Detecting boolean asymmetric relationships with a loop counting technique and its implications for analyzing heterogeneity within gene expression datasets

Haosheng Zhou[1], Wei Lin[2], Sergio R. Labra[3], Stuart A. Lipton[4] Nicholas J. Schork[2], Aaditya V. Rangan[1*],

1 Courant Institute of Mathematical Sciences, New York University, New York, NY, United States of America
2 The Translational Genomics Research Institute, Quantitative Medicine and Systems Biology, Phoenix, AZ, United States of America
3 Scripps Research Institute, Department of Molecular Medicine and Chemistry, La Jolla, CA, United States of America
4 Scripps Research Institute, Neurodegeneration New Medicines Center, Department of Molecular Medicine, La Jolla, CA, United States of America

* avr209@nyu.edu

Abstract

Many traditional methods for analyzing gene-gene relationships focus on positive and negative correlations, both of which are a kind of 'symmetric' relationship. However, genes can also exhibit 'asymmetric' relationships, such as 'if-then' relationships used in boolean circuits. In this paper we develop a very general method that can be used to detect biclusters within gene-expression data that involve subsets of genes which are enriched for these 'boolean-asymmetric' relationships (BARs). These BAR-biclusters can correspond to heterogeneity that is driven by asymmetric gene-gene interactions, rather than more standard symmetric interactions. We apply our method to a single-cell RNA-sequencing data-set, demonstrating that the statistically-significant BAR-biclusters indeed contain additional information not present within more traditional 'boolean-symmetric'-biclusters. For example, the BAR-biclusters involve different subsets of cells, and highlight different gene-pathways within the data-set. Moreover, by combining the boolean-asymmetric- and boolean-symmetric-signals, one can build linear classifiers which outperform those built using only traditional boolean-symmetric signals.

Author summary

One important step when analyzing large data-sets is to search for 'heterogeneity', i.e., subsets of the data that exhibit special characteristics. In the context of gene-expression data, this heterogeneity can take the form of a 'bicluster': a subset of samples across which certain genes interact in a special way. Traditionally, strategies for detecting biclusters have focused on gene-gene relationships such as correlations and anti-correlations; that is to say, gene-subsets that act in concert. In this paper we discuss a simple strategy for detecting biclusters that exhibit more general gene-gene relationships, such as one gene serving as a prerequisite for the expression of another, or a pair of genes which mutually exclude one another. Using a data-set from a study of

Alzheimer's disease, we demonstrate that these more general biclusters can be quite statistically significant, and can contain novel information not readily found in more traditional biclusters.

Introduction

It is widely recognized that relationships between genes play an important role in biological research [1]. Generally speaking, most common methods for finding/identifying genetic relationships focus on positive and negative correlations, which we'll refer to below as ***symmetric relationships*** [2–4]. In this paper, however, we'll focus on ***asymmetric relationships***, which are a different kind of genetic relationship introduced and studied in [5–9].

Our main goal will be to introduce a very simple method which can be used to make any standard biclustering technique sensitive to these asymmetric relationships. This strategy can be used to detect heterogeneity within a gene-expression data-set, revealing subsets of samples and genes which exhibit certain asymmetric relationships not exhibited by the rest of the data. This analysis can identify subsets of samples which behave differently from the rest, and (simultaneously) identify the genes which drive those differences.

Boolean asymmetric relationship (BAR)

Using terminology from [5], 'asymmetric' relationships are direct generalizations of the more typical positive and negative correlations exhibited by 'symmetric' relationships. To illustrate an example, consider two random variables X and Y, each with M observations $x_m \in \mathbb{R}$ and $y_m \in \mathbb{R}$, respectively. Each of the observed points $(x_m, y_m) \in \mathbb{R}^2$ lies in one of the four quadrants of the plane. A typical 'symmetric' relationship between X and Y is defined as either an abundance of observed points in the first and third quadrants (i.e., positive correlation) or as an abundance of points in the second and fourth quadrants (i.e., negative correlation). Thus, if we denote the joint probability density of X and Y as $\rho(x, y)$, a symmetric relationship involves an elevation of $\rho(x, y)$ in opposite quadrants. By contrast, an 'asymmetric' relationship refers to any nonuniformity in $\rho(x, y)$ which is not inversion-symmetric, e.g., if $\rho(x, y)$ is elevated in only one of the four quadrants.

To be clear, the term 'symmetric' here does not mean conventional mathematical symmetry (i.e., that $\rho(x, y)$ is invariant under the exchange of x and y). Instead, in this context, the term 'symmetric' refers to inversion-symmetry. Namely, that $\rho(+x, +y)$ should be approximately equal to $\rho(-x, -y)$.

While there are many different kinds of asymmetric relationships, only certain asymmetric relationships can persist after centering the data (i.e., centralizing the margin for both X and Y, or ensuring that the sample mean of the x_m and y_m are each 0). Assuming the data is centered, an asymmetric relationship must involve an elevation of $\rho(x, y)$ in three quadrants (i.e., $\rho(x, y)$ can have only exactly one relatively vacant quadrant). An example of this kind of asymmetric relationship is shown in Fig 1.

In this paper we focus on a particular kind of asymmetric relationship: boolean asymmetric relationship (BAR), which has proven to be of interest for genetic analysis [5]. In this kind of relationship we assume that the random variables X and Y are categorical variables that can each take on up to two values. This can happen when X and Y are naturally binary, or after continuous variables $X, Y \in \mathbb{R}$ have been binarized based on a given threshold [5]. As an example, consider X an indicator of a person's gender, and Y an indicator of balding. The joint distribution $\rho(x, y)$ is almost limited to three quadrants, because y_m can seldom be positive if x_m is female.

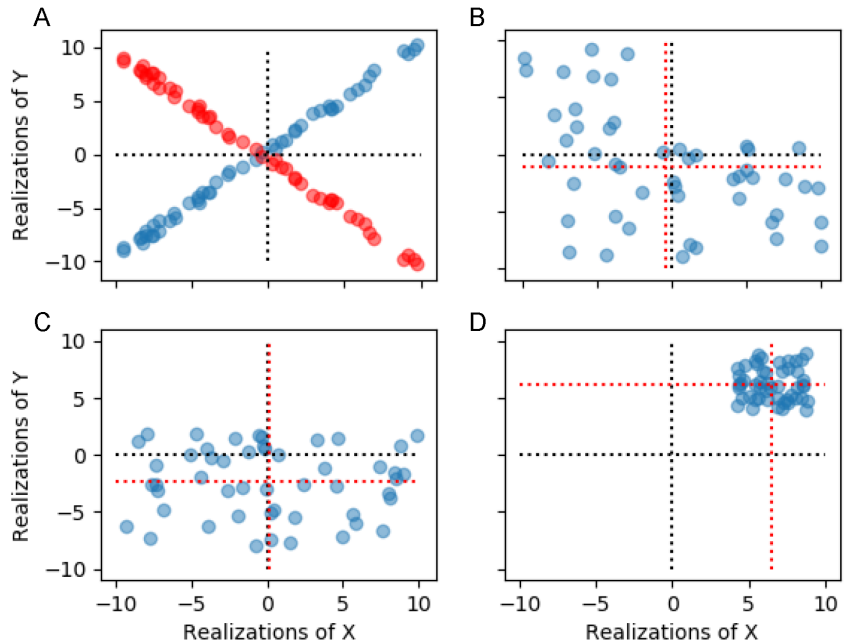


Fig 1. A plot for different kinds of relationships
(A) a positive correlation (blue) and a negative correlation (red) as symmetric relationships. (B) the kind of asymmetric relationships we are focusing on, with few points in the first quadrant. (C) and (D) asymmetric relationships under the standard coordinate axes (black dotted lines). However, they are no longer asymmetric if the points are centered with respect to the marginal means (red dotted lines).

Since each BAR involves underexpression in a single quadrant, there are four different kinds of BARs:

$Y \implies X$**:** This direction-specific interaction implies that X is necessary for Y. Put another way, Y cannot be high unless X is high (i.e., the 2nd quadrant is relatively empty).

$X \implies Y$**:** This is the converse of the previous interaction (i.e., the 4th quadrant is relatively empty).

X **'or'** Y**:** This two-way interaction implies that *either* X *or* Y is on (i.e, the 3rd quadrant is relatively empty).

X **'nand'** Y**:** This two-way interaction implies that *either* X *or* Y is off (i.e, 1st quadrant is relatively empty).

Note that the '$Y \implies X$' and '$X \implies Y$' relationships will reverse their directions if the x and y labels are interchanged; these two relationships are 'directed' in the sense that the ordering of x and y matters. Meanwhile, the 'or' and 'nand' relationships do

not change if the x and y labels are interchanged; these two relationships are 'undirected'.

As shown in [5], BARs are quite prevalent in gene-expression microarray data, often highly conserved across species, and are capable of identifying genes relevant for certain conditions. As alluded to above, the gene-gene BARs observed and discussed in [5] each suggest an associated genetic interaction; one gene can be necessary for the other, or the two genes can be linked in an 'or' or 'nand' pair. The ease with which these BARs can be interpreted makes it easier to link statistically-significant BARs to potential gene-gene interactions. These benefits suggest that BARs might serve as an important supplement to more traditional boolean symmetric relationships (BSR) when analyzing genetic data.

Prior work

Due to their importance and universality, BARs have been analyzed in many recent research efforts [6–9]. As just one example, [10] combined the use of a StepMiner algorithm and a sparsity test to find statistically-significant BARs, which can be generalized to find three-way- instead of merely pairwise-BARs as a way to reduce the computational burden and involve more general interactions between genes [11]. Wan et al. takes a different point of view, maximizing the coverage of the BARs under the constraints of a precision rate which is easy to apply for multinomial datasets [12]. Nation et al. investigates the conditional probability of overexpression between pairs of genes [13] and Wang et al. develop a two-step counting approach to construct biological pathways [14]. Chueh et al. proposes Boolean networks with time delay [15] and Hopfensitz et al. presents insights on the choice of binarization threshold that incorporates measurements at multiple resolutions [16].

Our motivation and contribution

To describe our contribution in more details, we first introduce some notation, with terminology taken from single-cell RNA-sequencing (scRNA-seq). Consider a data-set D including M samples (e.g., cells isolated in a scRNA-seq experiment), each observed across N variables (e.g., unique-molecular-identifiers, which we'll refer to as 'genes'). The value of the nth-gene observed within the mth-cell is stored at the array location D_{mn}.

Many of the existing analytical methods for detecting BSRs and/or BARs in genetic data emphasize relationships between gene-pairs that are statistically significant when considered across the entire cell-population. That is to say, they focus on identifying which of the $\frac{N(N-1)}{2}$ individual gene-pairs are significant when considered across all M of the cells. These pairwise relationships are certainly important, but are not the only important relationships that exist. Other important relationships include gene 'clusters' and 'biclusters'.

A cluster is a subset of genes for which each pair may only be weakly linked, but which together exhibit a collective signal (across all the cell-samples) that is strongly statistically significant. A ***bicluster***, on the other hand, is defined as subset of genes which only exhibits a collective signal across a subset of cells. Thus, a bicluster is a submatrix (defined in terms of a row- and column-subset) of the data-array D which exhibits a statistically significant structure which is not exhibited by the rest of the data-array. In the discussion below we'll use both terms 'cluster' and 'bicluster' to refer to biclusters.

Biclusters can highlight the drivers of heterogeneity within a data-set, which can be useful for developing new kinds of classifications, such as defining new cell-subtypes [21, 22, 24] and suggesting further investigation [17]. Generally speaking,

detecting all the statistically significant biclusters is very difficult, mainly because the number of possible biclusters is exponential in the number of cells and genes (i.e., there are 2^{M+N} different biclusters one could consider). Nevertheless, there are several biclustering methods which perform well in certain regimes [18–24]. Typically, these methods are built using strategies which either implicitly or explicitly search for BSRs between genes. Thus, the commonly available packages for biclustering gene expression data are designed to identify biclusters which exhibit an abundance of BSRs (i.e., 'BSR-biclusters'). See [19] for the Louvain Clustering algorithm, [20] for the UMAP method and see [21–24] for other methods which perform BSR-biclustering.

In this paper we develop a simple strategy for finding biclusters which exhibit an abundance of BARs (i.e., 'BAR-biclusters'). These BAR-biclusters can reveal heterogeneity which is distinct from that associated with BSR-biclusters. Our strategy boils down to a simple preprocessing step, where the original data is split to form a slightly larger dataset, which then can be passed to any standard biclustering algorithm that looks for BSR-biclusters.

As a proof of principle, we make use of the loop counting technique in [18] as a way of finding BSR-biclusters and generalize it to a new LCA algorithm that can find BAR-biclusters. We further use the LCA algorithm to analyze an organoid dataset by [25]. Using our new method, we demonstrate that BAR-biclusters contain information which can complement traditional BSR-biclustering.

The organoid dataset

As a proof of principle, we will apply our LCA algorithm to an scRNA-seq data-set studied in [25].

This data-set involves multiple cells sampled from organoids grown from patient-derived human induced pluripotent stem cells (hiPSCs). These organoids were designed to study Alzheimer's disease, and were prepared with three different isogenic types. The first is the baseline wild-type (WT), the second involves a mutation in the amyloid-precursor-protein (APP), and the third a mutation in presenilin-1 (PS1); both mutations are strongly associated with inherited alzheimer's disease. The organoids were allowed to develop (under conditions described in [25]) and are sampled at three different stages in development: 6-weeks, 3-months and 6-months. The cells sampled from the organoids were analyzed using scRNA-seq, and classified into cell-types by a supervised classifier trained on previous annotations (see [26–28]). Our analysis below will search for BAR-biclusters within the cell-types, identifying subsets of cells which exhibit a statistically-significant abundance of BARs across subsets of genes. As we'll show below, these BAR-biclusters contain information which is distinct from the typical BSR-biclusters found within the same dataset. As we'll discuss later on, we believe this extra information can be useful for basic tasks such as classification and identification, as well as for understanding the genetic drivers underlying the heterogeneity within the samples.

Materials and methods

Column splitting technique

The main computational technique we will use in this paper is 'column-splitting' (described below), which can transform any BAR-biclustering problem into a BSR-biclustering problem.

Consider the data-array $D \in \mathbb{R}^{M \times N}$ mentioned above. Let's assume, in keeping with [5], that some thresholds have already been chosen to convert each entry of D from

gene-expression values into categorical labels. Thus, each entry D_{mn} is either "low" (under-expressing), "high" (over-expressing) or "intermediate" (not significantly expressing, typically neglected). To describe our column-splitting technique, we'll use 'matlab' notation throughout the paper: referring to the nth-column of D as $D_{:,n}$.

Our column splitting technique proceeds as follows: Split each column of D into two columns to get a new binarized data-array $B \in \mathbb{R}^{M \times 2N}$. Each column $D_{:,n}$ from D will correspond to the two columns $B_{:,2n-1}$ and $B_{:,2n}$ in B. If D_{mn} is low, then $B_{m,2n-1} = 1$ and $B_{m,2n} = 0$. On the other hand, if D_{mn} is high, then $B_{m,2n-1} = 0$ and $B_{m,2n} = 1$. Thus, each column of D is split into one column indicating its 'low' state and one column indicating its 'high' state.

The technique is simple but effective. To illustrate how it works, assume that D has only two genes, indexed by columns $n = 1$ and $n = 2$, respectively. Furthermore, let's assume that genes 1 and 2 exactly satisfy a 'nand'-type BAR, implying that D_{m1} and D_{m2} each take on a range of labels, but cannot both be high simultaneously. Thus, the columns $B_{:,2}$ and $B_{:,4}$ (corresponding to the high-states of $D_{:,1}$ and $D_{:,2}$, respectively) must have a zero dot-product, while the column-pairs $(B_{:,1}, B_{:,3})$, $(B_{:,1}, B_{:,4})$ and $(B_{:,2}, B_{:,3})$ can all have nonzero dot-products. Thus, when contrasted against a scenario where genes 1 and 2 are independent random variables, the column-pair $(B_{:,2}, B_{:,4})$ exhibits a lower-than-expected correlation. Similarly, the column-pairs $(B_{:,1}, B_{:,3})$, $(B_{:,1}, B_{:,4})$ and $(B_{:,2}, B_{:,3})$ all exhibit a higher-than-expected correlation. A standard measurement of BSR can be used to detect these correlations amongst the columns of B, thus highlighting the BAR between the columns of D.

A similar statement holds more generally. Given any two distinct columns of D, say $D_{:,n}$ and $D_{:,n'}$, the corresponding columns of B form 4 nontrivial column-pairs: $(B_{:,2n-1}, B_{:,2n'-1})$, $(B_{:,2n-1}, B_{:,2n'})$, $(B_{:,2n}, B_{:,2n'-1})$ and $(B_{:,2n}, B_{:,2n'})$ (Note that $(B_{:,2n-1}, B_{:,2n})$ and $(B_{:,2n'-1}, B_{:,2n'})$ are orthogonal by construction). Each of the nontrivial column-pairs in B corresponds to a quadrant regarding the original columns of D. Each BAR corresponds to underexpression within a single quadrant. Thus, if the column-pair $(D_{:,n}, D_{:,n'})$ within D exhibits a BAR, then three of the the corresponding column-pairs within B will exhibit an elevated dot-product (i.e., a positively-correlated BSR) while the remaining quadrant will be underexpressed.

To sum up, the separation of "low" and "high" states turns the problem of finding BAR-biclusters into a problem of finding BSR-biclusters in a slightly larger matrix B. In principle, the matrix B can be processed by any algorithm which can detect BSR-biclusters; below we will use the specific BSR-biclustering algorithm in [18]. We also remark that, by splitting columns, the BSR-biclusters of B can indicate both BARs and BSRs within D. For this reason we will perform BSR-biclustering and BAR-biclustering simultaneously in our numerical experiments below; by comparing the results we will be able to isolate BAR-specific signals.

Loop counting low rank (LCLR)

As stated above, the column splitting technique is a very simple strategy for finding BAR-biclusters. This strategy can make use of any method which finds BSR-biclusters, but for demonstration we make use of the LCLR method for finding BSR-biclusters. This method is described in [18], and we review the important aspects here.

The LCLR method provides an approximate solution to the problem of finding submatrices with low numerical rank within a large matrix $D \in \mathbb{R}^{M \times N}$.

Given any binary matrix B, the LCLR method accumulates information about 'loops', referring to 2×2 submatrices within the binarized-array B. Because each entry B_{mn} is binary, there are only $2^4 = 16$ possibilities for each loop, and the ranks (i.e., dimensions) of each loop can be efficiently calculated and accumulated to form row- and column-scores.

The row-score for any row m is the number of rank-1 loops containing that row minus the number of rank-2 loops containing that row. This particular definition of the row-score is why LCLR is called a 'loop-counting' algorithm. The rank of any loop is determined by the product of its entries.

For example, given a single loop:

$$L = \begin{bmatrix} l_{mn} & l_{mn'} \\ l_{m'n} & l_{m'n'} \end{bmatrix}. \tag{1}$$

Then the product $l_{mn}l_{mn'}l_{m'n}l_{m'n'}$ is $+1$ if and only if the loop is of rank-1, and the product is -1 otherwise. So each loop of rank-1 contributes $+1$ to the score of the i-th row, while each loop of rank-2 contributes -1, consistent with the definition above for the score of a row.

By simple linear algebra, the score of the mth row can be written as:

$$\sum_{m',n,n'} l_{mn}l_{mn'}l_{m'n}l_{m'n'} = (BB^TBB^T)_{mm}. \tag{2}$$

Similarly, the score of the nth column can be written as:

$$(B^TBB^TB)_{nn}. \tag{3}$$

The LCLR algorithm proceeds iteratively: calculating the row- and column-scores and then eliminating those rows and columns with low scores, then recalculating the scores and eliminating more rows and columns until all rows and columns are eliminated. Those rows and columns which are retained the longest are highly likely to be part of a low-rank bicluster within the original data-set, provided that a bicluster of sufficient signal-strength exists (see S1 Appendix for more details, and [18] for statistical bounds relating the signal strength to the probability of detection).

Loop counting asymmetric (LCA)

As mentioned above, we can combine the column-splitting technique above with any BSR-biclustering algorithm in order to search for BAR-biclusters. Here we combine the column-splitting technique with the LCLR algorithm to form what we call the LCA (i.e., Loop Counting Asymmetric) algorithm. We choose the LCLR algorithm for two reasons. First, it has detection-thresholds which are similar to (or better than) many other commonly used biclustering methods, such as Louvain Clustering and UMAP [19, 20] (see supplementary information). Second, the LCLR-method naturally provides p-values for each bicluster as a measure of statistical significance (see [18] and [29]).

In contrast to the original LCLR algorithm, where the individual matrix-entries are drawn from $\{-1, +1\}$, in the LCA algorithm the entries of B are drawn from $\{0, +1\}$. Thus, for each loop of B, the score is still organized as the product of all four entries within, but with a different meaning: each nontrivial loop of B corresponds to a particular quadrant associated with two distinct columns of D. Consequently, each loop of B contributes to the score of three of the four different types of BAR associated with those two columns.

We refer to an individual application of the LCLR-method to the split data as Alg 1:

The complete BAR-biclustering algorithm, based on multiple applications of Alg 1, is presented below in Alg 2. This second algorithm extracts multiple LCA-biclusters from the data matrix, using the automatic bicluster selection technique described below in the 'Data-Analysis' section.

In our numerical experiments below we'll use both the LCA- and the original LCLR-algorithm. By comparing these results with one another we'll be able to determine the degree to which any particular bicluster is BAR- or BSR-specific.

Algorithm 1 Loop Counting Asymmetric (LCA) row- and column-ordering

Input: Data Matrix $D_{M\times N}$ and an 'elimination fraction', which we fix as 5% in step #5 below (see [29])

1: Assign one of the three labels: "low", "intermediate", "high" to each entry of D and split each column of D into two to get the binarized data matrix $B_{M\times 2N}$ whose entries take values $0, 1$

$$\forall \;\; k = 1, 2, ..., N, \qquad B_{mn} = \begin{cases} 1 & n = 2k, \; D_{mk} \text{ is high} \\ 1 & n = 2k-1, \; D_{mk} \text{ is low} \\ 0 & \text{else} \end{cases}$$

2: **repeat**

3: $RowScore = diag(BB^TBB^T)$

4: $ColScore = diag(B^TBB^TB)$

5: Sort $RowScore$ and eliminate 5% of the rows with the lowest scores from B. Do the same with the $ColScore$, eliminating 5% of the columns with the lowest scores from B as well.

6: **until** B runs out of rows or columns

Output: A list of integers for each row and column of B indicating in which iteration it is eliminated from B in the loop above

Algorithm 2 LCA-Biclustering

Input: Data Matrix $D_{M\times N}$

1: Normalization for matrix D

2: **repeat**

3: Run Alg 1 for matrix D to figure out in which iteration each row and column of B is eliminated

4: Use the automatic bicluster-selection technique (discussed below) to pick out an LCA-bicluster of B, along with its associated p-value.

5: Remove the rows and columns associated with the LCA-bicluster from D.

6: **until** no more biclusters are statistically-significant.

Output: A collection of LCA-biclusters and associated p-values.

Gene set enrichment analysis (GSEA)

In our numerical experiments below, we use both LCA and LCLR to analyze an snRNA-seq data-set from [25]. As we mentioned above, these two biclustering methods search for different kinds of biclusters and reveal different kinds of information within the data-set. To illustrate this point, we collect the gene-subsets of the statistically significant BAR- and BSR-biclusters that we find, and use gene set enrichment analysis (GSEA) to highlight the statistically significant gene-pathways that are associated with each bicluster. As we'll show below, this GSEA returns different subsets of pathways for the different biclusters. Importantly, there are many significant pathways associated with the BAR-biclusters that are not associated with the corresponding BSR-biclusters, indicating that the former contains information not found in the latter.

In terms of implementation, we use the GSEA software from [30]. For each pathway, we compare the enrichment p-values for (i) the BAR-biclusters (obtained with the LCA-algorithm) to the enrichment p-values of (ii) the trivial set of all genes analyzed, and (iii) the BSR-biclusters obtained with the LCLR-algorithm. We expect to find pathways which have more significant enrichment p-values for (i) than for (ii) and (iii).

Data Analysis

To illustrate the kinds of BAR-biclusters one might see in practice, and to show that these BAR-biclusters are distict from the more traditional BSR-biclusters, we apply our LCA-method to the scRNA-seq dataset described in [25]. As mentioned above, this data-set comprises 8 different kinds of cell-types drawn from multiple organoids, some of which have isogenic mutations which are commonly seen in hereditary forms of Alzheimer's disease. The cells within each cell-type are further divided into 3 stages of development, corresponding to the time that the organoid was allowed to grow before being sequenced.

Fix cell type

We apply our biclustering algorithms to each set of samples from each of the cell-types. Our goal will be to find statistically significant BAR- and BSR-biclusters within each cell type. While it is possible for these biclusters to extend across all isogenic variants and all stages of development, we expect at least a few of the biclusters to be biased towards cells from a particular isogenic variant or developmental stage. These isogenic- or stage-specific biclusters can be used to highlight genes or gene-pathways which drive heterogeneity within the data-set, and might be linked to Alzheimer-specific genetic mutations and/or development.

Normalize and binarize

The normalization process mainly consists of the truncation of genes, based on which binarization is done. In this particular data-set the vast majority of genes are either not expressed at all, or are rarely expressed across any of the cells in the sample. To focus on those genes that carry the most signal, we limit our analysis to the top 300 genes (approximately the top 1% genes) in terms of overall expression (after eliminating outlier genes with spurious expression values). This choice was made only for clarity of presentation, and our results do not change qualitatively if we include more genes.

As discussed in [5], the identification of BARs can depend on the choice of threshold used to define 'low', 'intermediate' and 'high'. While more sophisticated studies of BARs can attempt to select different thresholds for different genes, for our proof-of-principle example we use the same thresholding strategy for all genes. Namely,

for each gene, we define the lowest 15% of expression-values to be 'low', and the highest 15% of expression-values to be 'high', with the remainder classified as 'intermediate'. For normally-distributed data this simple strategy corresponds to defining a threshold proportional to the standard-deviation of the expression-data, as suggested in [5]. Moreover, this simple strategy produces a B-matrix which is appropriately centered for subsequent analysis. Once again, our approach is quite robust and the results do not change qualitatively if we choose different thresholds.

LCA & LCLR

The LCA algorithm is then applied for the binarized matrix, with the results in each iteration recorded. See Fig 2 for the visualization of LCA and LCLR.

We apply both LCA and LCLR to each cell-type array. As described above, the LCA algorithm can reveal both BAR-biclusters as well as BSR-biclusters, while the LCLR algorithm can only reveal biclusters with a significant BSR-signal. By comparing the results of these two algorithms, we can evaluate how strongly any particular bicluster is driven by BAR- versus BSR-signals (see the definition of the MSC ratio below).

In Fig 2, we show an example of the output obtained after running the LCA- and LCLR-algorithms on the data from the Astroglial cells. For this figure we perform a single pass for each algorithm, recording the row- and column-scores in each case. After using the scores to reorganize the rows and columns in descending order, we clearly see significant biclusters in the data (see the block-structure in Fig 2C and the low-rank structure in Fig 2D, highlighted by the red lines). These correspond, respectively to a large BAR- and BSR-bicluster. In this example we identify these two biclusters 'by hand', while a more robust and reproducible bicluster selection technique will be described later. We calculate the dominant principal-component (PC) of the BAR- and BSR-bicluster, and project each cell's expression data onto these two PCs. The resulting scatterplots are shown in Figs 2E and 2F. These scatterplots are colored by bicluster membership (see legend). Clearly, the subset of cells within the BAR-bicluster is different than the subset of cells within the BSR-bicluster, indicating that the BAR-biclustering method has found a distinct signal within the data.

Automatic bicluster selection

While the biclusters shown in Fig 2 are quite easily delineated by hand, many of the biclusters observed in a typical data-set will be less clear. Consequently, we need an automated method for determining the 'boundary' of each bicluster: i.e., which rows and columns define it.

For this purpose, we measure the mean-squared-correlation (MSC) at each iteration of the LCA- and LCLR-algorithms [29].

For any matrix A with size $m \times n$, its MSC is specified as:

$$MSC(A) = \frac{\sum_{i,j=1}^{n} \langle A_{:,i} A_{:,j} \rangle^2}{(mn)^2}. \tag{4}$$

where $\langle \cdot, \cdot \rangle$ is the standard inner product. Note that the MSC can be re-expressed as:

$$MSC(A) = \frac{1}{m^2 n^2} \sum_{i,i',j,j'} A_{i,j} A_{i,j'} A_{i',j'} A_{i',j}. \tag{5}$$

which is proportional to the sum of the row- or column-scores calculated by the LCLR method.

For matrices D with entries in $\{-1, 0, +1\}$, the $MSC(D)$ is a direct measure of the average squared-correlation between any two rows or columns of D; the MSC will be

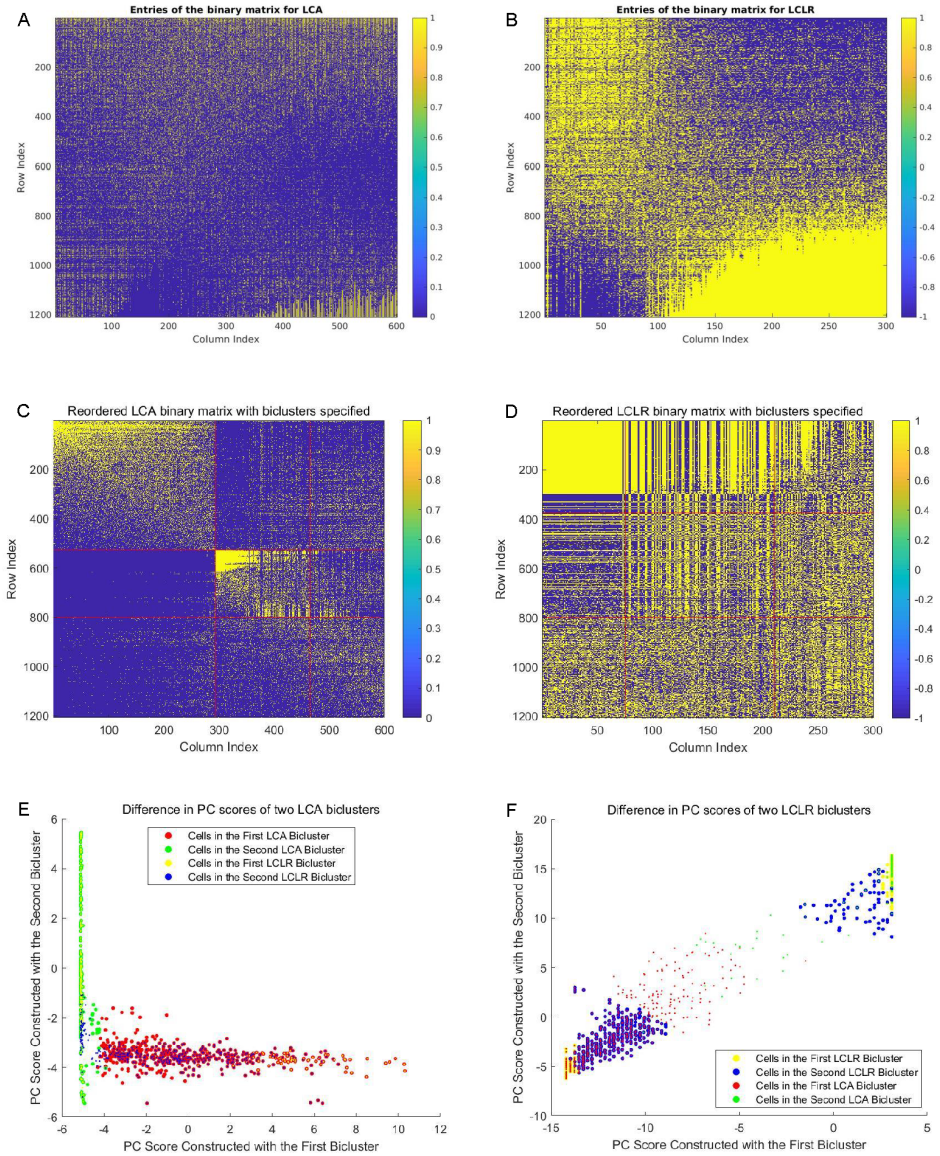


Fig 2. Visualization of LCA and LCLR
(A) the binarized matrix for LCA. (B) the binarized matrix for LCLR. (C) the reordered binarized matrix for LCA. (D) the reordered binarized matrix for LCLR. (E) the PC scores for LCA-biclusters. (F) the PC scores for LCLR-biclusters.
(A), (B) show the structure of raw matrix with no clear structures. (C), (D) show certain structure of the reordered matrix, with two clusters specified. (E), (F) exhibit clear boundary between PC scores constructed with two different clusters of the cells within, showing the effectiveness of LCA and LCLR. As can be seen, the membership of the biclusters of LCA and LCLR are quite different. Note that larger points stand for base group sample points while smaller points stand for contrast group sample points.

high if the matrix D is numerically low-rank. For binary matrices B with entries in $\{0,1\}$, the $MSC(B)$ will be high if many of the rows or columns of B 'overlap' (i.e., share many nonzero entries). Because of way we split the columns of the original data D to form B, each pair of overlapping columns of B corresponds to a particular quadrant associated with two genes of D; the MSC of B will be high if the matrix B is associated with an abundance of BSRs or BARs.

Here we remark that, because the MSC serves as an indicator of both BSRs and BARs, it is natural to measure $MSC(B)$ at each iteration of Alg 1. By comparing this $MSC(B)$ with a similar measurement performed on the appropriate subset of the original data-matrix D, we should be able to detect if the matrix B has an abundance of BARs. We make this intuition more formal below, first introducing some notation.

Let's define D to be (thresholded) data-matrix. Recall that D has entries in $\{-1,0,+1\}$, and when we run the LCLR-algorithm we use D. Let's also define B to be the binary matrix used in the LCA-algorithm. Recall that B has entries in $\{0,1\}$, and is obtained by splitting the columns of D. We denote this relationship via the 'column-map' μ:

$$\mu(n) = \left\lceil \frac{n}{2} \right\rceil . \tag{6}$$

such that any column n of B corresponds to the gene associated with column $\mu(n)$ of D.

Now, when running Alg 1 on B, we iterate multiple times, removing rows and columns with each iteration. Let l denote the iteration-index, and $\mathcal{M}(l;B)$ denote the row-subset of B remaining after iteration l. Similarly, let $\mathcal{N}(l;B)$ denote the column-subset of B remaining after iteration l. Finally, let's refer to $B(l)$ as the sub-matrix of B remaining after iteration l. That is, using 'matlab-notation':

$$B(l) = B_{\ \mathcal{M}(l;B)\ ,\ \mathcal{N}(l;B)}. \tag{7}$$

In order to extract the sub-matrix of D that corresponds to $B(l)$ we will use the same row-subset, as well as the corresponding genes, this subset of genes in D can be constructed via the column-map: $\mu(\mathcal{N}(l;B))$. Note that this column-map can (and often does) produce fewer columns than were originally in $\mathcal{N}(l;B)$.

Now, using the row-subset from $\mathcal{M}(l;B)$ and the gene-subset from $\mu(\mathcal{N}(l;B))$, we will define:

$$D(l) = D_{\ \mathcal{M}(l;B)\ ,\ \mu(\mathcal{N}(l;B))}.$$

Finally, we define the MSC-ratio as:

$$\text{MSC-Ratio}(l) = \frac{MSC(B(l))}{MSC(D(l))}. \tag{8}$$

In practice, after we run Alg 1, we measure the MSC-ratio for each iteration l.

We assess this empirical MSC-ratio under the null-hypothesis that the data-array is unstructured. To draw a sample from this null-hypothesis, we permute the entries of the normalized data-matrix and rerun the LCA-algorithm on the permuted data. This produces one sample trial from the null-hypothesis, and we run several such trials to estimate the distribution of the MSC-ratio under the null-hypothesis. Once we have sufficiently many trials, we can use the distribution of MSC-ratios under the null-hypothesis to determine the (empirical) p-value of the MSC-ratio of the original data. We can also use use the permuted trials to delineate the boundaries of the bicluster within the original data. To be more specific, we first calculate Z-scores for the empirical MSC ratio, with respect to the distribution of MSC ratios drawn from the null-hypothesis (denoted by MSC_Ratio_Zscore(l)). Once we have computed the Z-scores, we define the bicluster using the iteration l_{opt} which maximizes this Z-score:

$$l_{\text{opt}} \coloneqq \arg\max \text{MSC_Ratio_Zscore}(l). \tag{9}$$

As a result, $B(l_{\text{opt}})$ is the bicluster to select. See Fig 3 for an illustration.

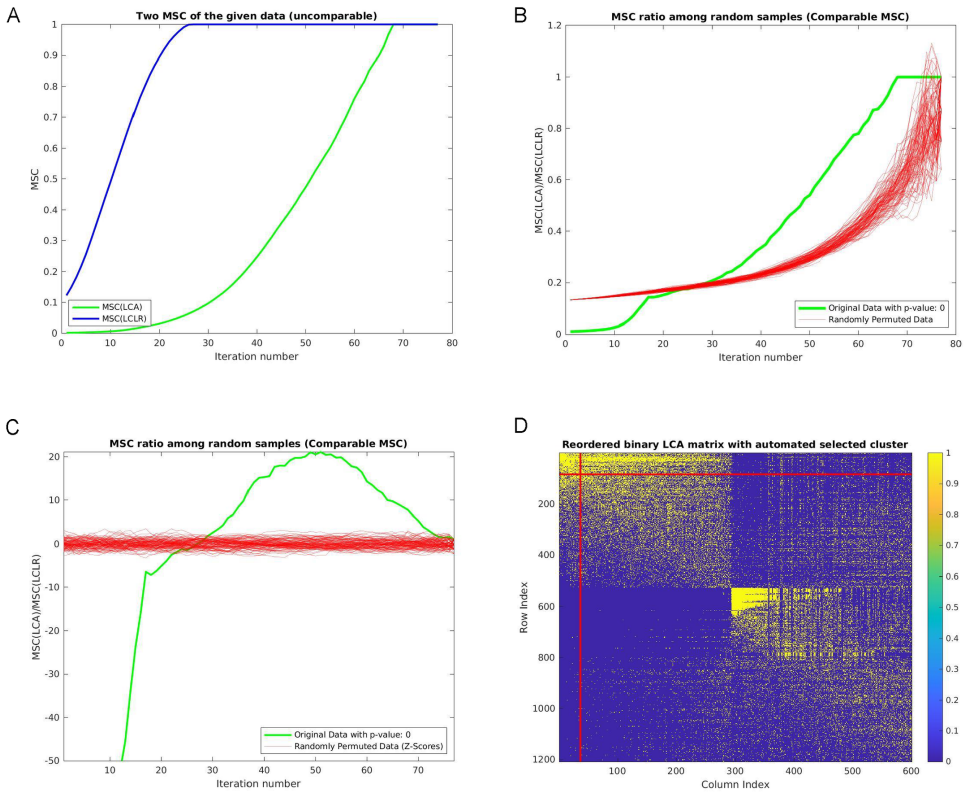


Fig 3. Visualization of automatic bicluster selection
(A) the MSC behavior for LCA and LCLR. (B) the MSC ratio behavior and the randomly generated samples. (C) the Z-scores of the the MSC ratio shown in (B). (D) the automatic bicluster selection result.
As (A) illustrates, for the loop-counting algorithm, the MSC always increases and ends up at 1. (B), (C) show curves of the MSC ratio for the given dataset (green) and 100 randomly sampled datasets (red). (D) shows the selected bicluster as the upper left corner of the two red straight lines.

Pearson test

After delineating the boundaries of the LCA-bicluster, we use the information from the isogenic- and stage-labels (which were not used in the LCA- or LCLR-algorithms) to test whether the bicluster is enriched for any particular isogenic-variant or developmental-stage. The null hypothesis is that the isogenic- and stage-labels are uncorrelated with bicluster membership.

A Pearson test applied to this hypothesis gives the following results in Fig 4:

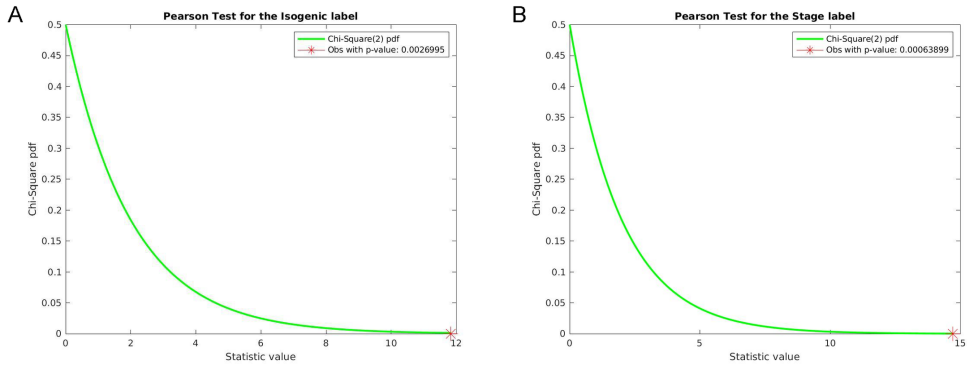


Fig 4. Result of Pearson test
(A) the result of Pearson test on isogenic labels. (B) the result of Pearson test on stage labels.
The green curve represents the χ^2 density while the red star represents the value of statistic computed for the selected cluster. In this case, two tests are both statistically significant at the level $\alpha = 0.01$

In the case above, the distributions of Isogenic and Stage labels within the selected LCA cluster are both statistically different from those within the whole dataset, implying that the BAR-bicluster illustrated in this example is indeed specific to certain isogenic-variants and certain developmental-stages. The specific enrichment of this BAR-bicluster is also different from the enrichment of the BSR-biclusters found within the same data-set.

GSEA results

To further investigate the BAR-biclusters found by the LSA-algorithm, we use GSEA on each of the respective gene-sets. We use the 'seek' software from [30], referencing the gene ontologies `go_bp_iea`, `go_cp_iea`, and `go_mf_iea`, which have each been curated by the seek developers. Our baseline level of significance is calculated using LCLR or by simply inputting all relevant genes we analyzed. We only list pathways that are more significant in the BAR-biclusters than in the corresponding BSR-biclusters.

It is typical for the LCA algorithm to identify gene-subsets which are more strongly enriched than the LCLR algorithm. One example, for a specific set of biclusters, is shown in Table 1 and 2.

Construction of boolean asymmetric relationship network (BARN)

One of the advantages of BAR-signals and BAR-biclusters is that the individual BARs are easily interpreted. As mentioned above, there are 2 different kinds of BSRs: namely positive and negative correlations. There are also four more kinds of BAR-specific interactions:

Table 1. Part of GSEA results on LCA-biclusters compared to trivial case

Gene Ontology	Gene Pathway	LCA-bicluster	trivial case
go_bp_iea	Negative Regulation of Transcription DNA Dependent	5.9×10^{-5}	1.1×10^{-2}
	Generation of Neurons	4.4×10^{-4}	-
go_cp_iea	Clathrin Sculpted Vesicle	5.7×10^{-4}	2.7×10^{-3}
	Neuron Part	1.9×10^{-5}	-
go_mf_iea	Structural Constituent of Cytoskeleton	3.2×10^{-5}	1.1×10^{-4}
	GTP Dependent Protein Binding	6.1×10^{-4}	-

For each gene ontology, two pathways are selected to support our conclusions. With the pathway above to be more significant in LCA-biclusters and the pathway below to be only appearing in LCA-biclusters. The last two columns show respective p-values of gene pathways.

Table 2. Part of GSEA results on LCA-biclusters compared to LCLR-biclusters

Gene Ontology	Gene Pathway	LCA-bicluster	LCLR-bicluster
go_bp_iea	Negative Regulation of RNA Metabolic Process	8.3×10^{-5}	1.6×10^{-3}
	Negative Regulation of Transcription DNA Dependent	5.9×10^{-5}	-
go_cp_iea	Mitochondrial Envelope	2.6×10^{-9}	1.8×10^{-6}
	Nuclear Chromosome Telomeric Region	3.5×10^{-3}	-
go_mf_iea	Macromolecule Transmembrane Transporter Activity	9.3×10^{-4}	1.1×10^{-3}
	Oxidoreductase Activity	3.6×10^{-5}	-

For each gene ontology, two pathways are selected to support our conclusions. With the pathway above to be more significant in LCA-biclusters and the pathway below to be only appearing in LCA-biclusters. The last two columns show respective p-values of gene pathways.

$Y \implies X$**:** This direction-specific interaction implies that X is necessary for Y. Put another way, Y cannot be high unless X is high (i.e., the 2nd quadrant is relatively empty).

$X \implies Y$**:** This is the converse of the previous interaction (i.e., the 4th quadrant is relatively empty).

X **'or'** Y**:** This two-way interaction implies that *either* X *or* Y is on (i.e, the 3rd quadrant is relatively empty).

X **'nand'** Y**:** This two-way interaction implies that *either* X *or* Y is off (i.e, 1st quadrant is relatively empty).

With these interpretations in mind, we can represent each of the biclusters produced by our LCA-analysis as a gene-gene interaction network. Unlike more traditional network representations (which only illustrate correlations and anti-correlations), or diagrams also highlight the BARs between the genes involved. A simple cartoon of a BARN (indicating the symbols used for each interaction) is shown in Fig 5, with subsequent Figs 6 and 7 illustrating the BARNs corresponding to two of our biclusters.

Note that the LCA algorithm focuses on biclusters with an overall abundance of BARs; the LCA algorithm by itself will not necessarily identify any particular BAR between any two genes. To determine which particular BARs are over-represented in any of the statistically-significant BAR-biclusters, we apply the following procedure.

First, recall that gene n of the original matrix D is split into columns $2n-1$ and $2n$ within the binary-matrix B, corresponding to values of $D_{:,n}$ being negative and positive, respectively. Given this construction, let's label the columns $2n-1$ and $2n$ (within B) as n_- and n_+, respectively.

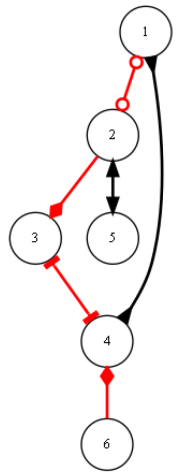


Fig 5. Different types of edges in BARN
The edge between 2 and 5 is a positive correlation, while the edge between 1 and 4 is a negative correlation. Both of them are BSRs, represented by an edge with the same arrowhead and arrowtail.
The edge between 1 and 2 is a 'nand' relationship, the edge between 3 and 4 is an "or" relationship, both undirected BARs. The edge from 2 to 3 represents $2 \implies 3$, the edge from 4 to 6 represents $6 \implies 4$, both directed BARs.
Red edges are BARs while black edges are BSRs.

Now, given any pair of genes n and n' within D, there will be 4 corresponding columns of B, denoted by n_-, n_+, n'_- and n'_+. These four columns of B in turn correspond to four nontrivial column-pairs: $\{n_+, n'_+\}$, $\{n_-, n'_+\}$, $\{n_-, n'_-\}$ and $\{n_+, n'_-\}$ (note that the other two column-pairs $\{n_-, n_+\}$ and $\{n'_-, n'_+\}$ are orthogonal by construction). These nontrivial column-pairs correspond, respectively, to quadrants 1 through 4 of the two-dimensional plane associated with genes n and n'.

The inner-product $\langle B_{:,n_+}, B_{:,n'_+} \rangle$ measures the (empirically observed) number of observations of the $\{n, n'\}$ gene-pair which fall into quadrant 1. Similarly, the inner-product $\langle B_{:,n_-}, B_{:,n'_+} \rangle$ measures the number of observations which fall into quadrant 2, and so forth. Thus, given any gene-pair $\{n, n'\}$ and any bicluster, we can use the inner-products of the appropriately chosen columns of B (within the bicluster) to determine the empirical joint-distribution $\rho_{n,n'}(x, y)$ corresponding to the two-dimensional plane associated with that gene-pair (i.e., across values of x and $y \in \{-, +\}$).

If we are to observe a joint-distribution with the shape:

$$\rho_{BSR+}(x, y) = \begin{cases} 1/2 & (x, y) = (+, +) \text{ i.e., quadrant 1} \\ 0 & (x, y) = (-, +) \text{ i.e., quadrant 2} \\ 1/2 & (x, y) = (-, -) \text{ i.e., quadrant 3} \\ 0 & (x, y) = (+, -) \text{ i.e., quadrant 4} \end{cases}. \tag{10}$$

then that joint-distribution would be a perfect BSR, corresponding to 'positively correlated' observations within the bicluster. Similarly, if the joint-distribution were to

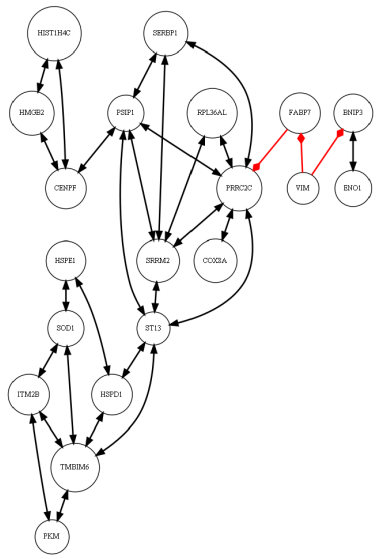


Fig 6. A BARN example
The vertices represent genes, with gene names specified. Only the most relevant edges are drawn in the network, i.e. those with total-variation-values in the bottom 5%.
The edges represent BAR or BSR as defined above, red edges are BARs while black edges are BSRs.
This BARN is constructed based on the second bicluster of Glutamatergic neurons.

look like:

$$\rho_{BSR-}(x,y) = \begin{cases} 0 & (x,y) = (+,+) \text{ i.e., quadrant 1} \\ 1/2 & (x,y) = (-,+) \text{ i.e., quadrant 2} \\ 0 & (x,y) = (-,-) \text{ i.e., quadrant 3} \\ 1/2 & (x,y) = (+,-) \text{ i.e., quadrant 4} \end{cases}. \tag{11}$$

then the gene-pair would be perfectly 'negatively correlated' within the bicluster.

Continuing in the same vein, we can define the idealized distribution:

$$\rho_{BAR1}(x,y) = \begin{cases} 0 & (x,y) = (+,+) \text{ i.e., quadrant 1} \\ 1/3 & (x,y) = (-,+) \text{ i.e., quadrant 2} \\ 1/3 & (x,y) = (-,-) \text{ i.e., quadrant 3} \\ 1/3 & (x,y) = (+,-) \text{ i.e., quadrant 4} \end{cases}. \tag{12}$$

as the BAR for which quadrant-1 is vacant (i.e., the 'nand' BAR). We can define the idealized joint-distributions ρ_{BAR2} through ρ_{BAR4} for the other quadrants analogously.

To assign a relationship to any particular gene-pair $\{n, n'\}$, we measure the empirical joint-distribution $\rho_{n,n'}$ as described above, and then compare $\rho_{n,n'}$ to each of the six different idealized joint-distributions ρ_{BSR+}, ..., ρ_{BAR4}. We use the idealized joint-distribution closest to the empirical joint-distribution to determine the type of BSR or BAR to assign to that gene-pair, using the total-variation as a measure of the difference between distributions. We also record the total-variation associated with each BSR- or BAR-label; by thresholding these total-variation values we can visualize a pruned BARN retaining only the most relevant edges.

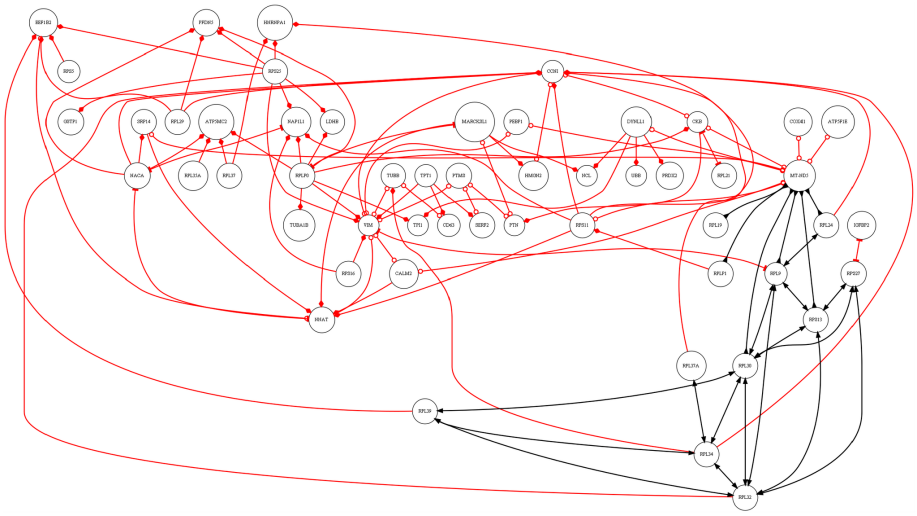


Fig 7. A BARN example with more BAR shown
The vertices represent genes, with gene names specified. Again, only the 5% most relevant edges are shown.
The BAR structure in this BARN is more complicated than that in the last plot. This BARN is constructed based on the fifth bicluster of GABAergic neurons.

A prediction use case

Last but not least, we demonstrate that the BAR-signals discovered by the LCA-algorithm can augment the standard LCLR-algorithm and improve classification. As a first step, we randomly divide the original data-set into two groups of cells (i.e., by row). We will use one of these groups as a 'training' group for the LCA- and LCLR-algorithms. After identifying biclusters within the training group, we use the BSR-signals within these biclusters to build a BSR-only linear-classifier for the isogenic- and stage-labels within the training group. We also use the BSR- and BAR-signals together to build a BAR + BSR linear-classifier for the training group. To simplify the model, we only consider a binary classification problem, i.e. fix a Stage label denoted as Stage 1, any prediction would be "is Stage 1" or "is not Stage 1". Finally, we validate these two linear classifiers within the second 'testing' group. As expected, the linear-classifier built using both the BAR- and BSR-signals outperforms the BSR-only classifier (see Fig 8). While this is only a single example, we see similar results throughout our analysis. The same general trend holds: BSR-signals provide extra information which can be used to augment BSR-signals and build better classifiers.

In terms of details, we build a linear-classifier by first measuring the dominant principal-component (PC) of each statistically significant BAR- and BSR-bicluster within the training set. Then, each cell is projected onto these PCs to give a bicluster-specific score to each cell. We then use logistic-regression to link these bicluster-specific scores to the ground-truth isogenic- or stage-label (within the training data). Finally, we use the output of this logistic-regression (i.e., the logistic link function) as a predictor of label within the testing data.

The linear-classifier we build using information from both the BAR- and BSR-biclusters outperforms the classifier built using only the BSR-biclusters. Shown in Fig 8 is the AUC-curve associated with each classifier. We remark that our strategy is

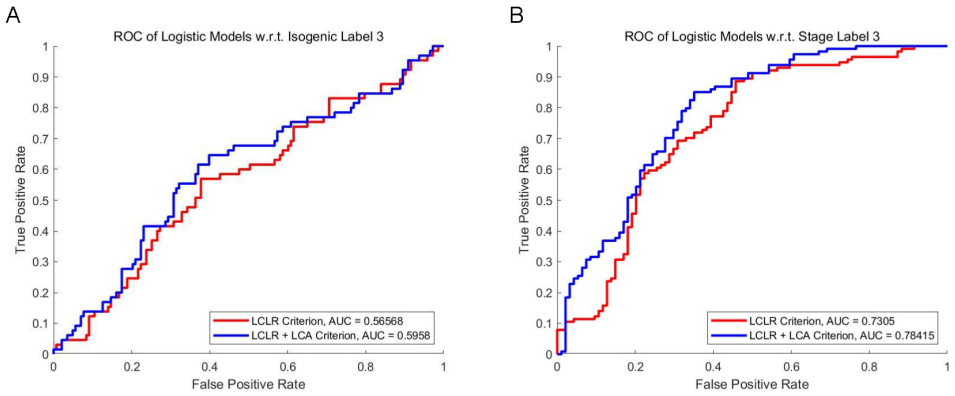


Fig 8. The effectiveness of BAR as a supplement of BSR
(A) prediction on whether a cell has the third Isogenic label. (B) prediction on whether a cell has the third Stage label. It turns out that after adding BAR information, AUC is higher.

only one of many, and other methods may be even more effective at predicting isogenic- or stage-label within this data-set.

Conclusion and discussion

We have introduced a very simple algorithm for finding BAR-biclusters. The essence of this algorithm involves splitting each column of the original data-set into two, allowing for individual quadrant information to be accessed using standard inner product calculations. This simple pre-processing step can be followed by any biclustering method that searches for BSRs. Above we demonstrated one simple BAR-biclustering method – the LCA-algorithm – built using the LCLR-algorithm of [18]. While we believe that the LCLR method is a good choice for biclustering general data-sets, we remark that an advantage may be gained by choosing a biclustering method which is appropriate for the domain of application and the specifics of the data-set at hand.

We use our LCA-algorithm to analyze an scRNA-seq data-set from [25]. As described above, the LCA-algorithm is capable of finding multiple isogenic- and stage-specific BAR-biclusters. Importantly, the cell- and gene-membership of these BAR-biclusters are quit different from the membership of the corresponding BSR-biclusters (found using the standard LCLR-algorithm). The BAR-biclusters can be used to construct BAR-networks, which illustrate new and potentially useful groups of gene-gene interactions. Furthermore, the BAR-signals revealed by the LCA-algorithm can be used to build better linear-classifiers for predicting the isogenic- and stage-labels; these improved classifiers might be useful for better categorizing future data-sets.

Looking back at the organoid dataset

Last but not least, we take a final look at the GSEA results for the BAR- and BSR-biclusters found within the snRNA-seq dataset of [25]. As commented on previously, most of these biclusters are enriched for some of the isogenic-variants. We

can label the biclusters enriched for the APP- and PS1-variants as 'AD-specific', and the biclusters enriched for the wild-type as 'non-specific' (or 'WT-specific'). With these labels in place, we can look for gene-pathways which are more strongly enriched within the AD-specific BAR-biclusters than the non-specific biclusters. Some results of this search are shown in table 3, with the remaining results shown in Appendix S2 Appendix. These results are suggestive, as many of these pathways may play a role in the differences between AD- and WT-organoids seen in [25]. For example, the genes and pathways related to ion-transport and ion-channel binding may be partially responsible for the differences in conductivity and current-density observed between the isogenic-variants. Similarly, the pathways related to neuronal cell generation might play a role in the differences in arborization observed between the AD- and WT-organoids. So far these are only suggestions that follow naturally from the BAR-specific analysis above; confirmation of these suggestions will obviously require several more careful experiments.

Table 3. Difference of gene pathways between WT-biclusters and AD-biclusters

Cell Type	Gene Pathway
GABAergic Neurons	Hydrogen Ion Transmembrane Transporter Activity
Astroglia Cells	Hydrogen Ion Transmembrane Transporter Activity
Ventral Progenitor Cells	Hydrogen Ion Transmembrane Transporter Activity
Radia Glial Cells	Ion Channel Binding
Astroglia Cells	Neuronal Cell Body
Ventral Progenitor Cells	Neuronal Cell Body
Astroglia Cells	Generation of Neurons
Astroglia Cells	Glutamate Secretion

A few pathways selected among all pathways to demonstrate. To see a complete list, refer to S2 Appendix.

Acknowledgments

This research was supported by the following NIH grants: 3U19AG023122-13S2, UH3AG064706 and U19 AG065169.

References

1. Brazma A, Vilo J. Gene expression data analysis. FEBS letters. 2000;480(1):17–24.

2. Allocco DJ, Kohane IS, Butte AJ. Quantifying the relationship between co-expression, co-regulation and gene function. BMC bioinformatics. 2004;5(1):1–10.

3. Lee HK, Hsu AK, Sajdak J, Qin J, Pavlidis P. Coexpression analysis of human genes across many microarray data sets. Genome research. 2004;14(6):1085–1094.

4. Tavazoie S, Hughes JD, Campbell MJ, Cho RJ, Church GM. Systematic determination of genetic network architecture. Nature genetics. 1999;22(3):281–285.

5. Sahoo D, Dill DL, Gentles AJ, Tibshirani R, Plevritis SK. Boolean implication networks derived from large scale, whole genome microarray datasets. Genome biology. 2008;9(10):1–17.

6. Vo D, Singh SC, Safa S, Sahoo D. Boolean implication analysis unveils candidate universal relationships in microbiome data. BMC bioinformatics. 2021;22(1):1–15.

7. Sinha S, Tsang EK, Zeng H, Meister M, Dill DL. Mining TCGA data using Boolean implications. PloS one. 2014;9(7):e102119.

8. Sinha S, Dill DL. Mining Large Heterogeneous Cancer Data Sets Using Boolean Implications. bioRxiv. 2016; p. 045021.

9. Dabydeen SA, Desai A, Sahoo D. Unbiased Boolean analysis of public gene expression data for cell cycle gene identification. Molecular biology of the cell. 2019;30(14):1770–1779.

10. Sahoo D, Dill DL, Tibshirani R, Plevritis SK. Extracting binary signals from microarray time-course data. Nucleic acids research. 2007;35(11):3705–3712.

11. Zhu C, Jiang R, Chen T. Constructing a Boolean implication network to study the interactions between environmental factors and OTUs. Quantitative Biology. 2014;2(4):127–141.

12. Wan YW, Xiao C, Guo NL. Network-based identification of smoking-associated gene signature for lung cancer. In: 2010 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). IEEE; 2010. p. 479–484.

13. Nation J, Okimoto G, Wenska T, Achari A, Maligro J, Yoshioka T, et al. A Comparative analysis of MRNA expression for sixteen different cancers. Draft available at http://math hawaii edu/ jb. 2017;.

14. Wang H, Lu HHS, Chueh TH. Constructing biological pathways by a two-step counting approach. Plos one. 2011;6(6):e20074.

15. Chueh TH, Lu HHS. Inference of biological pathway from gene expression profiles by time delay Boolean networks. PLos one. 2012;7(8):e42095.

16. Hopfensitz M, Mussel C, Wawra C, Maucher M, Kuhl M, Neumann H, et al. Multiscale binarization of gene expression data for reconstructing Boolean networks. IEEE/ACM transactions on computational biology and bioinformatics. 2011;9(2):487–498.

17. Madeira SC, Oliveira AL. Biclustering algorithms for biological data analysis: a survey. IEEE/ACM transactions on computational biology and bioinformatics. 2004;1(1):24–45.

18. Rangan AV. A simple filter for detecting low-rank submatrices. Journal of Computational Physics. 2012;231(7):2682–2690.

19. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment. 2008;2008(10):P10008.

20. McInnes L, Healy J, Melville J. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:180203426. 2018;.

21. Cheng Y, Church GM. Biclustering of expression data. In: Ismb. vol. 8; 2000. p. 93–103.

22. Gu J, Liu JS. Bayesian biclustering of gene expression data. BMC genomics. 2008;9(1):1–10.

23. Mitra S, Banka H. Multi-objective evolutionary biclustering of gene expression data. Pattern Recognition. 2006;39(12):2464–2477.

24. Bleuler S, Prelic A, Zitzler E. An EA framework for biclustering of gene expression data. In: Proceedings of the 2004 congress on evolutionary computation (IEEE Cat. No. 04TH8753). vol. 1. IEEE; 2004. p. 166–173.

25. Ghatak S, Dolatabadi N, Trudler D, Zhang X, Wu Y, Mohata M, et al. Mechanisms of hyperexcitability in Alzheimer's disease hiPSC-derived neurons and cerebral organoids vs isogenic controls. Elife. 2019;8:e50333.

26. Yoon S, Elahi LS, Pasca AM, Marton RM, Gordon A, Revah O, et al. Reliability of human cortical organoid generation Nature Methods. 2018;16:75-78.

27. Miura Y, Li MY, Birey F, Ikeda K, Revah O, Thete MV, et-al. Generation of human striatal organoids and cortico-striatal assembloids from human pluripotent stem cells. Nature Biotechnology. 2020;38(12):1421-1430.

28. Pasca SP. Assembling human brain organoids. Science. 2019;363(6423):126-127. doi: 10.1126/science.aau5729.

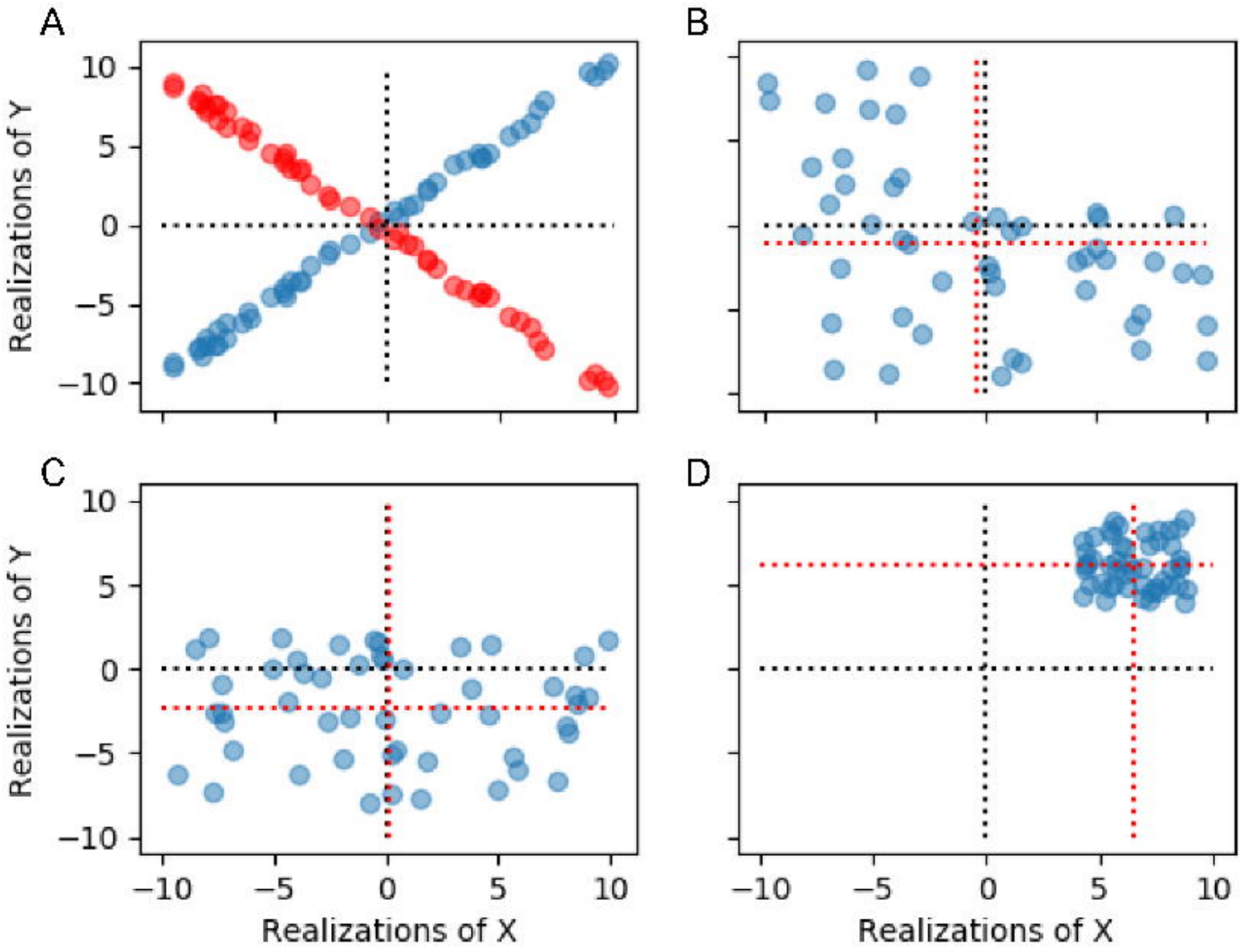
29. Rangan AV, McGrouther CC, Kelsoe J, Schork N, Stahl E, Zhu Q, et al. A loop-counting method for covariate-corrected low-rank biclustering of gene-expression and genome-wide association study data. PLoS computational biology. 2018;14(5):e1006105.

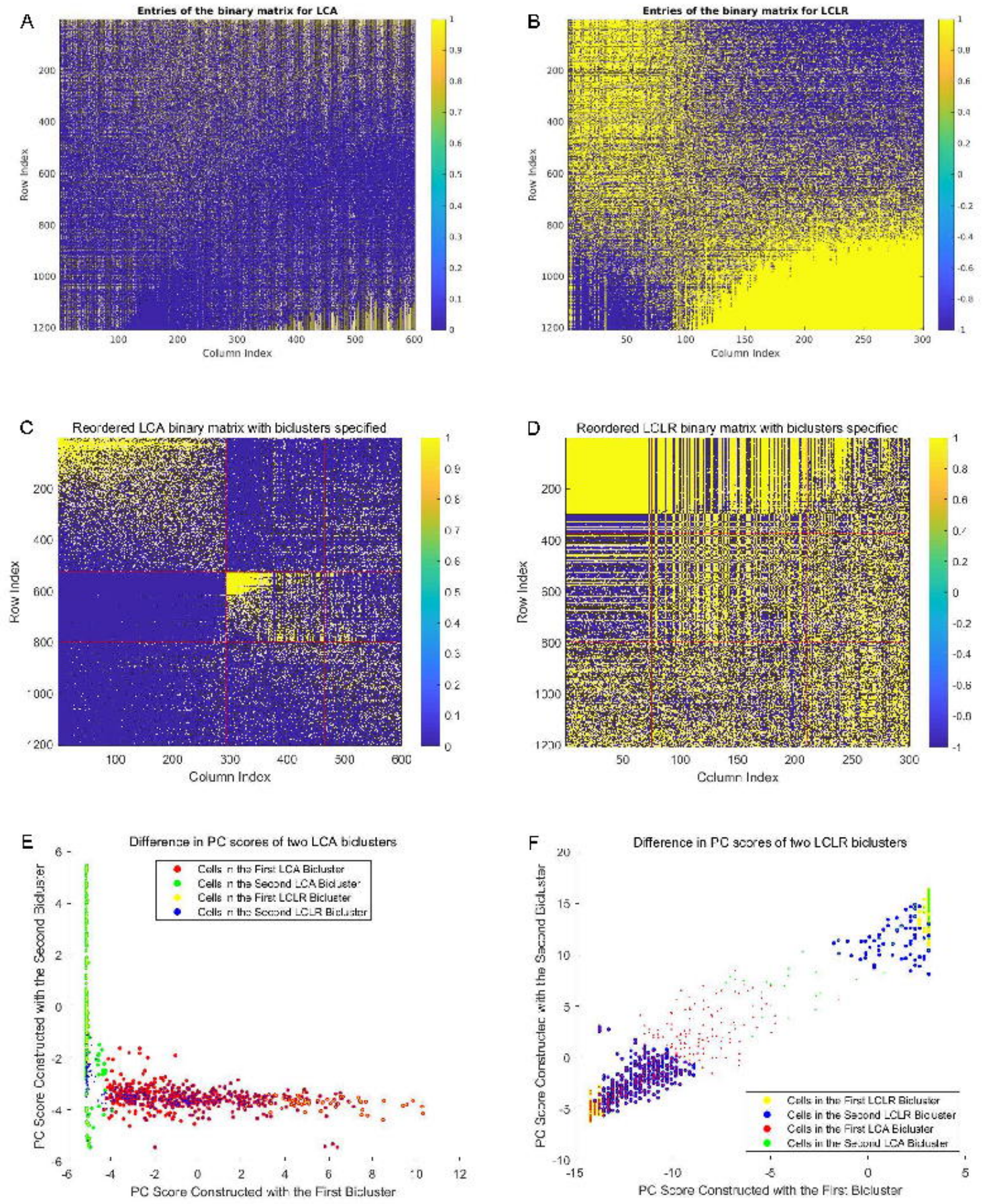
30. Zhu Q, Wong AK, Krishnan A, Aure MR, Tadych A, Zhang R, et al. Targeted exploration and analysis of large cross-platform human transcriptomic compendia. Nature methods. 2015;12(3):211–214.

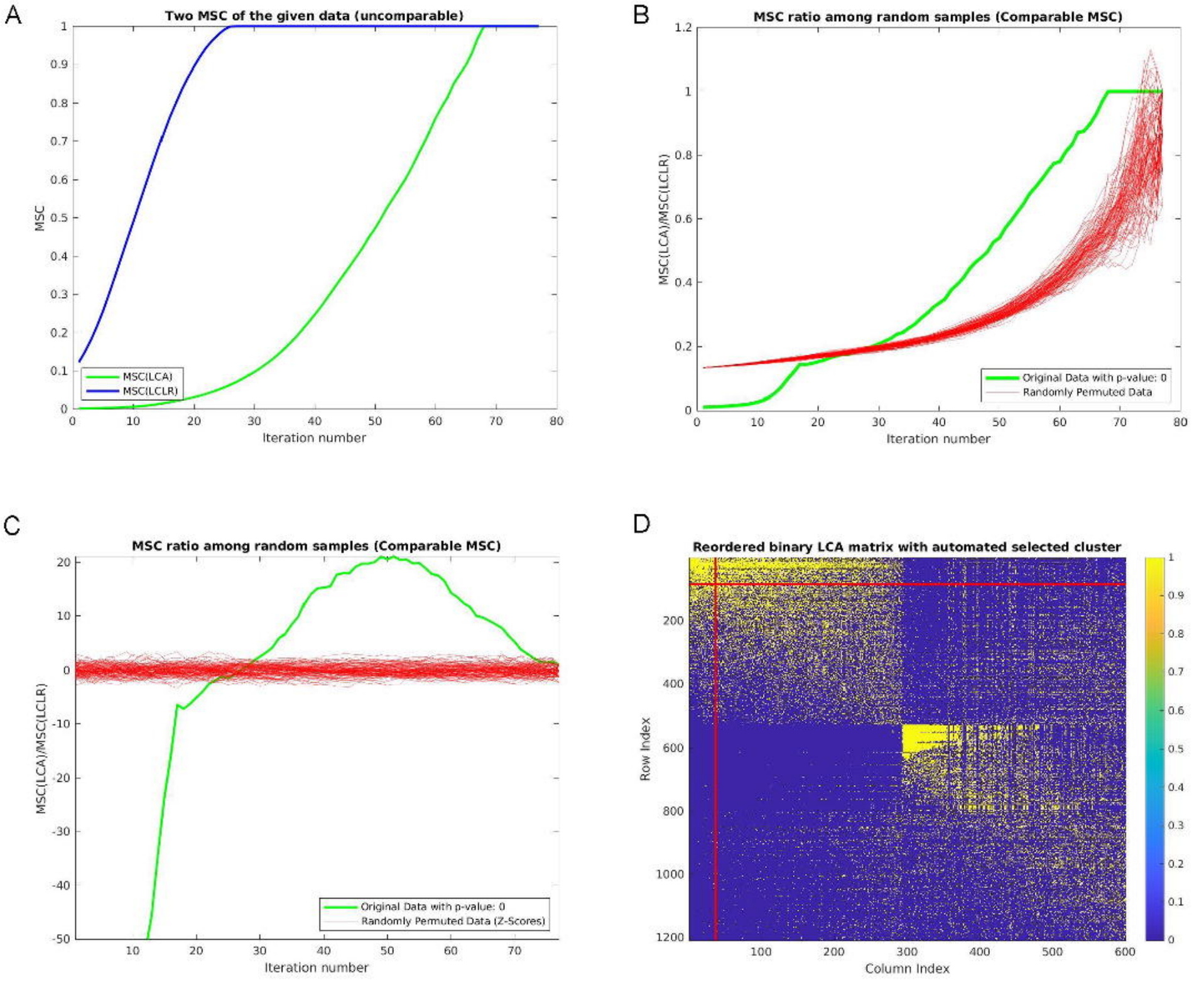
Supporting information

S1 Appendix More details on LCLR. Case studies on the comparison between LCLR and Umap and Louvain clustering.

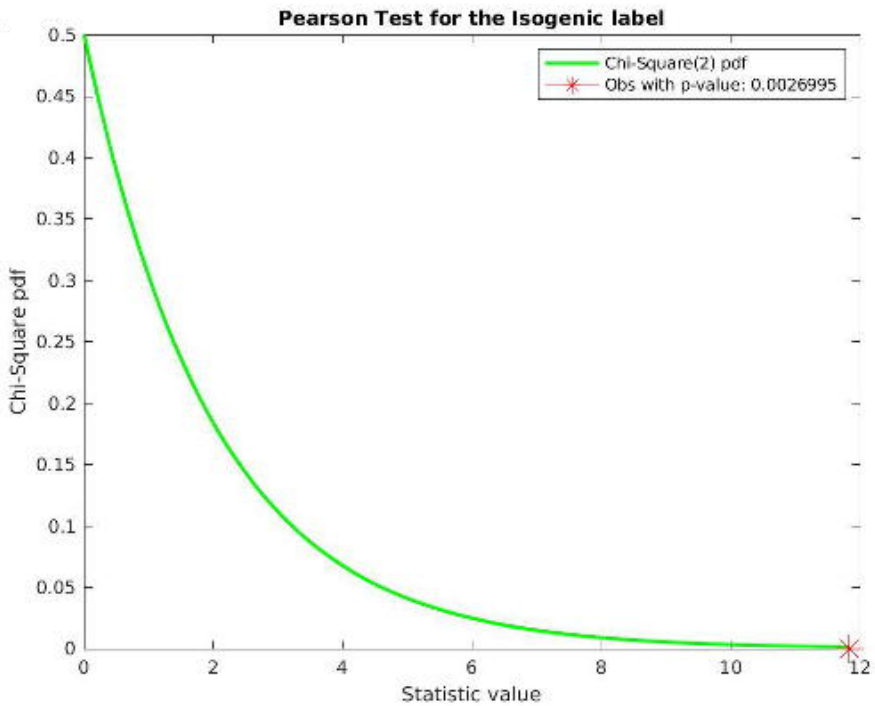
S2 Appendix GSEA results as the difference between WT-biclusters and AD-biclusters. A list of significant gene pathways and the corresponding p-values as the difference of GSEA results between WT-biclusters and AD-biclusters.


A
B
C
D
Realizations of Y
Realizations of X
10
5
0
−5
−10

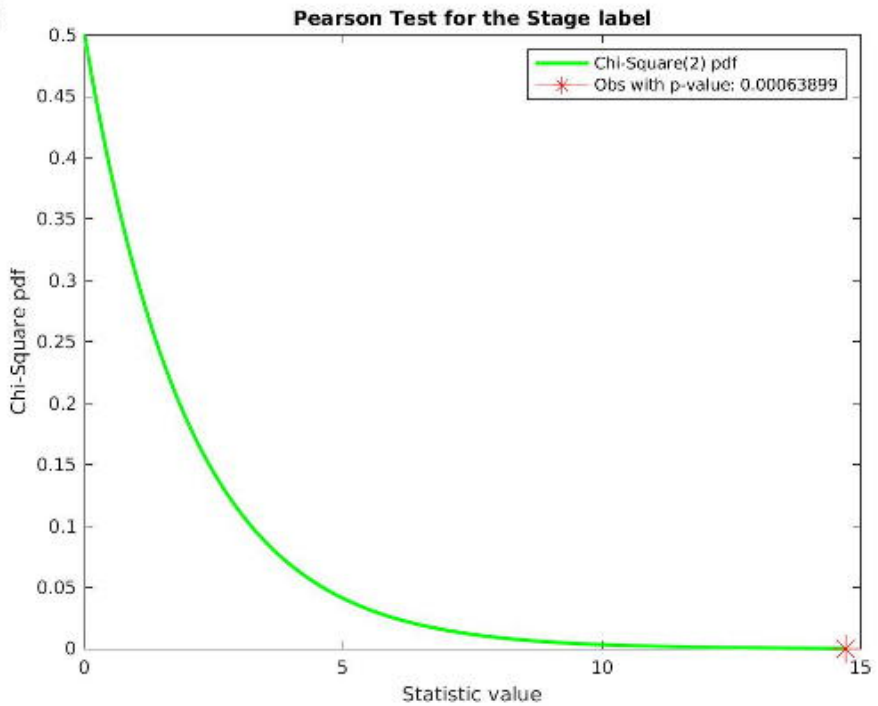

A
Entries of the binary matrix for LCA
Row Index
Column Index
B
Entries of the binary matrix for LCLR
Row Index
Column Index
C
Reordered LCA binary matrix with biclusters specified
Row Index
Column Index
D
Reordered LCLR binary matrix with biclusters specified
Row Index
Column Index
E
Difference in PC scores of two LCA biclusters
Cells in the First LCA Bicluster
Cells in the Second LCA Bicluster
Cells in the First LCLR Bicluster
Cells in the Second LCLR Bicluster
PC Score Constructed with the Second Bicluster
PC Score Constructed with the First Bicluster
F
Difference in PC scores of two LCLR biclusters
Cells in the First LCLR Bicluster
Cells in the Second LCLR Bicluster
Cells in the First LCA Bicluster
Cells in the Second LCA Bicluster
PC Score Constructed with the Second Bicluster
PC Score Constructed with the First Bicluster

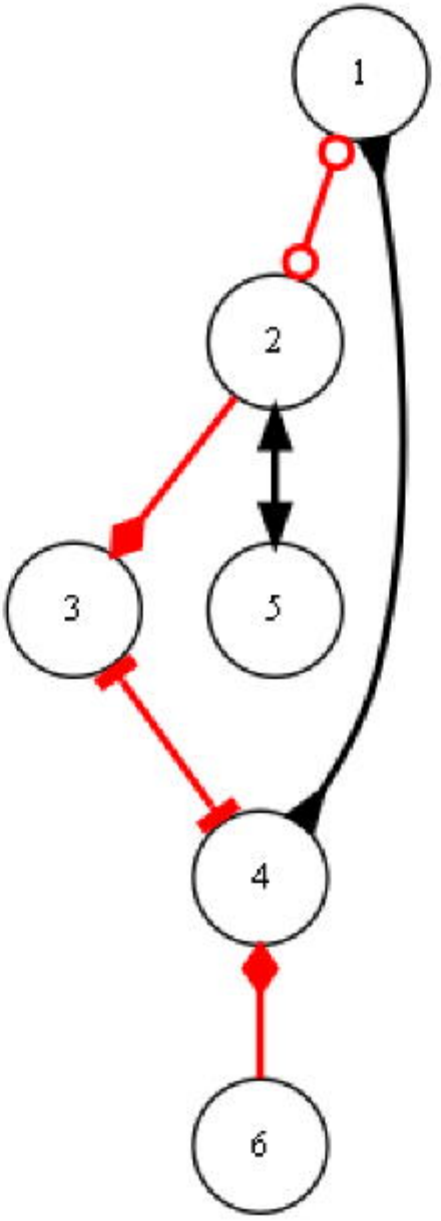

A
Two MSC of the given data (uncomparable)
MSC
Iteration number
MSC(LCA)
MSC(LCLR)
B
MSC ratio among random samples (Comparable MSC)
MSC(LCA)/MSC(LCLR)
Iteration number
Original Data with p-value: 0
Randomly Permuted Data
C
MSC ratio among random samples (Comparable MSC)
MSC(LCA)/MSC(LCLR)
Iteration number
Original Data with p-value: 0
Randomly Permuted Data (Z-Scores)
D
Reordered binary LCA matrix with automated selected cluster
Row Index
Column Index

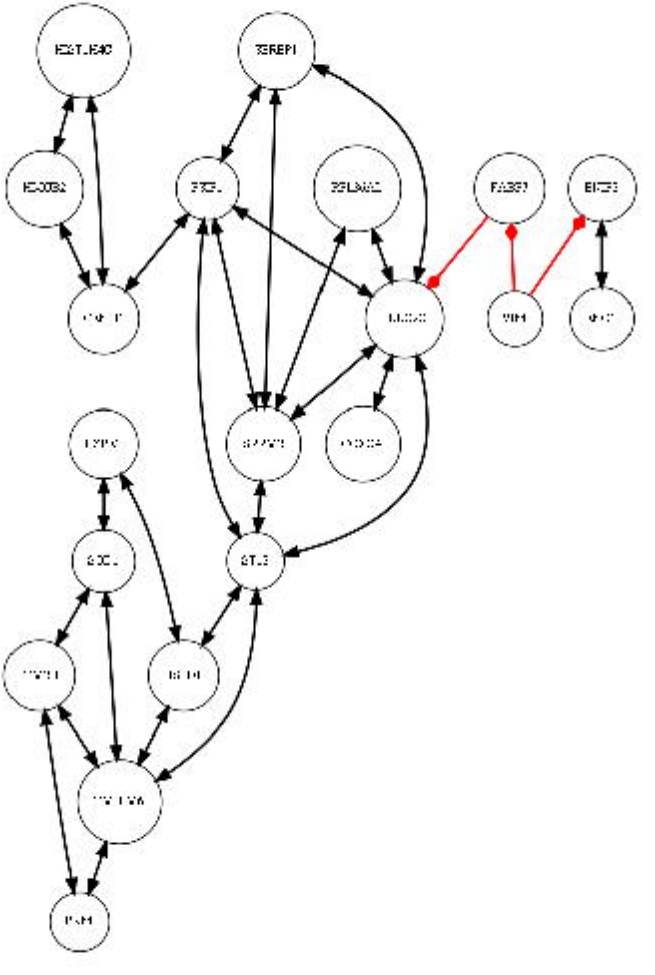
A

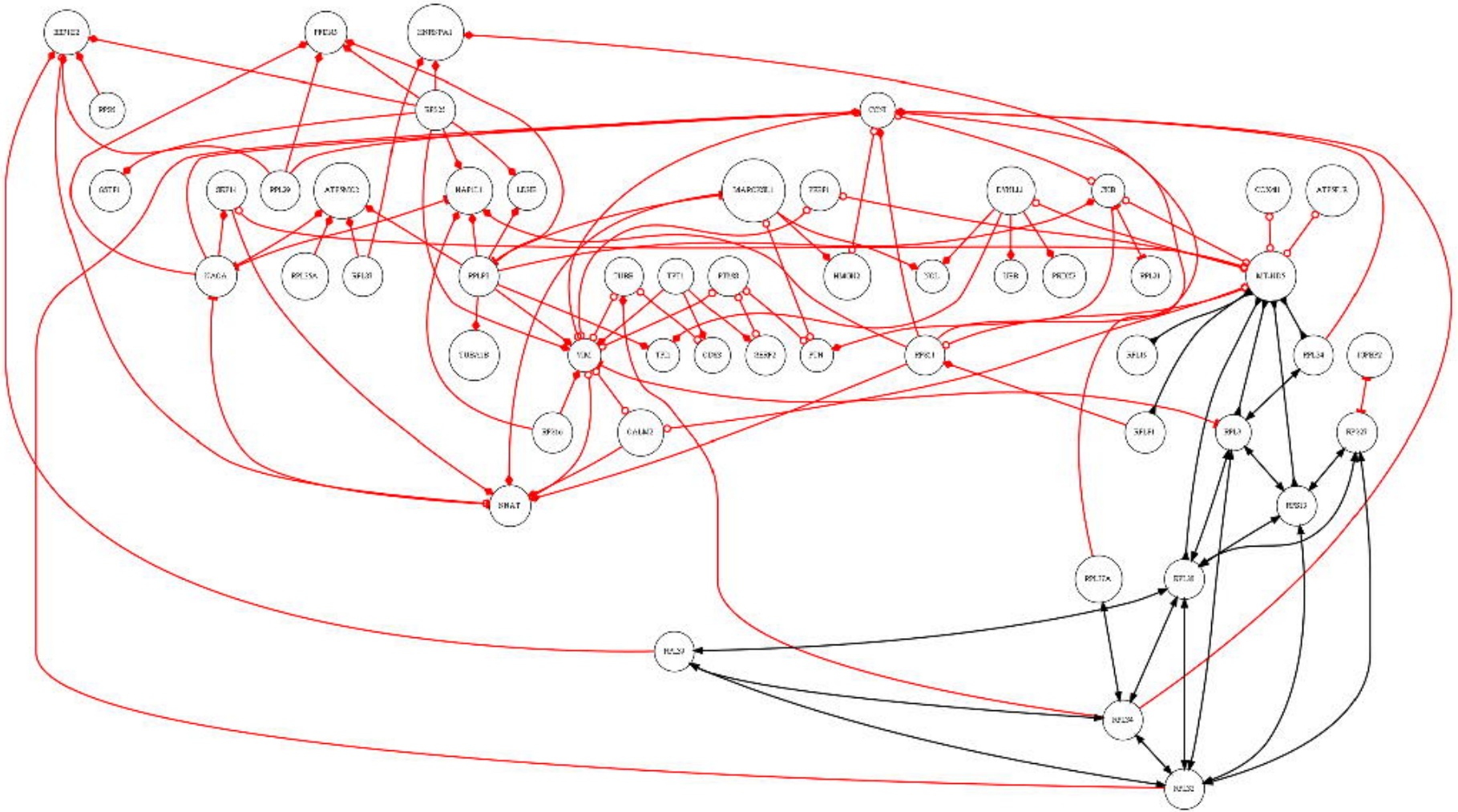

Pearson Test for the Isogenic label
Chi-Square(2) pdf
Obs with p-value: 0.0026995
Chi-Square pdf
Statistic value
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
0
2
4
6
8
10
12

B

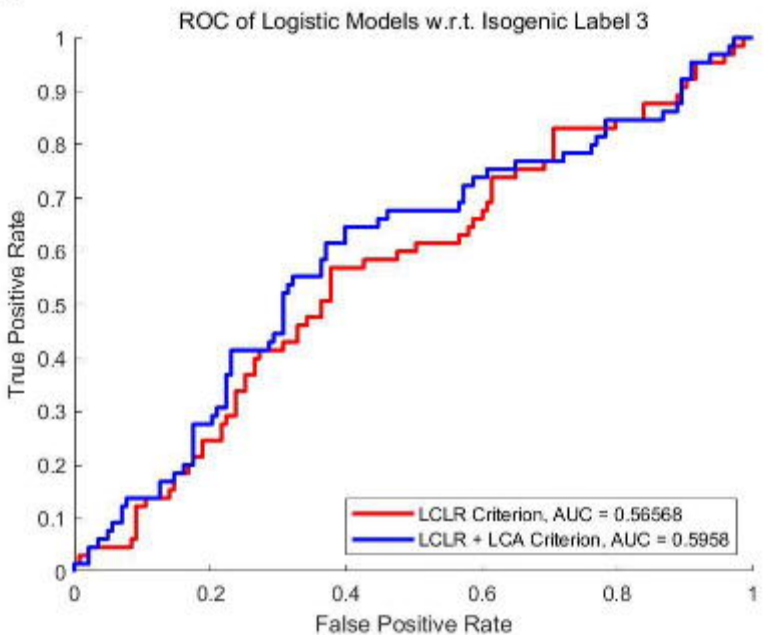

Pearson Test for the Stage label
Chi-Square(2) pdf
Obs with p-value: 0.00063899
Chi-Square pdf
Statistic value
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
0
5
10
15


1
2
3
5
4
6

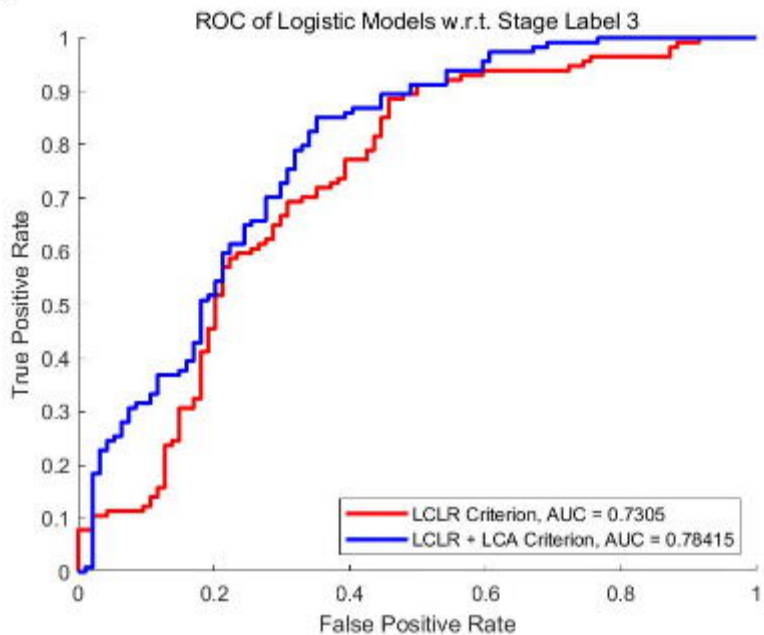




A


ROC of Logistic Models w.r.t. Isogenic Label 3
True Positive Rate
False Positive Rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.2
0.4
0.6
0.8
1
LCLR Criterion, AUC = 0.56568
LCLR + LCA Criterion, AUC = 0.5958

B


ROC of Logistic Models w.r.t. Stage Label 3
True Positive Rate
False Positive Rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.2
0.4
0.6
0.8
1
LCLR Criterion, AUC = 0.7305
LCLR + LCA Criterion, AUC = 0.78415