BMC Bioinformatics

RESEARCH

Open Access

SGCP: a spectral self-learning method for clustering genes in co-expression networks

Niloofar Aghaieabiane[1†] and Ioannis Koutis[1*†]

†Niloofar Aghaieabiane and Ioannis Koutis have contributed equally to this work.

*Correspondence: ikoutis@njit.edu

[1] Computer Science Department, New Jersey Institute of Technology, Newark, NJ 07102, USA

Abstract

Background: A widely used approach for extracting information from gene expression data employs the construction of a *gene co-expression network* and the subsequent computational detection of gene clusters, called *modules*. WGCNA and related methods are the de facto standard for module detection. The purpose of this work is to investigate the applicability of more sophisticated algorithms toward the design of an alternative method with enhanced potential for extracting biologically meaningful modules.

Results: We present *self-learning gene clustering pipeline* (SGCP), a spectral method for detecting modules in gene co-expression networks. SGCP incorporates multiple features that differentiate it from previous work, including a novel step that leverages gene ontology (GO) information in a *self-leaning* step. Compared with widely used existing frameworks on 12 real gene expression datasets, we show that SGCP yields modules with higher GO enrichment. Moreover, SGCP assigns highest statistical importance to GO terms that are mostly different from those reported by the baselines.

Conclusion: Existing frameworks for discovering clusters of genes in gene co-expression networks are based on relatively simple algorithmic components. SGCP relies on newer algorithmic techniques that enable the computation of highly enriched modules with distinctive characteristics, thus contributing a novel alternative tool for gene co-expression analysis.

Keywords: Gene co-expression networks, Gene modules, GO enrichment, WGCNA

Background

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 [1] and essential genes in microalgae [2], to assigning unknown genes to biological functions [3] and recognizing disease mechanisms [4], e.g. for coronary artery disease [5].

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 [6]. 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* [7, 8]. 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 [9].

The de facto standard automatic technique for module quality analysis is Gene Ontology (GO) enrichment, a method that 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 [9–12]. Statistically, in a given module, this method determines the significance of the GO terms for a test query by associating *p-values*. These 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 that appear more frequently than would be expected by chance [10]. As usual, the smaller the p-value the more significant the GO term.

Several frameworks and algorithms have been developed for GCNs construction and analysis such as [11–17]. Among them, Weighted Correlation Network Analysis (WGCNA) [14], is still the most widely accepted and used framework for module detection in GCNs [5, 9, 11, 12, 18]. 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 [19].

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 [12]. In another study, it is shown that a calibration pre-processing step in raw gene expression data results in increased GO enrichment [9]. Two other frameworks, the popular CoExpNets [11] and K-Module [18], have utilized k-means clustering [20] as a post-processing step to the output of WGCNA. Finally, in a comparative study, CEMiTool appears to have an advantage over WGCNA [21].

All existing frameworks share similar algorithmic components that derive from the original work on WGCNA [14]. Since the inception of WGCNA there has been major progress in algorithms for unsupervised network clustering and their mathematical understanding [22]. This work is informed and motivated by this recent progress. The objective is to adapt and apply these new algorithmic techniques toward the design of an alternative module detection method that can offer a credible and easy-to-use complementary tool for biological discovery.

Results

We have developed Self-Learning Gene Clustering Pipeline (SGCP), a user-friendly R package for GCNs construction and analysis.[1] Its integration with Bioconductor makes it easy to incorporate into existing workflows.

[1] See https://github.com/na396/SGCP for the latest version.

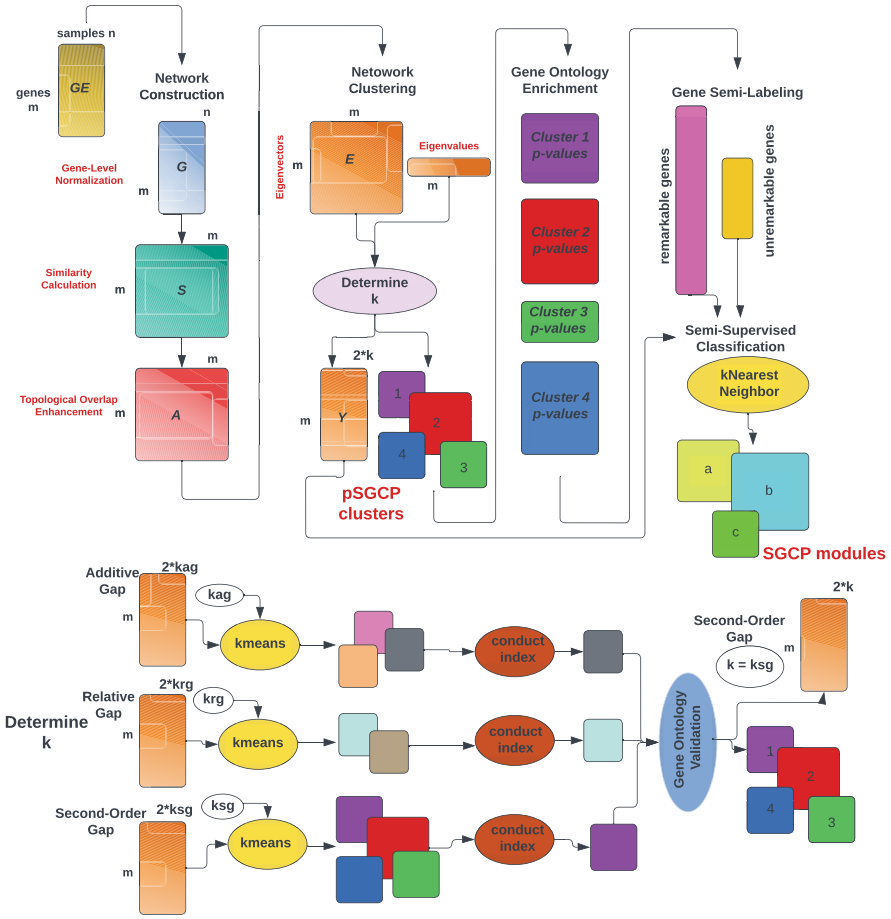

Fig. 1 The SGCP pipeline for gene clustering in gene co-expression networks. SGCP takes the gene expression matrix *GE* and outputs clusters and their refinements to modules after the semi-supervised classification steps. The steps for determining the number of clusters *k* are drawn below the main pipeline

An overview of the SGCP pipeline

SGCP differentiates itself from existing frameworks in several ways, discussed in Sect. 3. The workflow of SGCP is illustrated in Fig. 1.

In this subsection, 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:

(a) *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 in the form of topological overlap measure (TOM) [13]. The result of this step is an $m \times m$ symmetric adjacency matrix A (Sect. 4.1.1).

(b) *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 (Sect. 4.1.2). 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* (Sect. 4.1.2).

(c) *Gene ontology enrichment*: GO enrichment analysis is carried out on the initial clusters individually (Sect. 4.1.3).

(d) *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.

(e) *Supervised classification*: The supervised classification problem is solved with an appropriately selected and configured machine learning algorithm (either k-nearest neighbors [23], or one-vs-rest logistic regression [23]) 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 (Sect. 4.1.5).

Comparisons with baselines

We present a summary of our extensive experiments that demonstrate that SGCP outperforms three competing baselines on a wide variety of datasets. The GO enrichment results for all pipelines and all 12 datasets are posted on https://github.com/na396/SGCP.

We compare pSGCP (i.e. SGCP without semi-supervised cluster improvement) and SGCP with three pipelines (WGCNA, CoExpNets, CEMiTool) on 12 gene expression datasets: 4 DNA-microarray datasets, and 8 RNA-sequencing datasets as follow. These include expression arrays with a wide range of samples from 5 to 511, various organisms, along with different units [24]. Expression units provide a digital measure of the abundance of genes or transcripts. The datasets were downloaded from the NCBI Gene Expression Omnibus (GEO) database [25]. Details on the datasets are available in Table 1. We note that raw DNA-microarray datasets are normalized using robust multiarray analysis (RMA) [26] which is the most popular preprocessing step for Affymetrix [27] expression arrays data [28].

Table 1 Benchmark datasets.

Data	Type	Organism	#Samples	Unit	sft
GSE181225 [29]	RNA	Hs	5	RLE	26
GSE33779 [30]	DNA	Dm	90	probes	14
GSE44903 [31]	DNA	Rn	142	probes	30
GSE54456 [32]	RNA	Hs	174	RPKM	30
GSE57148 [33]	RNA	Hs	189	FPKM	14
GSE60571 [34]	RNA	Dm	235	FPKM	9
GSE107559 [35]	RNA	Hs	270	FPKM	3
GSE28435 [36]	DNA	Rn	335	probes	22
GSE104687 [37]	RNA	Hs	377	FPKM	18
GSE150961 [38]	RNA	Hs	418	TMM	5
GSE115828 [39]	RNA	Hs	453	CPM	12
GSE38705 [40]	DNA	Mm	511	probes	16

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

We look at the following metrics of quality.

1. *Average cluster quality.* We follow previous convention and methodology [41, 42], 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 Fig. 2a. 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 6/12 of the datasets, it improves the module quality.
2. *Most significant GO terms.* The summary evaluation includes all p-values for the GO terms, as reported by GOstats [43], but here we focus on the top 100 p-values for each pipeline. Figure 2b 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 5 datasets (GSE181225, GSE54456, GSE107559, GSE28435, GSE104687), 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.
3. *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 10 most significant GO terms in the prominent module and we show their

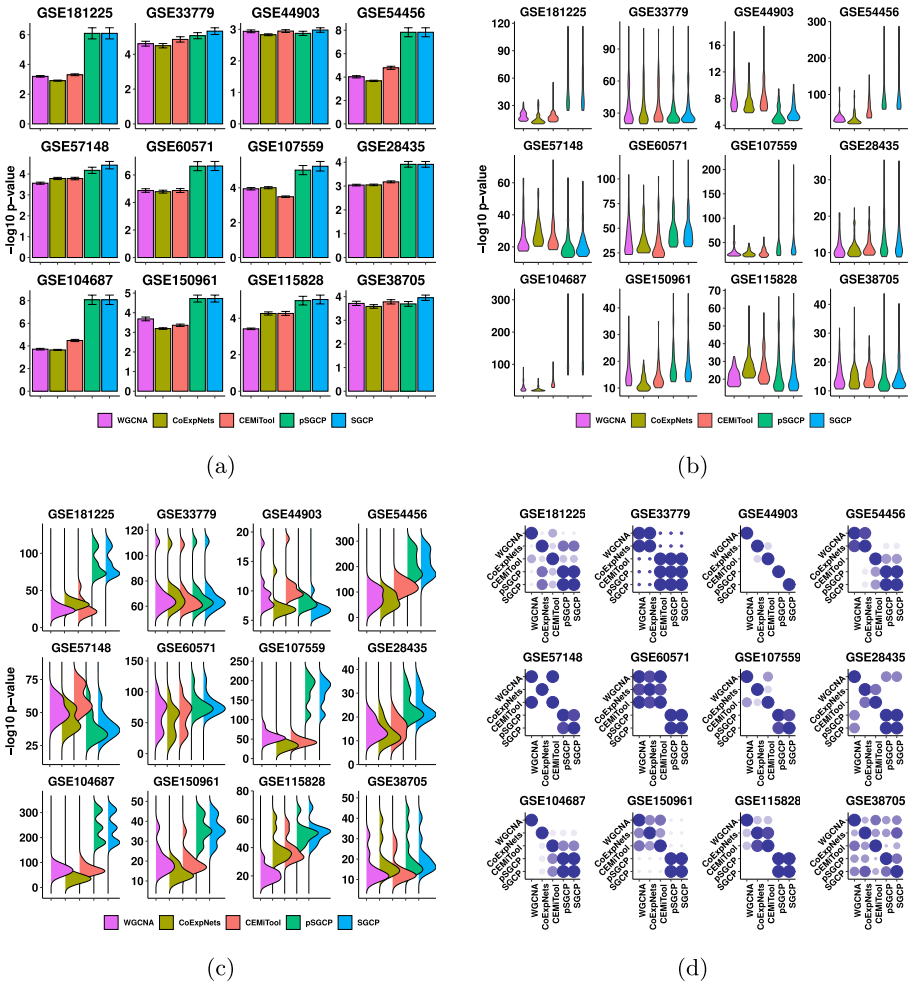

Fig. 2 Comparing WGCNA, CoExpNets, CEMiTool, pSGCP, and SGCP for gene ontology enrichment analysis in 12 real datasets: *p*-values are log-transformed. The order of the pipelines from left to right is WGCNA (purple), CoExpNets(yellow), CEMiTool (orange), pSGCP(green), and SGCP (blue). **a** All *p*-values 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. **b** Top 100 most significant *p*-values from all modules are shown as a violin plot. **c** Top 10 most significant *p*-values for the prominent module for each pipeline. **d** Overlaps in top-100 GO terms reported by the five different frameworks. For pipeline *p* in the *x*-axis and pipeline *q* in the *y*-axis, position (*p*, *q*) shows the number of GO terms reported by both *p* and *q*, among their top unique 100 GO terms. The bigger and darker the circle, the higher the overlap

p-values in Fig. 2c. 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 than SGCP. In 6 of the datasets, pSGCP and SGCP are astonishingly better than the other frameworks.

4. *Overlap in significant GO terms.* It is interesting to investigate the overlap of GO terms reported by the different frameworks. Here, we focus on the overlapping among the top 100 GO terms in the prominent module and we show their overlapping in Fig. 2d. The most significant GO terms reported by SGCP are mostly different from those reported by the baseline frameworks. 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.

Discussion

Contrasting SGCP with existing frameworks

SGCP deviates from commonly used existing pipelines for GCNs in three key ways:

1. *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 [44–48].
2. *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 [22].
3. *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.

The effect of supervised re-classification

Once initial clusters are produced, SGCP carries out an additional semi-supervised re-classification of genes to return final modules, as described in Sect. 2. A summary of the impact of this final step is given in Table 2 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. 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 Fig. 3. 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 2 clusters. In general, if there are clusters that are not enriched the re-classification step eliminates these clusters.

Table 2 Summary statistics of applying pipelines WGCNA, CoExpNets, CEMiTool, PSGCP, SGCP

	WGCNA		CoExpNets		CEMiTool		pSGCP		SGCP				
GSE	k	#GO-T	k	#GO-T	k	#GO-T	k	#GO-T	k	#GO-T	mth	% UNR	% CH
181225	48	7462	75	9027	32	6252	2	2598	2	2598	ag,rg	1%	0%
33779	22	5631	19	6213	17	5299	10	4144	7	3821	ag	56%	47.1%
44903	18	3298	27	4705	14	3303	4	987	4	1059	rg	29%	5%
54456	31	9386	46	14473	22	11056	3	6004	3	600	ag,rg	1%	1%
57148	45	13296	36	14110	33	12027	9	2833	5	2383	sg	46%	24%
60571	21	7107	19	8622	16	5564	2	2969	2	2971	ag,rg	21%	2%
107559	26	10915	20	12499	80	15952	14	5257	12	4913	sg	6%	48.4%
28435	51	7331	47	7769	31	6566	2	2052	2	2053	sg	12%	0%
104687	31	10339	28	1193	23	11369	2	6426	2	6426	ag,rg	0%	0%
150961	9	3619	18	606	17	4856	2	2111	2	2111	ag,rg	9%	0%
115828	51	12611	10	7693	10	5231	3	1934	3	1926	sg	33%	0%
38705	8	3320	12	4123	7	3308	6	2824	4	2610	sg	39%	62%

For each of the 5 pipelines, k is the number of clusters, and #GO-T indicates 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 indicates the percentage of the entire genes that are unremarkable. %CH indicates the percentage of the unremarkable genes whose labels have changed after the semi-labeling step

The conductance measure

Spectral clustering targets the computation of clusters with a small conductance index as defined in Section 4.1.2 [22]. Thus, when optimizing for conductance, we implicitly hypothesize that smaller conductance should correspond to higher module enrichment. Figure 4 shows this correspondence.

We indeed have observed that there is correspondence between the cluster conductance index and cluster enrichment. Figure 4 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 6 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 Fig. 4a). Interestingly, from Fig. 4b, it can be seen that these clusters have higher enrichment.

As discussed in Sect. 2, 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 Fig. 5, where the general correlation between conductance and enrichment is evident.

SGCP hyperparameters and computation

SPGC requires the computation of a number of eigenvectors. The implementation includes an option that enables the fast iterative computation of the required subset of eigenvectors, thus keeping its runtime to levels comparable with WGCNA and other competing methods. Computing the Gene Ontology Enrichment is a computationally time-expensive task. The process for selecting k, described in the Network Clustering

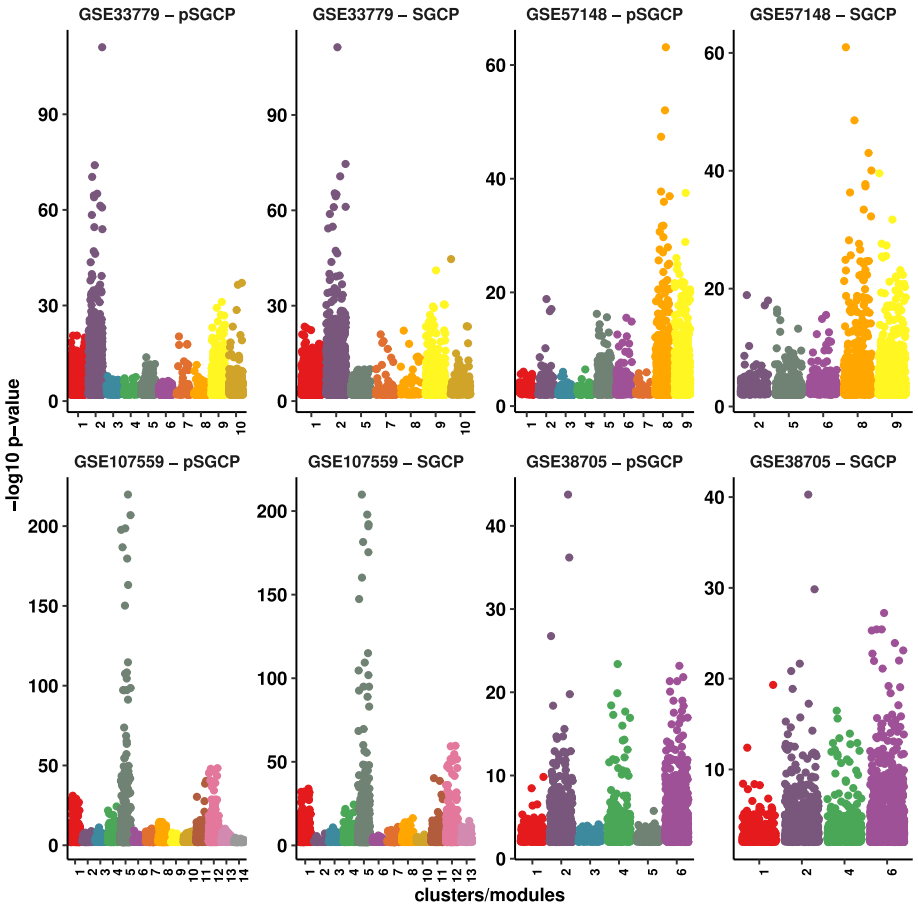

Fig. 3 Comparing pSGCP clusters and SGCP modules in 4 *real datasets*. In all cases, re-classification has resulted in a smaller number of modules relative to clusters. The labels of the eliminated clusters are 3, 4, 6, in GSE33779, 1, 3, 4, 10 in GSE57148, 9, 14 in GSE107559, and 3, 5 in GSE38705

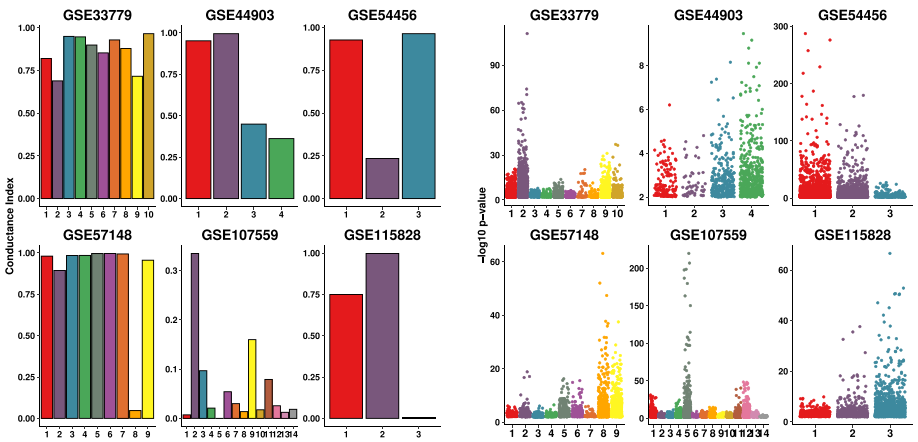

Fig. 4 Conductance index and log-transformed *p*-values *analysis in* 6 *real datasets*. For each data, the conductance index for the clusters (on the left) along with its corresponding log-transformed *p*-values distribution (on the right) is depicted. **a** Conductance index for each module per data. 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

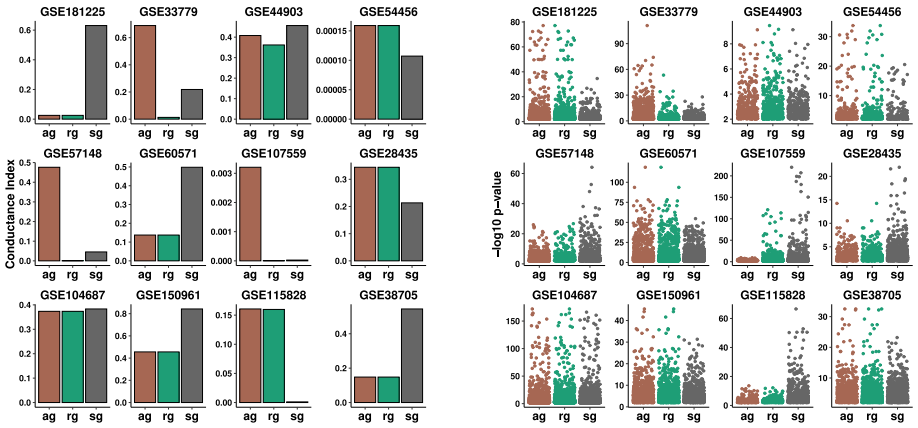

Fig. 5 *Conductance index and log-transformed p*-values analysis for additive gap ("ag"), relative gap ("rg"), and second-order gap ("sg") clusters in 12 *real datasets*. **a** Conductance index for the best cluster of each method on the 12 datasets. **b** log-transformed *p*-values of the selected clusters for "ag", "rg", and "sg" are shown. The higher the point, the more significant the GO term

step, is meant to reduce the amount of computation for the GO enrichment. However, SGCP enables the user to define their preferred number of clusters k. 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 of GSE44903 when our method does not outperform the baselines (see Sect. 2.2), a different choice of k does produce a 'winning' output for our framework. SGCP also includes a user-defined threshold about the percentage of GO terms used for finding remarkable genes and clusters.

Methods

SGCP methods

The input of SGCP 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.

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

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

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 use γ^2 coupled with the following exponential kernel for each pair of genes.

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

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) [13, 14]. The adjacency strength between gene i and j is calculated by the following formula:

$$a_{i,j} = \frac{l_{i,j} + s_{i,j}}{\min(k_i, k_j) + 1 - s_{i,j}}, \tag{3}$$

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.

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 define the scalar $a_i = 1^T DY_i/m$, where 1 is the all-ones vector. Then subtract t_i from each entry of Y_i.
- Let $Y_i := Y_i/(Y_i^T DY_i)^{1/2}$.
- Drop the first column of Y, as this is a trivial constant vector that does not affect the result.

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:

[2] The number of eigenvectors that is 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.

$$kag = \arg\max_i(\lambda_{i+1} - \lambda_i) \qquad \text{for} \quad i = 2, \ldots, m-1 \tag{4}$$

$$krg = \arg\max_i\left(\frac{1-\lambda_{i+1}}{1-\lambda_i}\right) \qquad \text{for} \quad i = 2, \ldots, m-2 \tag{5}$$

$$ksg = \arg\max_i\left(\frac{1-\lambda_{i+1}}{1-\lambda_i} - \frac{1-\lambda_{i+2}}{1-\lambda_{i+1}}\right) \qquad \text{for} \quad i = 2, \ldots, m \tag{6}$$

The quantities *kag*, *krg*, *ksg* are provisional values for the number of clusters. The final number of clusters is then calculated in the next steps.

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 [20] 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 Eq. 7.

$$conduct(C_i) = \frac{\sum_{u\in C_i, v\notin C_i} a_{u,v}}{\sum_{u\in C_i} deg(u)} \tag{7}$$

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 [43] 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:

$$\sum_{j\in c} -\log_{10}(P_j). \tag{8}$$

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 $Y'_{m\times 2k}$.

Step III: Gene ontology enrichment

The GOstats R package [43] 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.

Step IV: 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.

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) [49, 50] or logistic regression [51]. 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 [52] R-package.

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 1. The functions that are used for GCN construction and analysis in WGCNA, CoExpNets, and CEMiToo, are "blockwiseModules", "getDownstreamNetwork" and "cemitool" respectively.

Conclusion

We have proposed SGCP, a novel method for detecting modules of genes in gene co-expression networks. SGCP includes multiple features that differentiate it from existing frameworks and yields modules with significantly higher enrichment in Gene Ontology terms, on multiple benchmark datasets. SGCP identifies clusters whose most significant Gene Ontology terms are markedly different than those identified by WGCNA and other existing frameworks. SGCP's code is publicly available on Bioconductor, offering an alternative tool for gene co-expression analysis.

Acknowledgements
The authors would like to thank GWU Pr. Alexandros Tzatsos for helpful conversations.

Author contributions
N.A. and I.K. share an equal contribution in the design of the methods. N.A. authored the software, and designed and performed all experiments. Authors' Information Not applicable.

Funding
This work was partially supported by NSF grants CCF-2039863 and CCF-1813374.

Availability of data and materials
SGCP code, benchmark datasets and all results reported in Sect. 2 can be found in https://github.com/na396/SGCP. SGCP code is also available on Bioconductor (https://bioconductor.org/packages/devel/bioc/html/SGCP.html). The dataset identifiers are as follows: GSE33779 [30], DNA-microarray, provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE33779. GSE44903 [31], DNA-microarray, provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE44903. GSE28435 [36], DNA-microarray, provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE28435. GSE38705 [40], DNA-microarray, provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE38705. GSE181225 [29], RNA-sequencing, provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE181225. GSE54456 [32], RNA-sequencing provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE54456. GSE57148 [33], RNA-sequencing provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE57148. GSE60571 [34], RNA-sequencing provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE60571. GSE107559 [35], RNA-sequencing provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE107559. GSE104687 [37], RNA-sequencing provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE104687. GSE150961 [38], RNA-sequencing provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE150961. GSE115828 [39], RNA-sequencing provided at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE115828

Declarations

Ethics approval and consent to participate
Not applicable

Consent for publication
Not applicable

Competing interests
There are no competing interests.

Received: 26 April 2023 Accepted: 18 June 2024
Published online: 02 July 2024

References

1. Emamjomeh A, Saboori Robat E, Zahiri J, Solouki M, Khosravi P. Gene co-expression network reconstruction: a review on computational methods for inferring functional information from plant-based expression data. Plant Biotechnol Rep. 2017;11(2):71–86. https://doi.org/10.1007/s11816-017-0433-z.
2. Panahi B, Hejazi MA. Weighted gene co-expression network analysis of the salt-responsive transcriptomes reveals novel hub genes in green halophytic microalgae *Dunaliella salina*. Sci Rep. 2021;11(1):1607. https://doi.org/10.1038/s41598-020-80945-3.
3. Ma X, Zhao H, Xu W, You Q, Yan H, Gao Z, Su Z. Co-expression gene network analysis and functional module identification in bamboo growth and development. Front Genet. 2018;9:574. https://doi.org/10.3389/fgene.2018.00574.
4. Parsana P, Ruberman C, Jaffe AE, Schatz MC, Battle A, Leek JT. Addressing confounding artifacts in reconstruction of gene co-expression networks. Genome Biol. 2019;20(1):94. https://doi.org/10.1186/s13059-019-1700-9.
5. Liu J, Jing L, Tu X. Weighted gene co-expression network analysis identifies specific modules and hub genes related to coronary artery disease. BMC Cardiovasc Disord. 2016;16(1):54. https://doi.org/10.1186/s12872-016-0217-3.
6. Tieri P, Farina L, Petti M, Astolfi L, Paci P, Castiglione F. Network inference and reconstruction in bioinformatics. In: Ranganathan S, Gribskov M, Nakai K, Schönbach C, editors. Encyclopedia of bioinformatics and computational biology. Oxford: Academic Press; 2019. p. 805–13. https://doi.org/10.1016/B978-0-12-809633-8.20290-2.
7. Gat-Viks I, Sharan R, Shamir R. Scoring clustering solutions by their biological relevance. Bioinformatics. 2003;19(18):2381–9. https://doi.org/10.1093/bioinformatics/btg330.
8. Dam S, Võsa U, Graaf A, Franke L, Magalhães JP. Gene co-expression analysis for functional classification and gene-disease predictions. Brief Bioinform. 2017;19(4):575–92. https://doi.org/10.1093/bib/bbw139.
9. Aghaieabiane N, Koutis I. A novel calibration step in gene co-expression network construction. Front Bioinform. 2021. https://doi.org/10.3389/fbinf.2021.704817.
10. Khatri P, DrÃƒghici S. Ontological analysis of gene expression data: current tools, limitations, and open problems. Bioinformatics. 2005;21(18):3587–95. https://doi.org/10.1093/bioinformatics/bti565.
11. Botía JA, Vandrovcova J, Forabosco P, Guelfi S, D'Sa K, Consortium TUKBE, Hardy J, Lewis CM, Ryten M, Weale ME. An additional k-means clustering step improves the biological features of WGCNA gene co-expression networks. BMC Syst Biol. 2017;11(1), 47. https://doi.org/10.1186/s12918-017-0420-6.

12. Russo PST, Ferreira GR, Cardozo LE, Bürger MC, Arias-Carrasco R, Maruyama SR, Hirata TDC, Lima DS, Passos FM, Fukutani KF, Lever M, Silva JS, Maracaja-Coutinho V, Nakaya HI. CEMiTool: a bioconductor package for performing comprehensive modular co-expression analyses. BMC Bioinform. 2018;19(1):56. https://doi.org/10.1186/s12859-018-2053-1.
13. Zhang B, Horvath S. A general framework for weighted gene co-expression network analysis. Stat Appl Ggenet Mol Biol. 2005;4:1.
14. Langfelder P, Horvath S. WGCNA: an R package for weighted correlation network analysis. BMC Bioinform. 2008;9(1):559. https://doi.org/10.1186/1471-2105-9-559.
15. Petereit J, Smith S, Harris FC, Schlauch KA. Petal: co-expression network modelling in R. BMC Syst Biol. 2016;10(2):51. https://doi.org/10.1186/s12918-016-0298-8.
16. Godichon-Baggioni A, Maugis-Rabusseau C, Rau A. Clustering transformed compositional data using K-means, with applications in gene expression and bicycle sharing system data. J Appl Stat. 2019;46(1):47–65. https://doi.org/10.1080/02664763.2018.1454894.
17. Watson M. CoXpress: differential co-expression in gene expression data. BMC Bioinform. 2006. https://doi.org/10.1186/1471-2105-7-509.
18. Hou J, Ye X, Li C, Wang Y. K-module algorithm: an additional step to improve the clustering results of WGCNA co-expression networks. Genes. 2021;12(1):87. https://doi.org/10.3390/genes12010087.
19. Langfelder P, Zhang B, Horvath S. Defining clusters from a hierarchical cluster tree: the dynamic tree cut package for R. Bioinformatics. 2007;24(5):719–20. https://doi.org/10.1093/bioinformatics/btm563.
20. Hartigan JA, Wong MA. Algorithm AS 136: a K-means clustering algorithm. Appl Stat. 1979;28(1):100–8. https://doi.org/10.2307/2346830.
21. Cheng CW, Beech DJ, Wheatcroft SB. Advantages of CEMiTool for gene co-expression analysis of RNA-seq data. Comput Biol Med. 2020;125:103975. https://doi.org/10.1016/j.compbiomed.2020.103975.
22. Lee JR, Gharan SO, Trevisan L. Multiway spectral partitioning and higher-order cheeger inequalities. J ACM. 2014. https://doi.org/10.1145/2665063.
23. Bishop CM. Pattern recognition and machine learning (information science and statistics). Berlin: Springer; 2006.
24. Abbas-Aghababazadeh F, Li Q, Fridley BL. Comparison of normalization approaches for gene expression studies completed with high-throughput sequencing. PLoS ONE. 2018;13(10):0206312–0206312. https://doi.org/10.1371/journal.pone.0206312.
25. Barrett T, Wilhite SE, Ledoux P, Evangelista C, Kim IF, Tomashevsky M, Marshall KA, Phillippy KH, Sherman PM, Holko M, Yefanov A, Lee H, Zhang N, Robertson CL, Serova N, Davis S, Soboleva A. NCBI GEO: archive for functional genomics data sets-update. Nucleic Acids Res. 2013;41:991–5. https://doi.org/10.1093/nar/gks1193.
26. Irizarry RA, Hobbs B, Collin F, Beazer-Barclay YD, Antonellis KJ, Scherf U, Speed TP. Exploration, normalization, and summaries of high density oligonucleotide array probe level data. Biostatistics. 2003;4(2):249–64. https://doi.org/10.1093/biostatistics/4.2.249.
27. Lockhart DJ, Dong H, Byrne MC, Follettie MT, Gallo MV, Chee MS, Mittmann M, Wang C, Kobayashi M, Horton H, Brown EL. Expression monitoring by hybridization to high-density oligonucleotide arrays. Nat Biotechnol. 1996;14(13):1675–80. https://doi.org/10.1038/nbt1296-1675.
28. McCall MN, Bolstad BM, Irizarry RA. Frozen robust multiarray analysis (fRMA). Biostatistics (Oxford, England). 2010;11(2):242–53. https://doi.org/10.1093/biostatistics/kxp059.
29. Ruff SE, Vasilyev N, Nudler E, Logan SK, Garabedian MJ. PIM1 phosphorylation of the androgen receptor and 14-3-3 ζ regulates gene transcription in prostate cancer. Commun Biol. 2021;4(1):1221. https://doi.org/10.1038/s42003-021-02723-9.
30. Herranz R, Larkin OJ, Hill RJ, Lopez-Vidriero I, Loon JJ, Medina FJ. Suboptimal evolutionary novel environments promote singular altered gravity responses of transcriptome during *Drosophila metamorphosis*. BMC Evol Biol. 2013;13(1):133. https://doi.org/10.1186/1471-2148-13-133.
31. Theilhaber J, Rakhade SN, Sudhalter J, Kothari N, Klein P, Pollard J, Jensen FE. Gene expression profiling of a hypoxic seizure model of epilepsy suggests a role for mTOR and Wnt signaling in epileptogenesis. PLoS ONE. 2013;8(9):1–19. https://doi.org/10.1371/journal.pone.0074428.
32. Li B, Tsoi LC, Swindell WR, Gudjonsson JE, Tejasvi T, Johnston A, Ding J, Stuart PE, Xing X, Kochkodan JJ, Voorhees JJ, Kang HM, Nair RP, Abecasis GR, Elder JT. Transcriptome analysis of psoriasis in a large case-control sample: RNA-seq provides insights into disease mechanisms. J Invest Dermatol. 2014;134(7):1828–38. https://doi.org/10.1038/jid.2014.28.
33. Kim WJ, Lim JH, Lee JS, Lee S-D, Kim JH, Oh Y-M. Comprehensive analysis of transcriptome sequencing data in the lung tissues of COPD subjects. Int J Genom. 2015;2015:206937. https://doi.org/10.1155/2015/206937.
34. Chen Z-X, Oliver B. X chromosome and autosome dosage responses in *Drosophila melanogaster* heads. G3 (Bethesda Md). 2015;5(6):1057–63. https://doi.org/10.1534/g3.115.017632.
35. Puchalski RB, Shah N, Miller J, Dalley R, Nomura SR, Yoon J-G, Smith KA, Lankerovich M, Bertagnolli D, Bickley K, Boe AF, Brouner K, Butler S, Caldejon S, Chapin M, Datta S, Dee N, Desta T, Dolbeare T, Dotson N, Ebbert A, Feng D, Feng X, Fisher M, Gee G, Goldy J, Gourley L, Gregor BW, Gu G, Hejazinia N, Hohmann J, Hothi P, Howard R, Joines K, Kriedberg A, Kuan L, Lau C, Lee F, Lee H, Lemon T, Long F, Mastan N, Mott E, Murthy C, Ngo K, Olson E, Reding M, Riley Z, Rosen D, Sandman D, Shapovalova N, Slaughterbeck CR, Sodt A, Stockdale G, Szafer A, Wakeman W, Wohnoutka PE, White SJ, Marsh D, Rostomily RC, Ng L, Dang C, Jones A, Keogh B, Gittleman HR, Barnholtz-Sloan JS, Cimino PJ, Uppin MS, Keene CD, Farrokhi FR, Lathia JD, Berens ME, Iavarone A, Bernard A, Lein E, Phillips JW, Rostad SW, Cobbs C, Hawrylycz MJ, Foltz GD. An anatomic transcriptional atlas of human glioblastoma. Science. 2018;360(6389):660–3. https://doi.org/10.1126/science.aaf2666.
36. Spradling KD, Lumley LA, Robison CL, Meyerhoff JL, Dillman R, James F. Transcriptional analysis of rat piriform cortex following exposure to the organophosphonate anticholinesterase sarin and induction of seizures. J Neuroinflamm. 2011;8:83–83. https://doi.org/10.1186/1742-2094-8-83.
37. Miller JA, Guillozet-Bongaarts A, Gibbons LE, Postupna N, Renz A, Beller AE, Sunkin SM, Ng L, Rose SE, Smith KA, Szafer A, Barber C, Bertagnolli D, Bickley K, Brouner K, Caldejon S, Chapin M, Chua ML, Coleman NM, Cudaback E,

Cuhaciyan C, Dalley RA, Dee N, Desta T, Dolbeare TA, Dotson NI, Fisher M, Gaudreault N, Gee G, Gilbert TL, Goldy J, Griffin F, Habel C, Haradon Z, Hejazinia N, Hellstern LL, Horvath S, Howard K, Howard R, Johal J, Jorstad NL, Josephsen SR, Kuan CL, Lai F, Lee E, Lee F, Lemon T, Li X, Marshall DA, Melchor J, Mukherjee S, Nyhus J, Pendergraft J, Potekhina L, Rha EY, Rice S, Rosen D, Sapru A, Schantz A, Shen E, Sherfield E, Shi S, Sodt AJ, Thatra N, Tieu M, Wilson AM, Montine TJ, Larson EB, Bernard A, Crane PK, Ellenbogen RG, Keene CD, Lein E. Neuropathological and transcriptomic characteristics of the aged brain. eLife. 2017;6:31126. https://doi.org/10.7554/eLife.31126.

38. Mo A, Nagpal S, Gettler K, Haritunians T, Giri M, Haberman Y, Karns R, Prince J, Arafat D, Hsu N-Y, Chuang L-S, Argmann C, Kasarskis A, Suarez-Farinas M, Gotman N, Mengesha E, Venkateswaran S, Rufo PA, Baker SS, Sauer CG, Markowitz J, Pfefferkorn MD, Rosh JR, Boyle BM, Mack DR, Baldassano RN, Shah S, Leleiko NS, Heyman MB, Griffiths AM, Patel AS, Noe JD, Davis Thomas S, Aronow BJ, Walters TD, McGovern DPB, Hyams JS, Kugathasan S, Cho JH, Denson LA, Gibson G. Stratification of risk of progression to colectomy in ulcerative colitis via measured and predicted gene expression. Am J Hum Genet. 2021;108(9):1765–79. https://doi.org/10.1016/j.ajhg.2021.07.013.
39. Ratnapriya R, Sosina OA, Starostik MR, Kwicklis M, Kapphahn RJ, Fritsche LG, Walton A, Arvanitis M, Gieser L, Pietraszkiewicz A, Montezuma SR, Chew EY, Battle A, Abecasis GR, Ferrington DA, Chatterjee N, Swaroop A. Retinal transcriptome and eQTL analyses identify genes associated with age-related macular degeneration. Nat Genet. 2019;51(4):606–10. https://doi.org/10.1038/s41588-019-0351-9.
40. Bennett BJ, Farber CR, Ghazalpour A, Pan C, Che N, Wen P, Qi HX, Mutukulu A, Siemers N, Neuhaus I, Yordanova R, Gargalovic P, Pellegrini M, Kirchgessner T, Lusis AJ. Unraveling inflammatory responses using systems genetics and gene-environment interactions in macrophages. Cell. 2012;151(3):658–70. https://doi.org/10.1016/j.cell.2012.08.043.
41. Song L, Langfelder P, Horvath S. Comparison of co-expression measures: mutual information, correlation, and model based indices. BMC Bioinform. 2012;13(1):328. https://doi.org/10.1186/1471-2105-13-328.
42. Hu Y, Zhao H. CCor: a whole genome network-based similarity measure between two genes. Biometrics. 2016;72(4):1216–25. https://doi.org/10.1111/biom.12508.
43. Falcon S, Gentleman R. Using GOstats to test gene lists for go term association. Bioinformatics. 2006;23(2):257–8. https://doi.org/10.1093/bioinformatics/btl567.
44. Schaefer RJ, Michno J-M, Myers CL. Unraveling gene function in agricultural species using gene co-expression networks. Biochimica et Biophysica Acta Gene Regulat Mech. 2017;1860(1):53–63. https://doi.org/10.1016/j.bbagrm.2016.07.016.
45. Khanin R, Wit E. How scale-free are biological networks. J Comput Biol. 2006;13(3):810–8. https://doi.org/10.1089/cmb.2006.13.810.
46. Lima-Mendez G, Helden J. The powerful law of the power law and other myths in network biology. Mol Biosyst. 2009;5:1482–93. https://doi.org/10.1039/B908681A.
47. Broody AD, Clauset A. Scale-free networks are rare. Nat Commun. 2019;10(1):1017–1017. https://doi.org/10.1038/s41467-019-08746-5.
48. Clote P. Are RNA networks scale-free? J Math Biol. 2020;80(5):1291–321. https://doi.org/10.1007/s00285-019-01463-z.
49. Fix E, Hodges JL. Discriminatory analysis. Nonparametric discrimination: consistency properties. Int Stat Rev. 1989;57(3):238–47.
50. Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. Am Stat. 1992;46(3):175–85.
51. Peng C-YJ, Lee KL, Ingersoll GM. An introduction to logistic regression analysis and reporting. J Educ Res. 2002;96(1):3–14. https://doi.org/10.1080/00220670209598786.
52. Kuhn M. Building predictive models in R using the caret package. J Stat Softw. 2008;28(5):1–26. https://doi.org/10.18637/jss.v028.i05.

Publisher's Note