Supplementary Material

Identifying Multi-Layer Gene Regulatory Modules from Multi-Dimensional Genomic Data

Wenyuan Li *
Molecular and Computational Biology, University of Southern California
Los Angeles, CA 90089, USA

Shihua Zhang *
Molecular and Computational Biology, University of Southern California
Los Angeles, CA 90089, USA

Chun-Chi Liu
Institute of Genomics and Bioinformatics, National Chung Hsing University
Taiwan, China

Xianghong Jasmine Zhou †
Molecular and Computational Biology, University of Southern California
Los Angeles, CA 90089, USA
xjzhou@usc.edu

*Equally contributed joint first authors.
†To whom correspondence should be addressed.

Contents

S1 Theorem of the sMBPLS Algorithm

Before proving the theorem, please note that the sparsity penalty function we used in the problem formulation is the well-known L_1 norm which has been shown to be a convex function (Zou & Hastie, 2005). The details refer to Page 5 in `http://www.stanford.edu/class/ee364b/notes/subgradients_notes.pdf` So our problem formulation is convex.

Theorem S1.1. *The iterative sMBPLS algorithm can solve the sparse multi-block problem.*

Before proving this theorem, we first give the following lemma.

Lemma S1.2. *Let $\widehat{x}$ be the maximizer of $2\beta x - \alpha x^2 - P_\lambda(x)$, where $P_\lambda(x) = 2\lambda|x|$ and $\alpha > 0$. We have*

$$\widehat{x} = \frac{1}{\alpha}\mathit{sparse}(\beta) = \frac{1}{\alpha}\mathit{sign}(\beta)(|\beta| - \lambda)_+$$

where $\mathit{sparse}(\cdot)$ is the soft thresholding function.

The proof of the lemma is easy and thus omitted.

To prove that the algorithm sMBPLS can achieve the maximization of the criterion in Equation (2) of the manuscript, we can detail its regularization form as

$$\begin{aligned} L(\mathbf{w}_i, \mathbf{q}, \mathbf{b}) =& \Omega(\mathbf{t}, \mathbf{u}, \mathbf{w}_i, \mathbf{q}, \mathbf{b}) - \sum_{i=1}^{3} \alpha_i \|\mathbf{w}_i\|^2 - \beta\|\mathbf{q}\|^2 \\ =& 2\mathtt{cov}(\mathbf{t}, \mathbf{u}) - \sum_{i=1}^{3} P_{\lambda_i}(\mathbf{w}_i) - P_{\lambda_4}(\mathbf{q}) - \sum_{i=1}^{3} \alpha_i \|\mathbf{w}_i\|^2 - \beta\|\mathbf{q}\|^2 \\ =& 2\sum_{i=1}^{3} b_i \mathbf{w}_i^T X_i^T Y \mathbf{q} - \sum_{i=1}^{3} P_{\lambda_i}(\mathbf{w}_i) - P_{\lambda_4}(\mathbf{q}) - \sum_{i=1}^{3} \alpha_i \|\mathbf{w}_i\|^2 - \beta\|\mathbf{q}\|^2 \end{aligned} \tag{1}$$

where $\alpha_i > 0 (i = 1, 2, 3)$ and $\beta > 0$.

Lemma S1.3. *For a fixed $\mathbf{w}_i$ $(i = 1, 2, 3)$ and $\mathbf{q}$, the b_i $(i = 1, 2, 3)$ that maximizes Equation (1) and satisfies $\|\mathbf{b}\| = 1$ is,*

$$\mathbf{b} = \mathit{norm}\left(\begin{array}{c} \mathbf{w}_1^T X_1^T Y \mathbf{q} \\ \mathbf{w}_2^T X_2^T Y \mathbf{q} \\ \mathbf{w}_3^T X_3^T Y \mathbf{q} \end{array} \right) \tag{2}$$

where $\mathit{norm}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$.

Proof. Its proof is the same as that of Lemma S1.3. □

Lemma S1.4. *For a fixed $\mathbf{w}_i$ $(i = 1, 2, 3)$ and $\mathbf{b}$, the $\mathbf{q}$ that maximizes $L(\mathbf{w}_i, \mathbf{q}, \mathbf{b})$ in Equation (1) and satisfies $\|\mathbf{q}\|^2 = 1$ is,*

$$\mathbf{q} = \mathit{norm}\left(\mathit{sparse}_{\lambda_4}\left(\sum_{i=1}^{3} Y^T X_i \mathbf{w}_i b_i \right) \right) \tag{3}$$

Proof. Given the fixed $\mathbf{w}_i$ $(i = 1, 2, 3)$ and $\mathbf{b}$, maximizing $L(\mathbf{w}_i, \mathbf{q})$ is equivalent to the maximization of

$$2\sum_{i=1}^{3} b_i \mathbf{w}_i^T X_i^T Y \mathbf{q} - P_{\lambda_4}(\mathbf{q}) - \beta\|\mathbf{q}\|^2 = \sum_j \left\{ 2\left(\sum_{i=1}^{3} Y^T X_i \mathbf{w}_i b_i\right)_j q_j - p_{\lambda_4}(q_j) - \beta q_j^2 \right\} \tag{4}$$

where $(\mathbf{x})_j$ is the j-th element of the vector $\mathbf{x}$. The expression of the optimal q_j can be obtained by repeatedly applying Lemma S1.2 to $2\left(\sum_{i=1}^{3} Y^T X_i \mathbf{w}_i b_i\right)_j q_j - p_{\lambda_4}(q_j) - \beta q_j^2$. According to Lemma S1.2, the maximizer of (4) is obtained by applying the sparse function $\mathtt{sparse}_{\lambda_4}(\cdot)$ to the vector $\sum_{i=1}^{3} Y^T X_i \mathbf{w}_i b_i$ componentwise. Therefore, we obtain the optimal $\mathbf{q}$ as follows,

$$\mathbf{q} = \frac{1}{\beta}\mathtt{sparse}_{\lambda_4}\left(\sum_{i=1}^{3} Y^T X_i \mathbf{w}_i b_i\right) \tag{5}$$

Because $\mathbf{q}$ must satisfy $\|\mathbf{q}\|^2 = 1$, so we prove it. □

Lemma S1.5. *For a fixed* $\mathbf{q}$ *and* $\mathbf{b}$*, the* $\mathbf{w}_i$ *(*$i = 1, 2, 3$*) that maximizes* $L(\mathbf{w}_i, \mathbf{q})$ *in Equation (1) and satisfies* $\|\mathbf{w}_i\| = 1$ *is,*

$$\mathbf{w}_i = \mathit{norm}(\mathit{sparse}_{\lambda_i}(X_i^T Y \mathbf{q})) \tag{6}$$

Proof. Given the fixed $\mathbf{q}$ and $\mathbf{b}$, maximizing $L(\mathbf{w}_i, \mathbf{q}, \mathbf{b})$ is equivalent to the maximization of

$$2\sum_{i=1}^{3} b_i \mathbf{w}_i^T X_i^T Y \mathbf{q} - P_{\lambda_i}(\mathbf{w}_i) - \alpha_i\|\mathbf{w}_i\|^2 = \sum_j \left\{ 2\sum_{i=1}^{3} \left(X_i^T Y \mathbf{q}\right)_j (\mathbf{w}_i)_j - p_{\lambda_i}((\mathbf{w}_i)_j) - \alpha_i (\mathbf{w}_i)_j^2 \right\} \tag{7}$$

where $(\mathbf{w}_i)_j$ is the j-th element of the vector $\mathbf{w}_i$. The expression of the optimal $(\mathbf{w}_i)_j$ can be obtained by repeatedly applying Lemma S1.2 to $2\left(X_i^T Y \mathbf{q}\right)_j (\mathbf{w}_i)_j - p_{\lambda_4}((\mathbf{w}_i)_j) - \alpha_i (\mathbf{w}_i)_j^2$. According to Lemma S1.2, the maximizer of (7) is obtained by applying the sparse function $\mathtt{sparse}_{\lambda_i}$ to the vector $X_i^T Y \mathbf{q}$ componentwise. Therefore, we obtain the optimal $\mathbf{w}_i$ as follows,

$$\mathbf{w}_i = \frac{1}{\alpha_i}\mathtt{sparse}_{\lambda_i}\left(X_i^T Y \mathbf{q}\right) \tag{8}$$

Because $\mathbf{w}_i$ must satisfy $\|\mathbf{w}_i\|^2 = 1$, so we prove it. □

In the next, we will show the $\mathbf{b}, \mathbf{q}, \mathbf{w}_i$ obtained from the sMBPLS algorithm can achieve the equalities Equation (2) of Lemma S1.3, Equation (3) of Lemma S1.4, Equation (6) of Lemma S1.5, respectively. We summarize the steps of the sMBPLS algorithm as follows,

$$\mathbf{b} \propto T^T \mathbf{u} = \begin{bmatrix} \mathbf{t}_1^T \mathbf{u} \\ \mathbf{t}_2^T \mathbf{u} \\ \mathbf{t}_3^T \mathbf{u} \end{bmatrix} \propto \begin{bmatrix} \mathbf{t}_1^T Y \mathbf{q} \\ \mathbf{t}_2^T Y \mathbf{q} \\ \mathbf{t}_3^T Y \mathbf{q} \end{bmatrix} \propto \begin{bmatrix} \mathbf{w}_1^T X_1^T Y \mathbf{q} \\ \mathbf{w}_2^T X_2^T Y \mathbf{q} \\ \mathbf{w}_3^T X_3^T Y \mathbf{q} \end{bmatrix} \tag{9}$$

$$\mathbf{q} \propto \mathtt{sparse}_{\lambda_4}\left(Y^T \mathbf{t}\right) = \mathtt{sparse}_{\lambda_4}\left(Y^T \sum_{i=1}^{3} b_i \mathbf{t}_i\right) = \mathtt{sparse}_{\lambda_4}\left(\sum_{i=1}^{3} Y^T X_i \mathbf{w}_i b_i\right) \tag{10}$$

$$\mathbf{w}_i \propto \mathtt{sparse}_{\lambda_i}\left(X_i^T \mathbf{u}\right) = \mathtt{sparse}_{\lambda_i}\left(X_i^T Y \mathbf{q}\right) \tag{11}$$

So we can get the conclusion that the algorithm sMBPLS can maximize the sparse multi-block PLS problem formulated in Equation (2) of the manuscript.

S2 Details of Simulation Study

We conducted the simulation study for two purposes: (1) Test the module discovery ability of the sMBPLS algorithm; (2) Compare the performance of the sMBPLS and MBPLS algorithms on module discovery. In order to discover the multi-dimensional modules by using MBPLS, an intuitive two-step procedure can be performed: firstly applying MBPLS to the data, then selecting the top-ranking input and response variables to form a module by ordering the absolute values of the loadings and weight vectors. A more intuitive comparison way is to sort the loadings and latent variable from either MBPLS or sMBPLS, then to visualize X_1, X_2, X_3, Y whose rows and columns are reordered by the sortings of their loadings and weight vectors.

The data were generated by extending the simulation scenarios in recent sparse PLS literatures (Lê Cao et al., 2008; Chun & Keles, 2010) from two-block data (X and Y) to multi-block data (X_1, X_2, X_3, Y). We set $K = 70$ samples, $N_1 = 100, N_2 = 200, N_3 = 300$ input variables for X_1, X_2, X_3 and $M = 400$ response variables for Y, all with base error model being Gaussian with different variances. We divided the samples into three consecutive groups of sizes $5, 5, 60$. These sample groups are denoted by the sets (A, B, C). The multi-dimensional module will be embedded in the first two sample groups A and B. The simulation scenario of input variables introduces four types of data generation schemes:

1. $\mathbf{Type}_1(\mu_a, \mu_b, \mu_c)$: This is a random model that can generate embedded module across sample groups. A K-dimensional column vector $\mathbf{x}$ of the input data block is generated by the model $\mathbf{x} = H + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is an independent identically distributed random vector from the Gaussian $\mathcal{N}(0, \mathbf{I}_K)$, and $H \in \mathbb{R}^{K\times 1}$ is a hidden component representing the following mean structure,

$$H_i = \begin{cases} \mu_a, & \text{if } i \in \text{sample group A} \\ \mu_b, & \text{if } i \in \text{sample group B} \\ \mu_c, & \text{if } i \in \text{sample group C} \end{cases} \tag{12}$$

2. $\mathbf{Type}_2(\mu, \sigma, \delta)$: This is a random model with more complicated mean structure than **Type I**. A column vector $\mathbf{x}$ of the input data block is generated by the model $\mathbf{x} = H + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}_K)$, and H is defined as follows,

$$H = \mu + \sigma \mathbf{I}_{u_i \leqslant \delta} \qquad 1 \leqslant i \leqslant K \tag{13}$$

   u_i is an uniformly distributed random number from $\mathcal{U}(0, 1)$. $\mathbf{I}_{u_i \leqslant \delta}$ is a vector function whose i^{th} component is defined as $\begin{cases} 1, & u_i \leqslant \delta \\ 0, & u_i > \delta \end{cases}$.

3. $\mathbf{Type}_3(\rho, \mu)$: This is a random model to generate the multicolinearity data which occur often in biology. A matrix with p variables and K samples is generated by a multivariate normal distribution with auto-regressive covariance, i.e., $\mathcal{N}(\mu, \Sigma_{p\times p})$, where $\Sigma_{p\times p}$ is from an AR(1) process [1] with an auto-correlation ρ. We usually set $\rho = 0.9$ to activate the multicolinearity among p variables. In simulation data, we generated the variables with multicolinearity in X_2 of the simulation data (Figure S1).

[1] AR(1) is the abbreviation of the first-order auto-regressive process.

4. $\mathbf{Type}_4(\mu, \sigma)$: This is a typical normal random model, $\mathbf{x} \sim \mathcal{N}(\mu, \sigma \mathbf{I}_K)$.

Next, we will use the combinations of these four types of random model to generate three data blocks X_1, X_2, X_3.

For X_1, the first 5 columns are generated by $\mathbf{Type}_1(\mu_a = 2.4, \mu_b = -1.6, \mu_c = 0)$, the following 5 columns by $\mathbf{Type}_1(\mu_a = -1.6, \mu_b = 2.4, \mu_c = 0)$, then the following 30 columns by $\mathbf{Type}_2(\mu = 1, \sigma = -1, \delta = 0.7)$, finally the rest of columns by $\mathbf{Type}_3(\rho = 0.9, \mu = 0)$.

We generated the first 5 columns of X_2 by $\mathbf{Type}_1(\mu_a = 2, \mu_b = -1, \mu_c = 0)$, the following 5 columns by $\mathbf{Type}_1(\mu_a = -1, \mu_b = 2, \mu_c = 0)$, then the following 50 columns by $\mathbf{Type}_3(\rho = 0.9, \mu = 0)$, finally the rest of columns by $\mathbf{Type}_4(\mu = 2, \sigma = 2)$.

For X_3, the first 5 columns are generated by $\mathbf{Type}_1(\mu_a = 2, \mu_b = -1, \mu_c = 0)$, the following 5 columns by $\mathbf{Type}_1(\mu_a = -1, \mu_b = 2, \mu_c = 0)$, then the following 200 columns by $\mathbf{Type}_4(\mu = 0, \sigma = 1)$, finally the rest of columns by $\mathbf{Type}_3(\rho = 0.9, \mu = 0)$.

The response data block Y is the general model of $Y = \sum_{i=1}^{3} b_i X_i \boldsymbol{\beta}_i + \Xi$, where $\Xi \sim \mathcal{N}(0, \mathbf{I}_{K \times M})$. We set $b_1 = b_2 = b_3 = 1$ to make each input data block equally contribute to Y. The first 5 columns of $\boldsymbol{\beta}_1 \in \mathbb{R}^{N_i \times M}$ are generated by $\mathbf{Type}_1(1, -0.5, 0.5, 0.5)$ that is based on the four groups of input variables of X_1 aforementioned. Then the following 5 columns are generated by $\mathbf{Type}_1(-0.5, 1, 0.5, 0.5)$. Finally the rest columns by $\mathbf{Type}_4(\mu = 0, \sigma = 1)$. The $\boldsymbol{\beta}_2$ and $\boldsymbol{\beta}_3$ are generated in the same random scheme of $\boldsymbol{\beta}_1$.

In this simulation setting, a multi-dimensional module of small size is embedded in the data blocks whose sizes are $10 \sim 40$ times to the module size, and the module's strength level (entry values of the module) is comparable to the rest of data blocks. Figure S1 plots an example of this simulation data by color-scaled images. The data were simulated 50 times, and each time we embedded only one module.

In order to discover the multi-dimensional modules by using MBPLS, an intuitive two-step procedure can be performed: firstly applying MBPLS to the data, then selecting the top-ranking input and response variables to form a module by ordering the absolute values of the loadings and weight vectors. A more intuitive comparison way is to sort the loadings and latent variable from either MBPLS or sMBPLS, then to visualize X_1, X_2, X_3, Y whose rows and columns are reordered by the sortings of their loadings and weight vectors. The multi-dimensional module would appear in the left-top and right-bottom corners (whose whose corresponding variables and sampels have large absolute values of weights) in the reordered blocks. For example, the reordered blocks by MBPLS as shown in Figure S1(B)-panel2 (in Supplementary material) are observed to have no clear modular submatrices in corners; while a multi-dimensional module can be observed in reordered blocks by sMBPLS in Figure S1(B)-panel3 and zoomed out in Figure S1(C).

We systematically compared sMBPLS and MBPLS on these 50 simulation data. We found that MBPLS always failed for all 50 simulation data in that it assigns totally different variables and samples to the discovered module. In contrast, 100% of modules identified by sMBPLS have significant overlaps with predefined modules over at least three dimensions at the significance level 0.01. An example is detailed in Figure S1. Since the MBPLS method maximizes the covariance between all input and response variables across all samples as shown in Figure S1(B), it overlooks embedded modules when the module's signal is overwhelmed by background noise. On the contrary, the sparsity penalty forces sMBPLS to focus on "local" (i.e., across a small subsets of variables and samples) peaks in the covariance, which correspond to (multi-dimensional) modules of relatively small size. Our results show that the sMBPLS method is more accurate at identifying modules in noisy data, and thus more suitable for biological applications.

We also compared the sMBPLS with the sparse version of PLS which is applied to the combined single input block (i.e. X_1, X_2, X_3 merged to a single block X). The same 50 times of simulation data was used. The result showed that all modules identified by sparse PLS have at least 1 dimension missing and 56% modules have 2 dimensions missing. The lack of such power for the single-block approach may attribute to the unbalanced covariance structures across multiple blocks, i.e., the covariance signals of some blocks may be overwhelmed by those of other blocks. The result is even worse when we compared with a popular biclustering algorithm "SAMBA" (Tanay et al., 2002) which is applied to the single block by merging X_1, X_2, X_3, Y. All modules identified by SAMBA have at least one dimensions missing, and 84%/34% modules have at least two/three dimensions missing.

S3 Details of Overlap Significance Test

Given two modules each of which contains a sample set and four genomic dimensions (i.e., CNV, DM, ME and GE), we applied the following procedure to test the overlap significance of these two modules.

1. **Measure sample overlap:** Apply the right-tailed Hypergeometric test to sample sets of two modules, and output a p-value to measure their overlap significance.
2. **Measure overlap of a dimension:** Apply the right-tailed Hypergeometric test to feature sets of two modules for each genomic dimension, then output four p-values, each of which measures the overlap significance of a dimension between two modules.
3. **Determine overlap significance of two modules:** If four of these five overlap tests (i.e., overlap tests in the above two steps) are significant (e.g., p-value< 0.05), we consider these two modules are significantly overlapped.

S4 Figure of Module Size Distribution

See Figure S2.

S5 Methylation Data Processing

The DNA methylation marks were obtained from the TCGA project (McLendon et al., 2008; The_Cancer_Genome_Atlas_Research_Network, 2011). Specifically, the Illumina Infinium HumanMethylation27 BeadChip platform was used to obtain the methylation profiles which can be downloaded in the TCGA website (The_Cancer_Genome_Atlas_Research_Network, 2011). The platform has been carefully verified to be a sensitive, reproducible method for genome-wide screening of methylation events and can generate data on a large number of informative loci or marks (27,578 CpG measurements) for each sample. Detailed preprocessing procedures can be found on the TCGA website (we used the Level 3 data, `http://cancergenome.nih.gov/`).

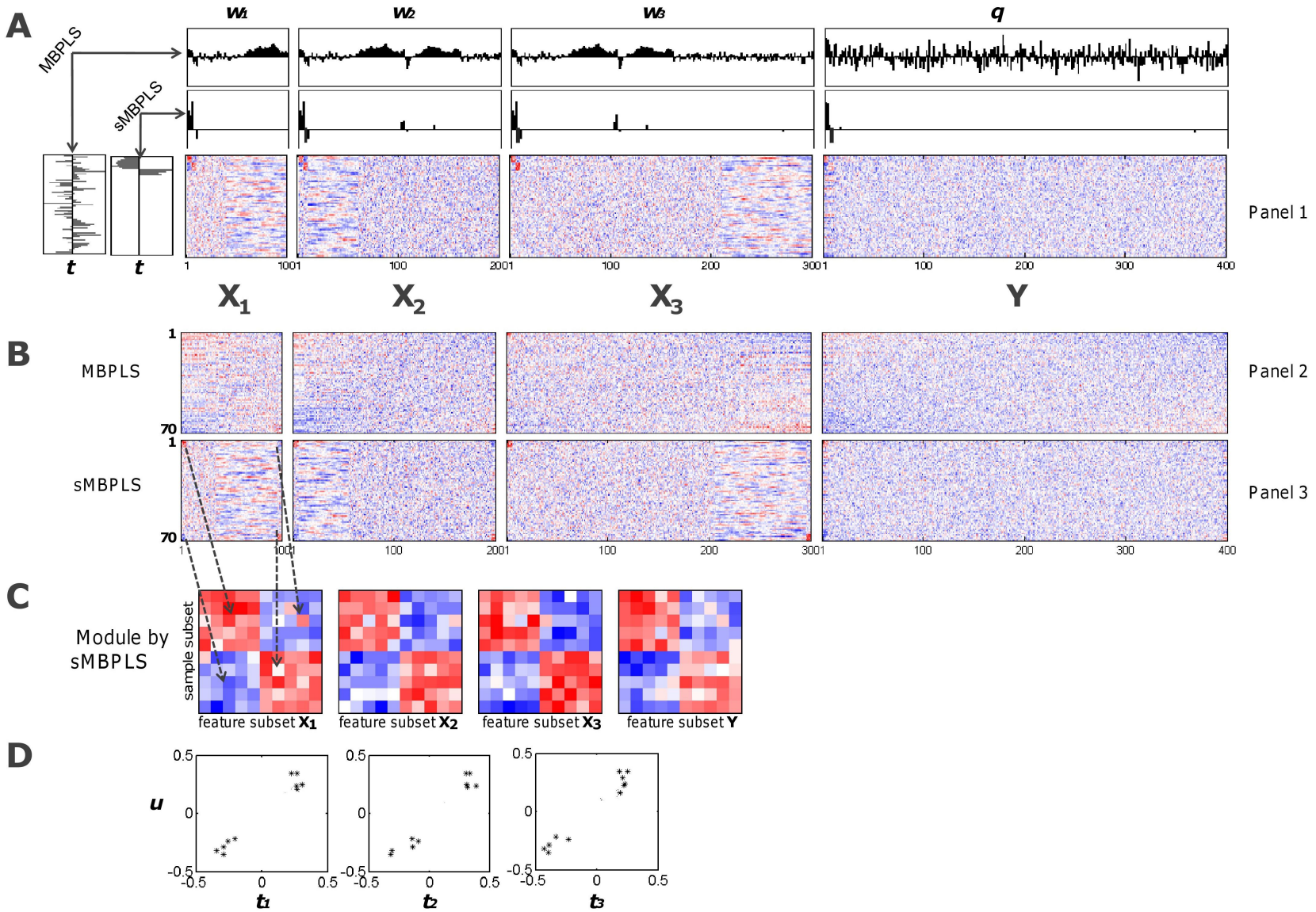

Figure S1. Illustration of the simulation study. (A) Panel 1 of the heat maps shows the original simulated data. The module is placed at the left-top corner of each data block. The horizontal and vertical bar plots show the weights of corresponding variables (loading vectors $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \mathbf{q}$) and latent variable $\mathbf{t}$ by appying MBPLS and sMBPLS to the permutated X_1, X_2, X_3, Y respectively. Please note that the variables with multicolinearity in X_2 are assigned very large loading weights by MBPLS, while sMBPLS successfully ignores multicolinearity and identifies the module's signal (refer to loading weights of sMBPLS). (B) Panel 2 and 3 of heat maps are the reordered matrices based on the descending order of elements in loading vectors and latent variables of MBPLS and sMBPLS, respectively. Therefore, the module should be observed in the left-top and right-bottom corners (whose corresponding variables and sampels have large absolute values of weights) in the reordered matrices. (C) The heat map of the multi-dimensional module identified by sMBPLS from the randomly permutated simulation data in (A). (D) The latent variable $\mathbf{u}$ and $\mathbf{t}_i$ value of the sMBPLS method show significant correlation structures.

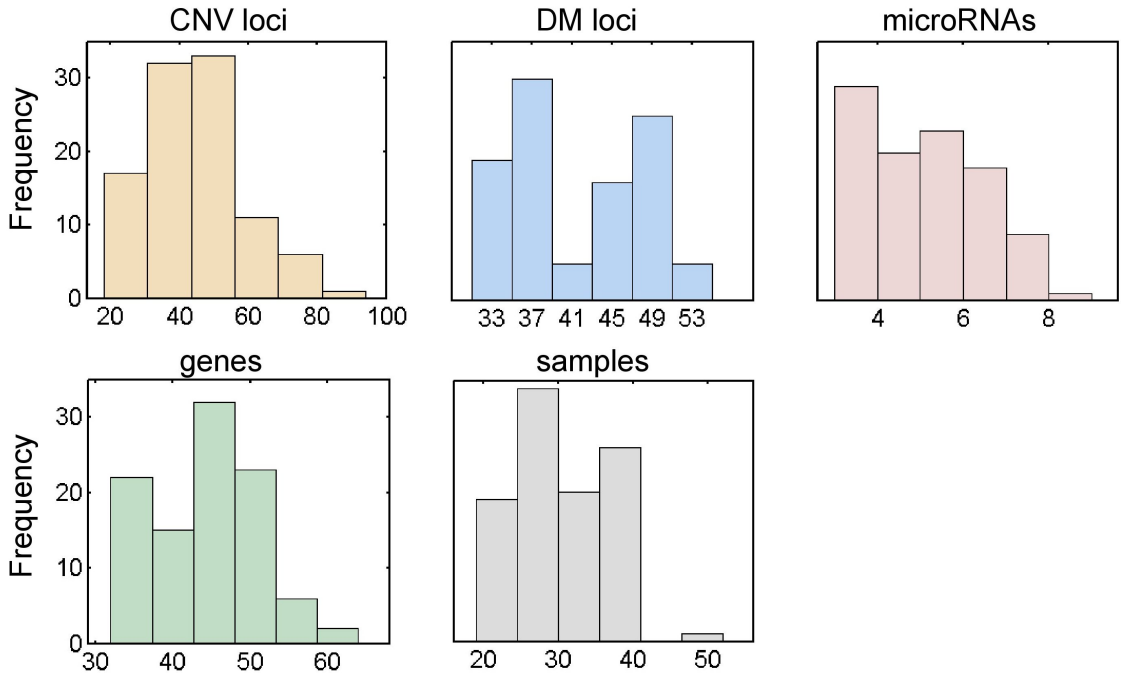

Figure S2. Module size distribution of CNV loci, DM marks, MicroRNAs, genes and samples in multi-dimensional regulatory modules. The average size of CNV loci, DM , MicroRNAs, genes and samples are 45.19, 41.97, 5.50, 44.74, 30.21 respectively, and the minimum size of them are 18, 31, 3, 32, 19, and the maximum size are 94, 55, 9, 64, 52 in all the detected modules.

S6 Analysis of Gene and microRNA Dimensions

When we used the experimentally validated target database of miRNAs in "miRecords (version 2)" (Xiao et al., 2009), there are 2 modules containing genes that are known targets of the miRNAs in the same modules. This is due to the scarce experimentally validated targets. When "miRecords (version 3)" was used, 8 modules are found to have genes to be known targets of its miRNAs. So this indicates that with the accumulation of known targets of miRNAs, more modules will be validated to contain genes to be known target of miRNAs in the same modules. When we used the database "miRBase" on predicted targets of miRNAs (Griffiths-Jones et al., 2008), all modules have genes that are predicted targets of miRNAs in the same modules.

S7 Comparison based on GO Enrichment Analysis

The top 100 biclusters with the highest scores identified by the "SAMBA" biclustering method are dominated by two dimensions: gene and CNV. On average, each bicluster contains 295.9 CNV loci, 2.5 methylation marks, 7.9 miRNAs, and 658.9 genes. That is, each bicluster contains extremely large numbers of CNV loci and genes compared with other dimensions. In addition, 59% (22%) biclusters missed at least one (two) dimensions, therefore, these biclusters do not fit our purpose of simultaneously exploring 4 dimensions – the main motivation of our study. For the 100 modules identified by the sparse PLS method, however, on average each module contains 84.1 CNV loci, 19.1 methylation marks, 10.3 miRNAs, and 47.0 genes. These modules have much more balanced number of variables from the four dimensions. For GO enrichment comparison, we used all modules identified by sparse PLS (each dimension of which contains at least 2 genomic features), and we considered all genes in the GE dimension, CNV-harbored genes, Methylation adjacent genes, and miRNAs. Given the sMBPLS method (or sparse PLS) and an enriched GO category (q-value<0.05), we used the minus *log* of the smallest q-value of the enriched module to represent the x-axis (or y-axis) coordinate of this GO category for the method sMBPLS (or sparse PLS) (see Figure S3). The results show that the sMBPLS can identify more functionally homogeneous modules with more diverse GO terms than the sparse PLS method.

S8 Overlap Analysis of Modules

Since our method finds the next module on the deflated matrix by "subtracting out" the submatrices corresponding to the module already identified. Figure S4 visually illustrates the effect before and after the matrix deflation process on the simulation data. This example demonstrates that "subtract out" formula is effective to remove signal of modules and prevent the method from getting stuck in the same local minima. However, such procedure may lead to a certain degree of overlaps between the modules identified. We used the overlap significance test (details in the Section S3) on the identified 100 modules to investigate how distinct these modules are. Our results showed that only 1 pair of modules have significant overlap at the level of p-value after Bonferroni correction <0.05.

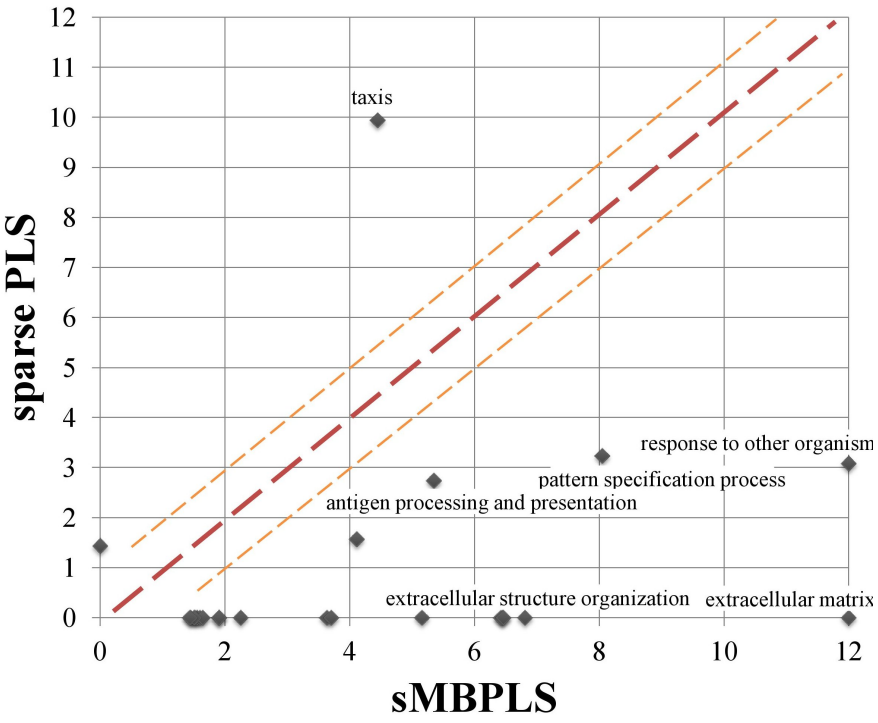

Figure S3. Comparison of enriched fourth level GO categories between our sMBPLS method and the sparse PLS method. All categories that were enriched on one of the two algorithms were selected. y-axis: minus log q-value for GO enrichment using sMBPLS algorithm. x-axis: minus log q-value for the sparse PLS method. Points below the central diagonal line represent categories that were more enriched using the sMBPLS algorithm, and points above the line represent categories more enriched using sparse PLS. Points below (above) the light dashed lines represent differences greater than one order of magnitude between the two methods. Points on the x-axis (y-axis) represent categories which were only enriched using sMBPLS (sparse PLS).

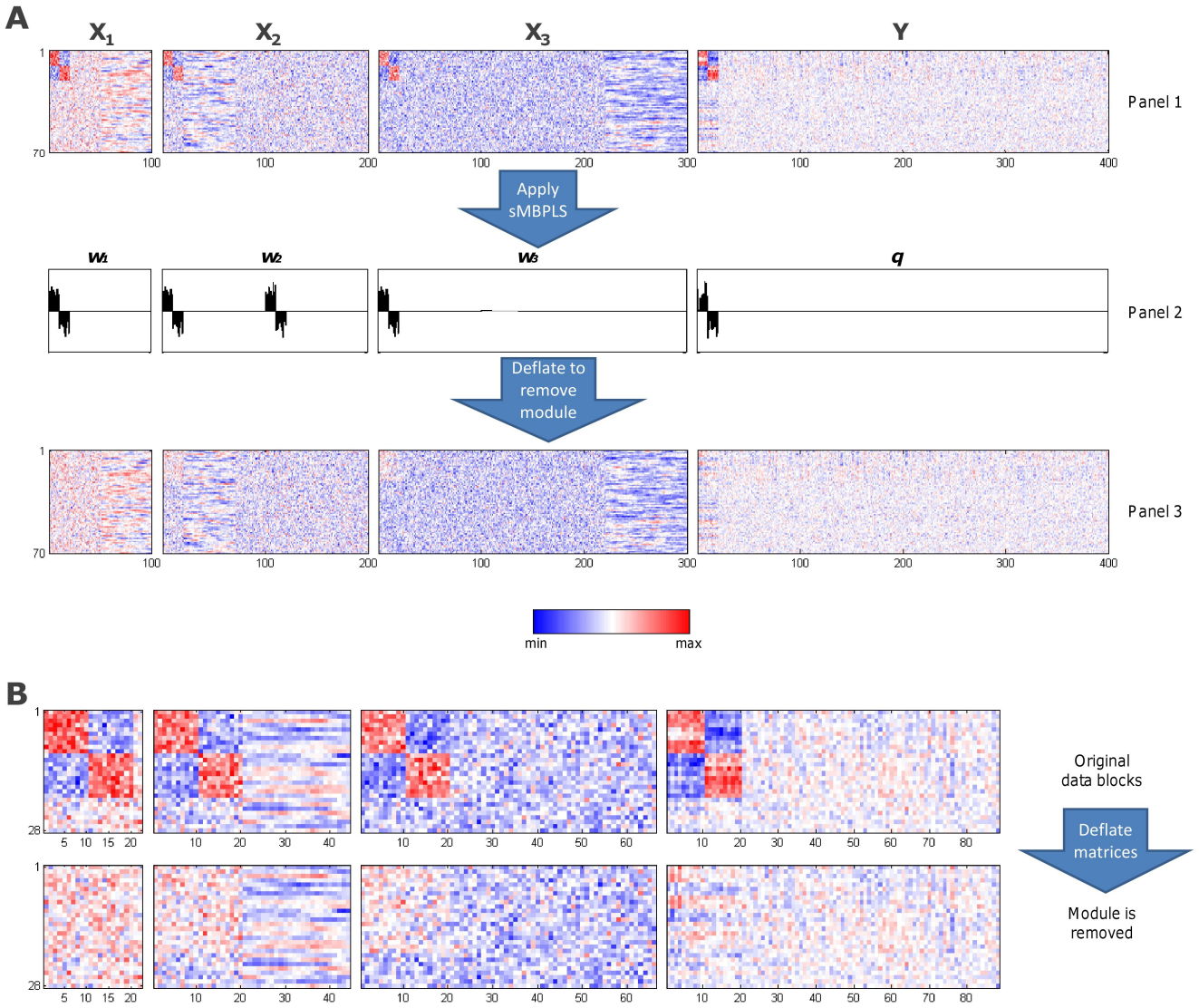

Figure S4. Illustration of the module removal by using matrix deflation formula. (A) Panel 1 shows the heat map of the original simulated data. The embedded module is placed at the left-top corner of each data block. Panel 2 plots the weights of corresponding variables (loading vectors $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3, \mathbf{q}$) by applying sMBPLS to data blocks. Panel 3 shows the heat map of the simulated data after removing the signal of the sMBPLS module by using the matrix deflation formula presented in Section 2.4 of the manuscript. It can be found that the module's signal is completely removed from the data blocks. Therefore, the sMBPLS algorithm can repeat its discovery procedure to identify next module on the deflated data blocks. (B) Zoom in heat maps of original data and deflated data.

S9 Signal extraction procedure of the Encyclopedia of DNA Elements (ENCODE) data

Recently, in the ENCODE production phase (September 2007 $\sim$ present) (Thomas et al., 2007), there are 191 ENCODE genome-wide tables for ChIP-seq. To perform enrichment analysis, we integrated these ChIP-seq samples in Genome-wide ENCODE data including chromatin modification, TF binding, Methylation, and chromatin accessibility. UCSC database provides the peak tables for these ChIP-Seq data.

However, different data types have different criterion for the significance. We used both top N and threshold condition to select peaks as follows:

1. If the table is Methylation-Seq data, select the peaks with score > 0.6;
2. Else if the peaks have p-value, select the peaks that are in top 20K and their p-values are $<$ 1E-4;
3. Otherwise, select the peaks that are in top 20K and their signal values > 1.

Then for each table, we selected the target genes whose transcriptional start site is nearby the peaks within 1000 base pairs.

References

Chun H, Keles S (2010) Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72:3–25.

Griffiths-Jones S, Saini H, van Dongen S, Enright A (2008) miRBase: tools for microRNA genomics. *Nucleic Acids Res* 37:D154–D158.

Lê Cao KA, Rossouw D, Robert-Granié C, Besse P (2008) A sparse PLS for variable selection when integrating omics data. *Statistical Applications in Genetics and Molecular Biology* 7.

McLendon R, Friedman A, Bigner D, Van Meir E, Brat D, Mastrogianakis G, Olson J, Mikkelsen T, Lehman N, Aldape K et al. (2008) Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature* 455:1061–1068.

The_Cancer_Genome_Atlas_Research_Network (2011) Integrated genomic analyses of ovarian carcinoma. *Nature* 474:609–615.

Tanay A, Sharan R, Shamir R (2002) Discovering statistically significant biclusters in gene expression data. *Bioinformatics* 18 Suppl 1:S136–144.

Thomas DJ, Rosenbloom KR, Clawson H, Hinrichs AS, Trumbower H, Raney BJ, Karolchik D, Barber GP, Harte RA, Hillman-Jackson J, Kuhn RM, Rhead BL, Smith KE, Thakkapallayil A, Zweig AS, Haussler D, Kent WJ (2007) The ENCODE project at UC santa cruz. *Nucleic Acids Res* 35:D663–D667.

Xiao F, Zuo Z, Cai G, Kang S, Gao X, Li T (2009) miRecords: an integrated resource for microRNA-target interactions. *Nucleic Acids Res* 37:D105–D110.

Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67:301–320.