcriptome during drosophilametamorphosis. *BMC Evolutionary Biology*, 13(1):133, 2013.

[42] Jie Hou, Xiufen Ye, Chuanlong Li, and Yixing Wang. K-module algorithm: An additional step to improve the clustering results of WGCNA co-expression networks. *Genes*, 12(1):87, 2021.

[43] Radmila Hrdlickova, Masoud Toloue, and Bin Tian. RNA-seq methods for transcriptome analysis. *WIREs RNA*, 8(1):e1364, 2017.

[44] Yiming Hu and Hongyu Zhao. CCor: A whole genome network-based similarity measure between two genes. *Biometrics*, 72(4):1216–1225, 2016.

[45] Rafael A. Irizarry, Bridget Hobbs, Francois Collin, Yasmin D. Beazer-Barclay, Kristen J. AntOnellis, Uwe Scherf, and Terence P. Speed. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2):249–264, 04 2003.

[46] Daxin Jiang, Chun Tang, and Aidong Zhang. Cluster analysis for gene expression data: a survey. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1370–1386, 2004.

[47] Guangfu Jin, Jielin Sun, Sarah D. Isaacs, Kathleen E. Wiley, Seong-Tae Kim, Lisa W. Chu, Zheng Zhang, Hui Zhao, Siqun Lilly Zheng, William B. Isaacs, and Jianfeng Xu. Human polymorphisms at long non-coding RNAs (lncRNAs) and association with prostate cancer risk. *Carcinogenesis*, 32(11):1655–1659, 08 2011.

[48] Thomas B. Kepler, Lynn Crosby, and Kevin T. Morgan. Normalization and analysis of DNA microarray data by self-consistency and local regression. *Genome Biology*, 3(7):research0037.1, Jun 2002.

[49] Raya Khanin and Ernst Wit. How scale-free are biological networks. *Journal of Computational Biology*, 13(3):810–818, 2006.

[50] Purvesh Khatri and Sorin Drăghici. Ontological analysis of gene expression data: current tools, limitations, and open problems. *Bioinformatics*, 21(18):3587–3595, 2005.

[51] Woo Jin Kim, Jae Hyun Lim, Jae Seung Lee, Sang-Do Lee, Ju Han Kim, and Yeon-Mok Oh. Comprehensive analysis of transcriptome sequencing data in the lung tissues of COPD subjects. *International Journal of Genomics*, 2015:206937, Mar 2015.

[52] Justin B. Kinney and Gurinder S. Atwal. Equitability, mutual information, and the maximal information coefficient. *Proceedings of the National Academy of Sciences*, 111(9):3354–3359, 2014.

[53] Max Kuhn. Building predictive models in r using the caret package. *Journal of Statistical Software*, 28(5):1–26, 2008.

[54] Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[55] Sapna Kumari, Jeff Nie, Huann-Sheng Chen, Hao Ma, Ron Stewart, Xiang Li, Meng-Zhu Lu, William M. Taylor, and Hairong Wei. Evaluation of gene association methods for coexpression network construction and biological knowledge discovery. *Plos One*, 7(11):1–17, 11 2012.

[56] Peter Langfelder and Steve Horvath. WGCNA: an r package for weighted correlation network analysis. *BMC Bioinformatics*, 9(1):559, Dec 2008.

[57] Peter Langfelder and Steve Horvath. Fast r functions for robust correlations and hierarchical clustering. *Journal of Statistical Software*, 46:i11, 2012.

[58] Peter Langfelder, Bin Zhang, and Steve Horvath. Defining clusters from a hierarchical cluster tree: the dynamic tree cut package for r. *Bioinformatics*, 24(5):719–720, 11 2007.

[59] James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. Multiway spectral partitioning and higher-order cheeger inequalities. *Journal of ACM*, 61(6), Dec 2014.

[60] Bingshan Li, Lam C. Tsoi, William R. Swindell, Johann E. Gudjonsson, Trilokraj Tejasvi, Andrew Johnston, Jun Ding, Philip E. Stuart, Xianying Xing, James J. Kochkodan, John J. Voorhees, Hyun M. Kang, Rajan P. Nair, Goncalo R. Abecasis, and James T. Elder. Transcriptome analysis of psoriasis in a large case-control sample: RNA-seq provides insights into disease mechanisms. *Journal of Investigative Dermatology*, 134(7):1828–1838, Jul 2014.

[61] Bo Li and Colin N. Dewey. RSEM: accurate transcript quantification from RNA-seq data with or without a reference genome. *BMC Bioinformatics*, 12(1):323, Aug 2011.

[62] Min Li, Jian-er Chen, Jian-xin Wang, Bin Hu, and Gang Chen. Modifying the dpclus algorithm for identifying protein complexes based on new topological structures. *BMC Bioinformatics*, 9(1):398, 2008.

[63] Peipei Li, Yongjun Piao, Ho Sun Shon, and Keun Ho Ryu. Comparing the normalization methods for the differential analysis of illumina high-throughput rna-seq data. *BMC Bioinformatics*, 16(1):347, Oct 2015.

[64] Gipsi Lima-Mendez and Jacques van Helden. The powerful law of the power law and other myths in network biology. *Molecular BioSystems*, 5:1482–1493, 2009.

[65] Jing Liu, Ling Jing, and Xilin Tu. Weighted gene co-expression network analysis identifies specific modules and hub genes related to coronary artery disease. *BMC Cardiovascular Disorders*, 16(1):54, Mar 2016.

[66] Zhi-Ping Liu. Quantifying gene regulatory relationships with association measures: A comparative study. *Frontiers in Genetics*, 8:96, 2017.

[67] D J Lockhart, H Dong, M C Byrne, M T Follettie, M V Gallo, M S Chee, M Mittmann, C Wang, M Kobayashi, H Horton, and E L Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 14(13):1675–80, 1996.

[68] David J. Lockhart and Elizabeth A. Winzeler. Genomics, gene expression and dna arrays. *Nature*, 405(6788):827–836, Jun 2000.

[69] Harvey Lodish, Arnold Berk, Chris A Kaiser, Monty Krieger, Matthew P Scott, Anthony Bretscher, Hidde Ploegh, Paul Matsudaira, et al. *Molecular cell Biology*. New York, NY, Macmillan, sixth edition, 2008.

[70] Michael I. Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 15(12):550, Dec 2014.

[71] Lina E. Lundberg, Margarida L. A. Figueiredo, Per Stenberg, and Jan Larsson. Buffering and proteolysis are induced by segmental monosomy in drosophila melanogaster. *Nucleic Acids Research*, 40(13):5926–5937, 03 2012.

[72] Xuelian Ma, Hansheng Zhao, Wenying Xu, Qi You, Hengyu Yan, Zhimin Gao, and Zhen Su. Co-expression gene network analysis and functional module identification in bamboo growth and development. *Frontiers in Genetics*, 9:574, 2018.

[73] Kirk J MantiOne, Richard M Kream, Hana Kuzelova, Radek Ptacek, Jiri Raboch, Joshua M Samuel, and George B Stefano. Comparing bioinformatic gene expression profiling methods: microarray and RNA-seq. *Medical science monitor basic research*, 20:138 – 142, 2014.

[74] Adam A. Margolin, Ilya Nemenman, Katia Basso, Chris Wiggins, Gustavo Stolovitzky, and Andrea Favera, Riccardo Dalla Califano. ARACNE: An lgorithm for the reconstruction of gene regulatory networks in a mammalian cellular. *BMC Bioinformatics*, 7(s7):1, 2006.

[75] Mike J. Mason, Guoping Fan, Kathrin Plath, Qing Zhou, and Steve Horvath. Signed weighted gene co-expression network analysis of transcriptional regulation in murine embryonic stem cells. *BMC Genomics*, 10(1):327, 2009.

[76] Matthew N. McCall, Benjamin M. Bolstad, and Rafael A. Irizarry. Frozen robust multiarray analysis (fRMA). *Biostatistics*, 11(2):242–253, Apr 2010.

[77] Jeremy A Miller, Angela Guillozet-Bongaarts, Laura E Gibbons, Nadia Postupna, Anne Renz, Allison E Beller, Susan M Sunkin, Lydia Ng, Shannon E Rose, Kimberly A Smith, Aaron Szafer, Chris Barber, Darren Bertagnolli, Kristopher Bickley, Krissy Brouner, Shiella Caldejon, Mike Chapin, Mindy L Chua, Natalie M Coleman, Eiron Cudaback, Christine Cuhaciyan, Rachel A Dalley, Nick Dee, Tsega Desta, Tim A Dolbeare, Nadezhda I Dotson, Michael Fisher, Nathalie Gaudreault, Garrett Gee, Terri L Gilbert, Jeff Goldy, Fiona Griffin, Caroline Habel, Zeb Haradon, Nika Hejazinia, Leanne L Hellstern, Steve Horvath, Kim Howard, Robert Howard, Justin Johal, Nikolas L Jorstad, Samuel R Josephsen, Chihchau L Kuan, Florence Lai, Eric Lee, Felix Lee, Tracy Lemon, Xianwu Li, Desiree A Marshall, Jose Melchor, Shubhabrata Mukherjee, Julie Nyhus, Julie Pendergraft, Lydia Potekhina, Elizabeth Y Rha, Samantha Rice, David Rosen, Abharika Sapru, Aimee Schantz, Elaine Shen, Emily Sherfield, Shu Shi, Andy J Sodt, Nivretta Thatra, Michael Tieu, Angela M Wilson, Thomas J Montine, Eric B Larson, Amy Bernard, Paul K Crane, Richard G Ellenbogen, C Dirk Keene, and Ed Lein. Neuropathological and transcriptomic characteristics of the aged brain. *Elife*, 6:e31126, Nov 2017.

[78] Angela Mo, Sini Nagpal, Kyle Gettler, Talin Haritunians, Mamta Giri, Yael Haberman, Rebekah Karns, Jarod Prince, Dalia Arafat, Nai-Yun Hsu, Ling-Shiang Chuang, Carmen Argmann, Andrew Kasarskis, Mayte Suarez-Farinas, Nathan Gotman, Emebet Mengesha, Suresh Venkateswaran, Paul A. Rufo, Susan S. Baker, Cary G. Sauer, James Markowitz, Marian D. Pfefferkorn, Joel R. Rosh, Brendan M. Boyle, David R. Mack, Robert N. Baldassano, Sapana Shah, Neal S. LeLeiko, Melvin B. Heyman, Anne M. Griffiths, Ashish S. Patel, Joshua D. Noe, Sonia Davis Thomas, Bruce J. Aronow, Thomas D. Walters, Dermot P.B. McGovern, Jeffrey S. Hyams, Subra Kugathasan, Judy H. Cho, Lee A. Denson, and Greg Gibson. Stratification of risk of progression to colectomy in ulcerative colitis via measured and predicted gene expression. *The American Journal of Human Genetics*, 108(9):1765–1779, 2021.

[79] Ali Mortazavi, Brian A Williams, Kenneth McCue, Lorian Schaeffer, and Barbara Wold. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature Methods*, 5(7):621–628, May 2008.

[80] Frank Nielsen. *Introduction to HPC with MPI for Data Science*. Cham, Switzerland, Springer Cham, first edition, 2016.

[81] Jelili Oyelade, Itunuoluwa Isewon, Funke Oladipupo, Olufemi Aromolaran, Efosa Uwoghiren, Faridah Ameh, Moses Achas, and Ezekiel Adebiyi. Clustering algorithms: Their application to gene expression data. *Bioinformatics and Biology Insights*, 10:237–253, Nov 2016.

[82] Bahman Panahi and Mohammad Amin Hejazi. Weighted gene co-expression network analysis of the salt-responsive transcriptomes reveals novel hub genes in green halophytic microalgae Dunaliella salina. *Scientific Reports*, 11(1):1607, Jan 2021.

[83] Taesung Park, Sung-Gon Yi, Sung-Hyun Kang, SeungYeoun Lee, Yong-Sung Lee, and Richard Simon. Evaluation of normalization methods for microarray data. *BMC Bioinformatics*, 4(1):33, Sep 2003.

[84] Princy Parsana, Claire Ruberman, Andrew E. Jaffe, Michael C. Schatz, Alexis Battle, and Jeffrey T. Leek. Addressing confounding artifacts in reconstruction of gene co-expression networks. *Genome Biology*, 20(1):94, 2019.

[85] Chao-Ying Joanne Peng, Kuk Lida Lee, and Gary M. Ingersoll. An introduction to logistic regression analysis and reporting. *The Journal of Educational Research*, 96(1):3–14, 2002.

[86] Juli Petereit, Sebastian Smith, Frederick C. Harris, and Karen A. Schlauch. petal: Co-expression network modelling in r. *BMC Systems Biology*, 10(2):51, 2016.

[87] Ralph B. Puchalski, Nameeta Shah, Jeremy Miller, Rachel Dalley, Steve R. Nomura, Jae-Guen Yoon, Kimberly A. Smith, Michael Lankerovich, Darren Bertagnolli, Kris Bickley, Andrew F. Boe, Krissy Brouner, Stephanie Butler, Shiella Caldejon, Mike Chapin, Suvro Datta, Nick Dee, Tsega Desta, Tim Dolbeare, Nadezhda Dotson, Amanda Ebbert, David Feng, Xu Feng, Michael Fisher, Garrett Gee, Jeff Goldy, Lindsey Gourley, Benjamin W. Gregor, Guangyu Gu, Nika Hejazinia, John Hohmann, Parvinder Hothi, Robert Howard, Kevin Joines, Ali Kriedberg, Leonard Kuan, Chris Lau, Felix Lee, Hwahyung Lee, Tracy Lemon, Fuhui Long, Naveed Mastan, Erika Mott, Chantal Murthy, Kiet Ngo, Eric Olson, Melissa Reding, Zack Riley, David Rosen, David Sandman, Nadiya Shapovalova, Clifford R. Slaughterbeck, Andrew Sodt, Graham Stockdale, Aaron Szafer, Wayne Wakeman, Paul E. Wohnoutka, Steven J. White, Don Marsh, Robert C. Rostomily, Lydia Ng, Chinh Dang, Allan JOnes, Bart Keogh, Haley R. Gittleman, Jill S. Barnholtz-Sloan, Patrick J. Cimino, Megha S. Uppin, C. Dirk Keene, Farrokh R. Farrokhi, Justin D. Lathia, Michael E. Berens, Antonio IavarOne, Amy Bernard, Ed Lein, John W. Phillips, Steven W. Rostad, Charles Cobbs, Michael J. Hawrylycz, and Greg D. Foltz. An anatomic transcriptional atlas of human glioblastoma. *Science*, 360(6389):660–663, 2018.

[88] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 459–467. Marina Del Rey, CA, ACM, Feb 2018.

[89] John Quackenbush. Microarray data normalization and transformation. *Nature Genetics*, 32(4):496–501, Dec 2002.

[90] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2013.

[91] Mohan S. Rao, Terry R. Van Vleet, Rita Ciurlionis, Wayne R. Buck, Scott W. Mittelstadt, Eric A. G. Blomme, and Michael J. Liguori. Comparison of RNA-seq and microarray gene expression platforms for the toxicogenomic evaluation of liver from short-term rat toxicity studies. *Frontiers in Genetics*, 9:636, 2019.

[92] Rinki Ratnapriya, Olukayode A. Sosina, Margaret R. Starostik, Madeline Kwicklis, Rebecca J. Kapphahn, Lars G. Fritsche, Ashley Walton, Marios Arvanitis, Linn Gieser, Alexandra Pietraszkiewicz, Sandra R. Montezuma, Emily Y. Chew, Alexis Battle, Gonçalo R. Abecasis, Deborah A. Ferrington, Nilanjan Chatterjee, and Anand Swaroop. Retinal transcriptome and eQTL analyses identify genes associated with age-related macular degeneration. *Nature Genetics*, 51(4):606–610, Apr 2019.

[93] David N. Reshef, Yakir A. Reshef, Hilary K. Finucane, Sharon R. Grossman, Gilean McVean, Peter J. Turnbaugh, Eric S. Lander, Michael Mitzenmacher, and Pardis C. Sabeti. Detecting novel associations in large data sets. *Science*, 334(6062):1518–1524, 2011.

[94] Mark D Robinson and Alicia Oshlack. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biology*, 11(3):R25, Mar 2010.

[95] Sophie E. Ruff, Nikita Vasilyev, Evgeny Nudler, Susan K. Logan, and Michael J. Garabedian. PIM1 phosphorylation of the androgen receptor and 14-3-3 ζ regulates gene transcription in prostate cancer. *Communications Biology*, 4(1):1221, Oct 2021.

[96] Pedro S. T. Russo, Gustavo R. Ferreira, Lucas E. Cardozo, Matheus C. Bürger, Raul Arias-Carrasco, Sandra R. Maruyama, Thiago D. C. Hirata, Diógenes S. Lima, Fernando M. Passos, Kiyoshi F. Fukutani, Melissa Lever, João S. Silva, Vinicius Maracaja-Coutinho, and Helder I. Nakaya. CEMitool: a bioconductor package for performing comprehensive modular co-expression analyses. *BMC Bioinformatics*, 19(1):56, 2018.

[97] A. Rényi. On measures of dependence. *Acta Mathematica Academiae Scientiarum Hungarica*, 10(3):441–451, 1959.

[98] Alex Sánchez and MC de Villa. A tutorial review of microarray data analysis. Technical report, Universitat de Barcelona, Barcelon, Spain, 2008.

[99] Robert J. Schaefer, Roman Briskine, Nathan M. Springer, and Chad L. Myers. Discovering functional modules across diverse maize transcriptomes using COB, the co-expression browser. *Plos One*, 9(6):1–11, 06 2014.

[100] Robert J. Schaefer, Jean-Michel Michno, and Chad L. Myers. Unraveling gene function in agricultural species using gene co-expression networks. *Biochimica et Biophysica Acta (BBA) - Gene Regulatory Mechanisms*, 1860(1):53 – 63, 2017. Plant Gene Regulatory Mechanisms and Networks.

[101] Elise A. R. Serin, Harm Nijveen, Henk W. M. Hilhorst, and Wilco Ligterink. Learning from co-expression networks: Possibilities and challenges. *Frontiers in Plant Science*, 7:444, 2016.

[102] Elise A. R. Serin, Harm Nijveen, Henk W. M. Hilhorst, and Wilco Ligterink. Learning from co-expression networks: Possibilities and challenges. *Frontiers in Plant Science*, 7:444, 2016.

[103] Andrew J. Severin, Jenna L. Woody, Yung-Tsi Bolon, Bindu Joseph, Brian W. Diers, Andrew D. Farmer, Gary J Muehlbauer, Rex T. Nelson, David Grant, James E. Specht, Michelle A. Graham, Steven B. Cannon, Gregory D. May, Carroll P. Vance, and Randy C. Shoemaker. RNA-seq atlas of glycine max: A guide to the soybean transcriptome. *BMC Plant Biology*, 10(1):160, 2010.

[104] Jifan Shi, Juan Zhao, Tiejun Li, and Luonan Chen. Detecting direct associations in a network by information theoretic approaches. *Science China Mathematics*, 62(5):823–838, 2019.

[105] Noah Simon and Robert Tibshirani. Comment on "detecting novel associations in large data sets" by reshef et al, science dec 16, 2011. *Biologyiv preprint Biologyiv:1401.7645*, 2014.

[106] Lin Song, Peter Langfelder, and Steve Horvath. Comparison of co-expression measures: mutual information, correlation, and model based indices. *BMC Bioinformatics*, 13(1):328, Dec 2012.

[107] Kimberly D Spradling, Lucille A Lumley, Christopher L Robison, James L Meyerhoff, and 3rd Dillman, James F. Transcriptional analysis of rat piriform cortex following exposure to the organophosphonate anticholinesterase sarin and induction of seizures. *Journal of Neuroinflammation*, 8:83–83, 2011.

[108] Rory Stark, Marta Grzelak, and James Hadfield. RNA sequencing: the teenage years. *Nature Reviews Genetics*, 20(11):631–656, Nov 2019.

[109] R. Steuer, J. Kurths, C. O. Daub, J. Weise, and J. Selbig. The mutual information: Detecting and evaluating dependencies between variables. *Bioinformatics*, 18(2):S231–S240, 2002.

[110] Joachim Theilhaber, Sanjay N. Rakhade, Judy Sudhalter, Nayantara Kothari, Peter Klein, Jack Pollard, and Frances E. Jensen. Gene expression profiling of a hypoxic seizure model of epilepsy suggests a role for mTOR and wnt signaling in epileptogenesis. *Plos One*, 8(9):1–19, 09 2013.

[111] Paolo Tieri, Lorenzo Farina, Manuela Petti, Laura Astolfi, Paola Paci, and Filippo CastigliOne. Network inference and reconstruction in bioinformatics. In Shoba Ranganathan, Michael Gribskov, Kenta Nakai, and Christian Schönbach, editors, *Encyclopedia of Bioinformatics and Computational Biology*, pages 805 – 813. Oxford, England, Academic Press, 2019.

[112] Cole Trapnell, Brian A. Williams, Geo Pertea, Ali Mortazavi, Gordon Kwan, Marijke J. van Baren, Steven L. Salzberg, Barbara J. Wold, and Lior Pachter. Transcript assembly and quantification by RNA-seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature Biotechnology*, 28(5):511–515, May 2010.

[113] BjÖrn Usadel, Takeshi Obayashi, Marek Mutwil, Federico M. Giorgi, George W. Bassel, Mimi Tanimoto, Amanda Chow, Dirk Steinhauser, Staffan Persson, and Nicholas J. Provart. Co-expression tools for plant biology: opportunities for hypothesis generation and caveats. *Plant, Cell & Environment*, 32(12):1633–1651, 2009.

[114] BjÖrn Usadel, Fabien Poree, Axel Nagel, Marc Lohse, Angelika Czedik-eysenberg, and Mark Stitt. A guide to using mapman to visualize and compare omics data in plants: a case study in the crop species, maize. *Plant, Cell & Environment*, 32(9):1211–1229, 2009.

[115] Sipko van Dam, Urmo Võsa, Adriaan van der Graaf, Lude Franke, and João Pedro de Magalhães. Gene co-expression analysis for functional classification and gene–disease predictions. *Briefings in Bioinformatics*, 19(4):575–592, 01 2017.

[116] Monique G. P. van der Wijst, Dylan H. de Vries, Harm Brugge, Harm-Jan Westra, and Lude Franke. An integrative approach for building personalized gene regulatory networks for precision medicine. *Genome Medicine*, 10(1):96, 2018.

[117] Elisabet Van Loon, Stéphane Gazut, Saleh Yazdani, Evelyne Lerut, Henriette de Loor, Maarten Coemans, Laure-Héléne Noël, Lieven Thorrez, Leentje Van Lommel, Frans Schuit, Ben Sprangers, Dirk Kuypers, Marie Essig, Wilfried Gwinner, Dany Anglicheau, Pierre Marquet, and Maarten Naesens. Development and validation of a peripheral blood mRNA assay for the assessment of antibody-mediated kidney allograft rejection: A multicentre, prospective study. *EBioMedicine*, 46:463 – 472, 2019.

[118] Y. X. Rachel Wang, Michael S. Waterman, and Haiyan Huang. Gene coexpression measures in large heterogeneous samples using count statistics. *Proceedings of the National Academy of Sciences*, 111(46):16371–16376, 2014.

[119] M. Watson. Coxpress: Differential co-expression in gene expression data. *BMC Bioinformatics*, 7(509), 2006.

[120] Seung Yon Rhee, Valerie Wood, Kara Dolinski, and Sorin Draghici. Use and misuse of the gene ontology annotations. *Nature Reviews Genetics*, 9(7):509–515, 2008.

[121] Bin Zhang and Steve Horvath. A general framework for weighted gene co-expression network analysis. *Statistical Applications in Genetics and Molecular Biology*, 4(1), 2005.

[122] Yi Zhao, Hui Li, Shuangsang Fang, Yue Kang, Wei wu, Yajing Hao, Ziyang Li, Dechao Bu, Ninghui Sun, Michael Q. Zhang, and Runsheng Chen. NONCODE 2016: an informative and valuable data source of long non-coding RNAs. *Nucleic Acids Research*, 44(D1):D203–D208, 11 2015.

[123] Yingdong Zhao, Ming-Chung Li, Mariam M. Konaté, Li Chen, Biswajit Das, Chris Karlovich, P. Mickey Williams, Yvonne A. Evrard, James H. Doroshow, and Lisa M. McShane. TPM, FPKM, or normalized counts? a comparative study of quantification measures for the analysis of RNA-seq data from the NCI patient-derived models repository. *Journal of Translational Medicine*, 19(1):269, Jun 2021.

[124] Yingwen Zhao, Jun Wang, Jian Chen, Xiangliang Zhang, Maozu Guo, and Guoxian Yu. A literature review of gene function prediction by modeling gene ontology. *Frontiers in Genetics*, 11:400, 2020.