Low-Rank Regularization for Learning Gene Expression Programs

Guibo Ye[1,2], Mengfan Tang[1], Jian-Feng Cai[3], Qing Nie[2,4], Xiaohui Xie[1,4]*

1 Department of Computer Science, University of California Irvine, Irvine, California, United States of America, 2 Department of Mathematics, University of California Irvine, Irvine, California, United States of America, 3 Department of Mathematics, University of Iowa, Iowa City, Iowa, United States of America, 4 Center for Complex Biological Systems, University of California Irvine, Irvine, California, United States of America

Abstract

Learning gene expression programs directly from a set of observations is challenging due to the complexity of gene regulation, high noise of experimental measurements, and insufficient number of experimental measurements. Imposing additional constraints with strong and biologically motivated regularizations is critical in developing reliable and effective algorithms for inferring gene expression programs. Here we propose a new form of regulation that constrains the number of independent connectivity patterns between regulators and targets, motivated by the modular design of gene regulatory programs and the belief that the total number of independent regulatory modules should be small. We formulate a multi-target linear regression framework to incorporate this type of regulation, in which the number of independent connectivity patterns is expressed as the rank of the connectivity matrix between regulators and targets. We then generalize the linear framework to nonlinear cases, and prove that the generalized low-rank regularization model is still convex. Efficient algorithms are derived to solve both the linear and nonlinear low-rank regularized problems. Finally, we test the algorithms on three gene expression datasets, and show that the low-rank regularization improves the accuracy of gene expression prediction in these three datasets.

Citation: Ye G, Tang M, Cai J-F, Nie Q, Xie X (2013) Low-Rank Regularization for Learning Gene Expression Programs. PLoS ONE 8(12): e82146. doi:10.1371/journal.pone.0082146

Editor: Mark R. Muldoon, Manchester University, United Kingdom

Received February 1, 2013; **Accepted** October 30, 2013; **Published** December 17, 2013

Funding: The work was partly supported by a grant from National Institute of Health and a grant from National Science Foundation. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: xhx@ics.uci.edu

Introduction

Systematically discovering gene expression programs within cells is a fundamental goal in both basic and applied biomedical researches, and is crucial for elucidating factors determining cell types, controlling cellular states, or switching cells from healthy states to diseased ones. Although the total number of genes within an organism is usually large (e.g., $\sim$20,000 in humans), most of these genes are believed to be regulated by a much smaller subset of genes called regulators (e.g., transcription factors, signalling molecules, growth factors, etc.) A challenge in computational biology is how to use machine learning methods to automatically discover the mapping from regulators to target genes, thereby inferring the underlying regulatory programs, from a given set of observations [1].

More specifically, suppose we are given a set of observations $\mathbf{z}=\{\mathbf{x}_i,\mathbf{y}_i\}_{i=1}^m$, where $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^p$ denotes the expression of p regulators and $\mathbf{y}_i \in \mathcal{Y} \subset \mathbb{R}^L$ denotes the expression of L target genes in sample i. The goal of learning gene expression programs is to infer the mapping $\vec{f}=(f_1,\ldots,f_L):\mathcal{X}\rightarrow\mathcal{Y}$ that fits the observation $\mathbf{z}$, and to provide biological interpretations of the inferred mapping.

In addition to the purpose of uncovering gene regulatory mechanisms, the gene expression program learning problem arises recently also in a practical and applied setting in biotechnology development. High-throughput gene expression profiling using Affymetrix arrays typically costs $\sim$\$300 and \$800 for human and mouse, respectively, which is still too expensive to be used in large-scale perturbation, drug or small molecule screening, which typically requires tens of thousands of or even millions of expression profiles [2]. This constraint has motivated the development and adoption of the Luminex bead technology, which is able to measure $\sim$1000 genes at a much lower cost ($\sim$\$5 per profile). Because the gene expression is so highly correlated, scientists are proposing to use Luminex bead to measure 1000 carefully chosen "landmark" genes and to computationally extrapolate all remaining ones. This strategy will be able to significantly cut the cost of expression profiling; however, it also calls for better and more efficient methods for target gene expression prediction.

Although the gene expression program learning problem formulated above fits into standard supervised learning, solving the problem is difficult for a number of reasons. First, the total number of parameters determining the mapping from regulators to target genes is typically much greater than the total number of observations. Secondly, the gene expression measurements based on high-throughput techniques are known to be highly noisy. These factors make the gene expression program inference highly challenging. A number of methods have been proposed for gene expression program learning, including methods based on probabilistic graphical models [3–5], information-theoretic approaches [6,7], and ordinary differential equations (ODEs) [8,9].

See [1,10] for reviews of these and other approaches. However, the performance of these methods tend to be modest.

In this work, we formulate the gene expression program learning as a multi-target (more specifically L-target) regression problem, and use Tikhonov regularization to constrain the space of the L-target mapping. Two main forms of regularization with biological motivations have been proposed in the literature: a) *sparsity* - each target is likely regulated by only a few regulators instead of all, and b) *modularity* - the expression program is organized into modules, each consisting of a certain combination of regulators, and the total number of independent modules should be small. The sparsity regularization is well recognized and widely used in gene expression program inference [9,11,12]. Some of these previous work are based on graphical models [4,13], while others are based on regression. Within the regression framework, a common strategy of imposing sparsity is to use ℓ_1 norm regularization on regression coefficients, similar to the framework of Lasso (least absolute shrinkage and selection operator) [11,14].

The modularity regularization is much more difficult to handle and is the focus of this work. A popular approach is the probabilistic graphic model proposed by Segal et al. [15], which assigns target genes into different modules and constrains the genes within each module to be identically distributed, and models the regulatory program associated with each module using a rule-based decision tree. However, the Segal model is difficult to train, requiring long running time and only being able to find locally optimal solutions. In addition, the Segal model only captures the qualitative relationship between the regulators and targets since its main purpose is not on predicting the expression of target genes. Another approach taking the modularity structure into account is the SIMoNe proposed by Chiquet et al. [16]. SIMoNe models gene expression data using Gaussian graphical models, and imposes sparsity constraints on the inverse covariance matrix and introduces hidden nodes to the Gaussian graph to learn the network modularity. The primary goal of SIMoNe is to infer the gene regulatory networks in an unsupervised way without distinguishing regulator and target genes, which is different from our main objective.

Here we propose a new approach to incorporating the modularity constraint. We use the rank of the connectivity matrix between regulators and targets to represent the number of independent regulatory modules between them. The modularity regularization is then formulated as a low-rank constraint within a multi-target linear regression framework. The resulting model is convex, and we describe an efficient algorithm to find its globally optimal solution. We further show that the low-rank regularized regression problem can also be generalized to nonlinear cases, where we regularize the dimension of the hypothesis space of the L-target regression function. We prove that the resulting nonlinear low-rank model is still convex and derive an efficient algorithm to solve it. Finally, we benchmark the performance of the low-rank regulation models on two real biological datasets, and show that the low-rank regulation technique consistently improve prediction accuracy in both cases when compared to the Lasso model.

Methods

Learning gene expression programs in linear space

We begin by introducing some notations. Let $\mathbb{R}$ be the set of real numbers and $\mathbb{R}_+$ the subset of non-negative ones. Denote $\langle\cdot,\cdot\rangle_{\mathbb{R}^L}$ to be the inner product of $\mathbb{R}^L$. We assume that we have L target genes and define $\mathbb{N}_L=\{1,2,\ldots,L\}$. We further assume that for the ℓ-th target gene ($\ell\in\mathbb{N}_L$), m samples $\{\mathbf{x}_i,y_i^\ell\}_{i=1}^m$ are available, where $\mathbf{x}_i\in\mathcal{X}\subset\mathbb{R}^p$ denotes the expression of p regulators and $y_i^\ell\in\mathcal{Y}_\ell\subset\mathbb{R}$ denotes the expression of the ℓ-th target gene in sample i. Let $\mathbf{y}_i=(y_i^1,\ldots,y_i^L)^T$ and $\mathcal{Y}=\mathcal{Y}_1\times\ldots\times\mathcal{Y}_L$. The goal of learning gene expression programs is to infer the mapping $\vec{f}=(f_1,\ldots,f_L):\mathcal{X}\to\mathcal{Y}$ that fits the observation $\mathbf{z}=(\mathbf{x}_i,\mathbf{y}_i)_{i=1}^m$, and to provide biological interpretations of the inferred mapping.

In this section, we assume that each target gene is linearly regulated by the regulators. That is, for each $\ell\in\mathbb{N}_L$, $f_\ell=\omega_\ell^T\mathbf{x}$, where ω_ℓ is a fixed vector of coefficients.

Next we describe two types of regularization that can be used for gene expression program learning: one is the *sparsity* regularization, which has already been widely used in the literature, and the second is the *low-rank* regularization, which has not been used in the gene expression program learning, although having recently become popular in other problem domains such as matrix completion, covariance matrix estimation, metric learning, etc [17–20].

Sparsity regularization

Given the observation $\{\mathbf{x}_i,y_i^\ell\}_{i=1}^m$, a natural way to infer ω_ℓ is to solve a least-square minimization problem:

$$\hat{\omega}_\ell=\arg\min_{\omega_\ell\in\mathbb{R}^p}\frac{1}{2m}\sum_{i=1}^m\|\omega_\ell^T\mathbf{x}_i-\mathbf{y}_i^\ell\|^2, \qquad (1)$$

where the norm is the ℓ_2 norm by default. However, for the gene expression program learning problem, the ω_ℓ inferred by least-square minimization tends to be poor for a number of reasons: 1) the observations as measured by microarrays are usually very noisy, and 2) p is usually much larger than m, which can lead to overfitting. Various regularization techniques have been introduced to prevent overfitting including ridge regression [21] and Lasso [14]. Since each target gene is likely regulated by only a few regulators instead of all, a commonly used regularization technique in gene expression program learning is to impose an ℓ_1-norm based sparsity regularization on ω_ℓ as in Lasso [9,11,12,14]:

$$\hat{\omega}_\ell=\arg\min_{\omega_\ell\in\mathbb{R}^p}\frac{1}{2m}\sum_{i=1}^m\|\omega_\ell^T\mathbf{x}_i-\mathbf{y}_i^\ell\|^2+\lambda_\ell\|\omega_\ell\|_1, \qquad (2)$$

where $\lambda_\ell>0$ is a regularization parameter. We will call (2) the *Lasso model* in the following.

Low-rank regularization

In the Lasso model, we treat each regulation function f_ℓ separately and learn them independently. However, it is well-known that the expression values of target genes are often highly correlated, and biologists believe that this high correlation is caused by sharing of regulatory programs among different genes. In addition, although there exist p regulators in an organism, the number regulatory programs (called modules by biologists) active in a particular experimental setting is often much lower than p. These considerations suggest that instead of learning each ω_ℓ separately for each gene, we should learn all ω_ℓ's jointly, and impose a new regularization on the dimension of the span of the ω_ℓ's.

Let $W=(\omega_1,\ldots,\omega_L)$ be a $p\times L$ matrix with each column corresponding to one ω_ℓ. Constraining the dimension of the span of the ω_ℓ's is equivalent to regularizing the rank of W, which motivates us to propose the following model to learn gene expression programs

$$\hat{W} = \arg \min_{W \in \mathbb{R}^{p \times L}} \frac{1}{2m} \| Y - XW \|_F^2 + \lambda \| W \|_*, \tag{3}$$

where $X = (\mathbf{x}_1, \ldots, \mathbf{x}_m)^T$, $Y = (\mathbf{y}_1, \ldots \mathbf{y}_m)^T$ and $\|\cdot\|_F$ denotes the Frobenius norm for matrix. $|W\|_*$ is the nuclear norm of matrix W, defined to be the sum of the singular values of W. The nuclear norm is a convex function and is often used as a convex relaxation of the rank of W [22,23]. Since nuclear norm is convex, model (3) is a convex optimization problem. We will call (3) the *linear low-rank model* in the following. The linear low-rank model has not been proposed for gene expression analysis, although it has appeared in other problem domains such as matrix completion, covariance matrix estimation, metric learning, etc [17–20].

Low-rank regularization for learning gene expression programs in nonlinear space

Next we show that the low rank regularization can also be extended to learn nonlinear gene expression program. We start by proposing a low-rank regularization model in the Hilbert space, then prove that the model is a well-defined convex problem, and finally provide an algorithm to solve the model in the reproducing kernel Hilbert space (RKHS).

Low-rank model in Hilbert Space. We assume that each target gene is nonlinearly regulated by its regulators and the mapping $f_\ell \in \mathcal{H}_0$, which is a Hilbert space. Furthermore, we assume that the mappings of different target genes are related to each other in such a way that f_ℓ lies in a common low-dimensional subspace of $\mathcal{H}_0$. Note that the assumption of $\{f_\ell\}_{\ell \in \mathbb{N}_\ell}$ sharing a common subspace in Hilbert space is a natural generalization of the low-rank constraint in the linear case, where the weighting vectors $\{\omega_\ell\}_{\ell \in \mathbb{N}_\ell}$ share a low-dimensional subspace in Euclidean space $\mathbb{R}^d$.

Under the above assumption, the space $\mathrm{span}\{f_1, \ldots, f_L\}$, consisting of all linear combinations of functions $f_\ell (\ell \in \mathbb{N}_L)$, is a low-dimensional subspace in $\mathcal{H}_0$. Let $\vec{g} = (g_1, \ldots, g_L)$. Denote an operator $\mathcal{D}_{\vec{g}} : \mathbb{R}^L \mapsto \mathcal{H}_0$,

$$\mathcal{D}_{\vec{g}}(\mathbf{c}) = \sum_{\ell=1}^{L} c_\ell g_\ell, \tag{4}$$

where $\mathbf{c} = (c_1, \ldots, c_L)^T \in \mathbb{R}^L$. Let $\mathrm{Ran}(\mathcal{D}_{\vec{g}})$ be the range of $\mathcal{D}_{\vec{g}}$ and $\mathcal{D}_{\vec{g}}^*$ be the adjoint operator of $\mathcal{D}_{\vec{g}}$. Then, $\mathrm{Ran}(\mathcal{D}_{\vec{g}}) = \mathrm{span}\{g_\ell : \ell \in \mathbb{N}_L\}$. It is easy to see that $\mathcal{D}_{\vec{g}}$ is a compact operator, and the dimension of $\mathrm{Ran}(\mathcal{D}_{\vec{g}})$ is finite and determined by the number of nonzero singular values of the operator $\mathcal{D}_{\vec{g}}$. In order to enforce $\mathrm{span}\{g_\ell : \ell \in \mathbb{N}_L\}$ lying in a low-dimensional subspace in $\mathcal{H}_0$, we can choose the following regularization term $J_0(\vec{g}) = \#\{\sigma : \sigma \text{ is a nonzero singular value of } \mathcal{D}_{\vec{g}}\}$, which equals to the number of nonzero eigenvalues of $\sqrt{\mathcal{D}_{\vec{g}}^* \mathcal{D}_{\vec{g}}}$, and regularizes the dimension of $\mathrm{Ran}(\mathcal{D}_{\vec{g}})$. However, this regularization term is difficult to calculate as it is both nonconvex and nonsmooth. Motivated by the theory of compressed sensing and matrix completion [22,23], we use a convex relaxation of $J_0(\vec{g})$ by taking the ℓ_1 norm of all eigenvalues of $\sqrt{\mathcal{D}_{\vec{g}}^* \mathcal{D}_{\vec{g}}}$ as the regularization term, that is,

$$J_1(\vec{g}) = \|\mathcal{D}_{\vec{g}}\|_* = \|\sqrt{\mathcal{D}_{\vec{g}}^* \mathcal{D}_{\vec{g}}}\|_*, \tag{5}$$

where $\|\mathcal{D}_{\vec{g}}\|_* = \|\sqrt{\mathcal{D}_{\vec{g}}^* \mathcal{D}_{\vec{g}}}\|_* = \sum_i \sigma_i$ with σ_i being the i-th singular value of $\mathcal{D}_{\vec{g}}$.

We prove Theorem 1 in Material S1, which shows that the regularization term $J_1(\vec{g})$ is convex, and can be rewritten as $J_1(\vec{g}) = \|\sqrt{G(\vec{g})}\|_*$, where $G(\vec{g})$ is an $L \times L$ square matrix with the (i,j) entry being $\langle g_i, g_j \rangle_{\mathcal{H}_0}$, the inner product between g_i and g_j in $\mathcal{H}_0$.

Theorem 1. *Let $g_i \in \mathcal{H}_0, i = 1, \ldots, L$ and $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$ be the inner product in $\mathcal{H}_0$. The operator $\mathcal{D}_{\vec{g}}$ is defined by (4). Then*

1. *for any $h \in \mathcal{H}_0$, we have $\mathcal{D}_{\vec{g}}^* h = (\langle g_1, h \rangle_{\mathcal{H}_0}, \ldots, \langle g_L, h \rangle_{\mathcal{H}_0})^T$.*
2. *$\mathcal{D}_{\vec{g}}^* \mathcal{D}_{\vec{g}}$ is a linear operator from $\mathbb{R}^L \mapsto \mathbb{R}^L$ and $\mathcal{D}_{\vec{g}}^* \mathcal{D}_{\vec{g}} = G(\vec{g}) : = (\langle g_i, g_j \rangle_{\mathcal{H}_0})_{i,j \in \mathbb{N}_L}$.*
3. *$J_1(\vec{g})$ is convex. That is, for any $\vec{g}^1 = (g_1^1, \ldots, g_L^1)^T \in \mathcal{H}_0^L, \vec{g}^2 = (g_1^2, \ldots, g_L^2)^T \in \mathcal{H}_0^L$ and $\alpha \in [0,1]$, we have $J_1(\alpha \vec{g}^1 + (1-\alpha)\vec{g}^2) \le \alpha J_1(\vec{g}^1) + (1-\alpha) J_1(\vec{g}^2)$.*

Based on the above formulation and using least square error for the data fitting term, we therefore propose to learn gene expression programs in Hilbert space $\mathcal{H}_0$ by minimizing the following objective function

$$R(\vec{f}) = \frac{1}{2m} \sum_{\ell=1}^{L} \sum_{i=1}^{m} (y_i^\ell - f_\ell(\mathbf{x}_i))^2 + \lambda \|\sqrt{G(\vec{f})}\|_* \tag{6}$$

where $\lambda > 0$ is a regularization parameter. We will refer to this model as *nonlinear low-rank model*.

Linear case. Next we will show that model (6) can be viewed as a generalization of the low-rank model (3) from linear setting to nonlinear setting. We assume that each target $\ell \in \mathbb{N}_L$ is well described by a linear function defined, for every $\mathbf{x} \in \mathbb{R}^d$, as $f_\ell(\mathbf{x}) = \omega_\ell^T \mathbf{x}$, where ω_ℓ is a fixed vector of coefficients. As these linear functions are uniquely determined by those coefficients $\omega_\ell, \ell \in \mathbb{N}_L$, we can define the inner product for linear functions as

$$\langle w_i^T \mathbf{x}, w_j^T \mathbf{x} \rangle = \langle w_i, w_j \rangle_{\mathbb{R}^d} = w_i^T w_j. \tag{7}$$

That is, we have implicitly chosen the Hilbert space $\mathcal{H}_0 = \mathbb{R}^d$. Denote $W = (\omega_1, \ldots, \omega_T)$. Using (7), we have $G(\vec{f}) = (\langle f_i, f_j \rangle)_{ij} = (\langle \omega_i, \omega_j \rangle)_{ij} = W^T W$. Note that $\|\sqrt{W^T W}\|_* = \|W\|_*$. Therefore, (6) can be reformulated as

$$R(\vec{f}) = \frac{1}{2m} \sum_{\ell=1}^{L} \sum_{i=1}^{m} (y_i^\ell - \omega_\ell^T \mathbf{x}_i)^2 + \lambda \|W\|_*. \tag{8}$$

Kernel case. Reproducing kernel Hilbert space $\mathcal{H}_K$ is widely used in statistical inference and machine learning [24–26]. It is associated with a Mercer kernel K which is a continuous, symmetric and positive semidefinite function [27]. We denote its inner product as $\langle \cdot, \cdot \rangle_K$. The reproducing property of $\mathcal{H}_K$ states that $f(\mathbf{x}) = \langle f, K_\mathbf{x} \rangle_K$, for all $f \in \mathcal{H}_K$.

The nonlinear low-rank model (6) can be much simplified when $\mathcal{H}_0 = \mathcal{H}_K$. We prove the following representer theorem in Material S1.

Theorem 2. *Given a data set $\mathbf{z} : = \{\mathbf{x}_i, y_i^\ell\}_{i=1}^m$, then the minimizer*

$$\vec{f}^\mathbf{z} = \arg \min_{f_\ell \in \mathcal{H}_K} R_1(\vec{f}) \tag{9}$$

exists and each component $f_\ell^{\mathbf{z}}$ takes the following form

$$f_\ell^{\mathbf{z}} = \sum_{i=1}^{m} c_i^{\ell,\mathbf{z}} K(\mathbf{x}_i,\mathbf{x}),$$

where $c_i^{\ell,\mathbf{z}} \in \mathbb{R}$ for $i \in \mathbb{N}_m, \ell \in \mathbb{N}_L$.

As a consequence, the minimizer of (6) exists, and each component of $\vec{f}$ lies in the finite dimensional space spanned by $\{K_{\mathbf{x}_i} : i \in \mathbb{N}_m\}$, where $K_{\mathbf{x}_i}(\cdot) = K(\mathbf{x}_i,\cdot)$. More specifically, we can show that the solution to the nonlinear low-rank model (6) is

$$f_\ell(\mathbf{x}) = \sum_{i=1}^{m} c_i^\ell K(\mathbf{x}_i,\mathbf{x})$$

for each $l = 1,\cdots,L$, where c_i^l's are the coefficients. Furthermore, it can be shown that the coefficients are determined as the optimal solution that minimizes the following convex function

$$\min_{C \in \mathbb{R}^{m\times L}} \Phi(C) = \frac{1}{2m}\sum_{\ell=1}^{L}\sum_{i=1}^{m}(y_i^\ell - \mathbf{c}_\ell^T \mathbf{k}_i)^2 + \lambda \|\sqrt{C^T \mathbf{K} C}\|_*, \qquad (10)$$

where $\mathbf{c}_\ell = (c_1^\ell,\ldots,c_m^\ell)^T$ is a column vector, $C = (\mathbf{c}_1,\ldots,\mathbf{c}_L)$ is an $m \times L$ matrix, and $\mathbf{K} = (K(\mathbf{x}_i,\mathbf{x}_j))_{i,j=1}^m$ is an $m \times m$ with the i-th column denoted by $\mathbf{k}_i$. The problem (10) is of finite dimension, and next we describe an algorithm to solve it.

Algorithms

In this section, we derive computational algorithms to solve low-rank regularized linear model (3) and nonlinear model (10).

Low-rank regularized linear model. Decompose the objective function (3), $F(W) = \frac{1}{2m}\|Y - XW\|_F^2 + \lambda\|W\|_*$ into two parts with $f(W) = \frac{1}{2m}\|Y - XW\|_F^2$ and $g(W) = \lambda\|W\|_*$. The first component $f(W)$ is both convex and differentiable, and $\nabla f(W) = \frac{1}{m}X^T(XW - Y)$. However, the second component $g(W)$ is not differentiable, although it is still convex.

Define $Q_L(W,D)$ to be

$$\begin{aligned} Q_L(W,D) = {} & \frac{1}{2m}\|Y - XD\|_F^2 + \langle W - D, \nabla f(W)\rangle + \\ & \frac{L}{2}\|W - D\|_F^2 + \lambda\|W\|_*, \end{aligned} \qquad (11)$$

where $D \in \mathbb{R}^{m\times L}$ is a given matrix and L is a positive scalar. $Q_L(W,D)$ can be viewed as an approximation of $F(W)$ around D, and the approximation is accurate when W is sufficiently close to D. Although it is still a non-differentiable function of W, there exists a unique minimizer of $Q_L(W,D)$ for a given D, and the solution can be written down explicitly. Denote the minimizer of $Q_L(W,D)$ for a given D by $p_L(D)$. Next we write down an explicit formula for $p_L(D)$.

Given a matrix $A \in \mathbb{R}^{n_1 \times n_2}$ of rank r, denote the singular value decomposition of A by $A = U\Sigma V^T$, where $U \in \mathbb{R}^{n_1 \times r}, V \in \mathbb{R}^{r \times n_2}, U^T U = I, V^T V = I$, and $\Sigma = diag(\sigma_1,\ldots,\sigma_r)$ with $\sigma_i > 0$ being singular values. For any $\tau > 0$, define the following soft-thresholding operator of matrix A

$$\mathcal{D}_\tau(A) = U\mathcal{D}_\tau(\Sigma)V^T, \qquad (12)$$

where $\mathcal{D}_\tau(\Sigma) = \mathrm{diag}((\sigma_1 - \tau)_+, (\sigma_2 - \tau)_+, \ldots, (\sigma_r - \tau)_+)$ with $(x)_+ = 0$ if $x < 0$ and x otherwise. With this definition, it can be shown that [17]

$$p_L(D) = \mathcal{D}_{\frac{\lambda}{L}}(D + \frac{1}{L}\nabla f(D)) \qquad (13)$$

Using the above-mentioned notations and definitions, a Nesterov's algorithm [28] can be derived to solve the low-rank regularized linear model (3). The detailed steps are shown in Algorithm 1.

Algorithm 1. Nesterov's algorithm for solving (3) with backtracking Initialize $L_0, W_0 \in \mathbb{R}^{m\times L}$. Set $D_1 = W_0, t_1 = 1$

repeat

Step k, 1) Find the smallest nonnegative integers i_k such that with $\tilde{L} = 2^{i_k} L_{k-1}$

$$F(p_{\tilde{L}}(D_k)) \le Q_{\tilde{L}}(p_{\tilde{L}}(D_k), D_k).$$

Step k, 2) Set $L_k = 2^{i_k} L_{k-1}$ and compute

$$W_k = p_{L_k}(D_k),$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2},$$

$$D_{k+1} = W_k + \frac{t_k - 1}{t_{k+1}}(W_k - W_{k-1}).$$

until

Convergence

Low-rank regularized nonlinear model. We first convert the problem (10) into a more compact form by changing the optimization variables. Then we derive an algorithm to solve the problem based on the Nesterov's method [29,30]. Note that $\mathbf{K}$ is symmetric and positive semidefinite, so is its square root $\mathbf{K}^{\frac{1}{2}}$. Denote the i-th column of $\mathbf{K}^{\frac{1}{2}}$ by $\mathbf{k}_i^{\frac{1}{2}}$, i.e., $\mathbf{K}^{\frac{1}{2}} = (\mathbf{k}_1^{\frac{1}{2}},\ldots,\mathbf{k}_m^{\frac{1}{2}})$. Let $\tilde{C} = \mathbf{K}^{\frac{1}{2}}C$ and write $\tilde{C} = (\tilde{\mathbf{c}}_1,\ldots,\tilde{\mathbf{c}}_L)$. Then $\Phi(C)$ in equation (10) can be rewritten as a function of $\tilde{C}$

$$\begin{aligned} \Psi(\tilde{C}) &= \frac{1}{2m}\sum_{\ell=1}^{L}\sum_{i=1}^{m}\left(y_i^\ell - \tilde{\mathbf{c}}_\ell^T \mathbf{k}_i^{\frac{1}{2}}\right)^2 + \lambda\|\tilde{C}\|_* \\ &= \frac{1}{2m}\|Y - \mathbf{K}^{\frac{1}{2}}\tilde{C}\|_F^2 + \lambda\|\tilde{C}\|_*, \end{aligned} \qquad (14)$$

where $Y = (\mathbf{y}_1,\ldots,\mathbf{y}_m)^T$ with $\mathbf{y_i} = (y_i^1,\ldots,y_i^L)^T$. Thus finding a solution $C_{\mathbf{z}}$ of equation (10) is equivalent to identifying,

$$\tilde{C}_{\mathbf{z}} = \arg\min_{\tilde{C} \in \mathbb{R}^{m\times L}} \Psi(\tilde{C}), \qquad (15)$$

followed by setting $C_{\mathbf{z}} = \mathbf{K}^{-\frac{1}{2}}\tilde{C}_{\mathbf{z}}$ where $\mathbf{K}^{-\frac{1}{2}}$ is the (pseudo) inverse of $\mathbf{K}^{\frac{1}{2}}$ when $\mathbf{K}$ is (not) invertible.

Similar to the linear case, we first decompose $\Psi(\tilde{C})$ into two parts with $f(\tilde{C}) = \frac{1}{2m}\|Y - \mathbf{K}^{\frac{1}{2}}\tilde{C}\|_F^2$ and $g(\tilde{C}) = \lambda\|\tilde{C}\|_*$. $f(\tilde{C})$ is differentiable and $\nabla f(\tilde{C}) = \frac{1}{m}(\mathbf{K}^{\frac{1}{2}})^T(\mathbf{K}^{\frac{1}{2}}\tilde{C} - Y)$.

Define $Q_L(\tilde{C}, D)$ as the following form,

$$Q_L(\tilde{C},D) = \frac{1}{2m}\|Y - \mathbf{K}^{\frac{1}{2}}D\|_F^2 + \langle \tilde{C} - D, \nabla f(\tilde{C})\rangle + \frac{L}{2}\|\tilde{C} - D\|_F^2 + \lambda\|\tilde{C}\|_*,$$

where $D \in \mathbb{R}^{m \times L}$ is a given matrix and $L > 0$.

The unique minimizer of $Q_L(\tilde{C}, D)$ is denoted by $p_L(D)$, and we apply soft-thresholding operator (12) to give the explicit form of $p_L(D)$,

$$p_L(D) = \mathcal{D}_{\frac{\lambda}{L}}(D + \frac{1}{L}\nabla f(D)). \tag{16}$$

With the above-mentioned notations and definitions, we derive Algorithm 3(′)@ to solve problem (10).

Algorithm 2. Nesterov's algorithm for solving (10) with backtracking Initialize $L_0, \tilde{C}_0 \in \mathbb{R}^{m \times L}$. Set $D_1 = \tilde{C}_0, t_1 = 1$

repeat

Step k, 1) Find the smallest nonnegative integers i_k such that with $\tilde{L} = 2^{i_k}L_{k-1}$

$$\Psi(p_{\tilde{L}}(D_k)) \le Q_{\tilde{L}}(p_{\tilde{L}}(D_k), D_k).$$

Step k, 2) Set $L_k = 2^{i_k}L_{k-1}$ and compute

$$\tilde{C}_k = p_{L_k}(D_k),$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2},$$

$$D_{k+1} = \tilde{C}_k + \frac{t_k - 1}{t_{k+1}}(\tilde{C}_k - \tilde{C}_{k-1}).$$

until

Convergence

Return $C_{\mathbf{z}} = \mathbf{K}^{-\frac{1}{2}}\tilde{C}_k$

Results

Next we test the performance of the low-rank regularization models (both linear and nonlinear) described above on three real biological datasets. We will compare the performance of our models to the Lasso model (2), which imposes a sparsity constraint within a linear regression framework, and the SiMoNe model [16], which models the modularity structure with a Gaussian graphical model framework. In each of the three experiments, we divide the data into training and test datasets. The models are trained based on training data, and the performance of the resulting models are then evaluated based on test data. We use root-mean-square error (RMSE) to measure the differences between values predicted by each model and the the values actually observed. The average RMSE over all target genes measured on the training and test data will be called training and testing error, respectively. The regularization parameter λ of our models is automatically tuned through ten-fold cross-validation based only on the training data, and is set to be the value that gives rise to the best cross-validation performance.

Yeast gene expression data

We tested our models on a yeast gene expression dataset [31], which contains mRNA measurements of 2,355 genes of *Saccharomyces cerevisiae* responding to diverse environmental transitions including temperature shocks, amino acid starvation, hydrogen peroxide, etc. Overall the dataset contains microarray measurements of yeast genes in 173 environmental transitions (will be referred to as samples). The dataset was rescaled to make the expression values of each gene to be mean 0 and variance 1 across the 173 samples. We used a list of 321 candidate regulators manually compiled in [15] based on biological annotations (including transcription factors and signaling molecules) as our regulator genes. We used this dataset to learn the regulatory relationship between these 321 regulators and the other 2,034 genes, which will be called targets. We benchmarked the performance of our and control models using ten-fold cross-validation. More specially, we randomly partitioned the 173 samples into 10 nonoverlapping subsets. Each model was trained using nine of the ten subsets, the regularization parameter λ in each model was tuned via cross validation on 10 dimensional logarithmically spaced vector ranging between 10^{-3} and 10^2. After choosing the lambda, the test performance of the learned model was then measured using the remaining subset. We used root-mean-square error (RMSE) to measure the differences between values predicted by each model and the the values actually observed. The average RMSE over all target genes measured on the training and test data will be called training and testing error, respectively.

The training and test performance of four models - SiMoNe [16], linear low-rank (3), nonlinear low-rank (10), and Lasso (2), on the yeast gene expression dataset is summarized in Table 1. The linear low-rank model (3) reduces training error by 10.5% and testing error by 6.1% when compared to the Lasso model (2). The regression-based models, including ours and Lasso, significantly outperform SiMoNe in both training and test performance. If we look specifically at the prediction accuracy of each target gene, we note that 84% of the targets are predicted more accurately by the linear low-rank model than by Lasso (Figure 1). We used an ANOVA kernel to train the nonlinear low-rank model [32]. Although the training error of the nonlinear model is 6.9% smaller than that of the linear low-rank model, the testing performance of the two models is similar. The optimal rank returned by the linear model is 78 and the one returned by the nonlinear model is 88, suggesting the existence of approximately 78–88 regulatory modules that are active in this dataset.

Human hematopoietic gene expression data

We also tested our models on a human hematopoietic gene expression dataset [33], which measures mRNA expression values of human genes during hematopoietic differentiation. The dataset contains expression profiles of 8,968 genes in 38 hematopoietic

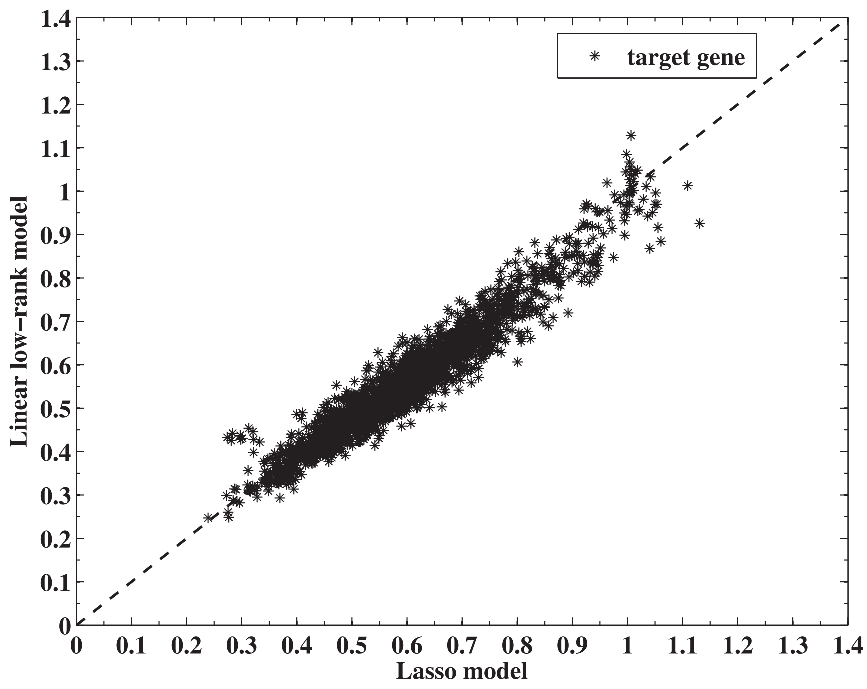

Figure 1. Comparison of the testing performance of the linear low-rank regularization model vs. Lasso on the yeast gene expression dataset. Each * indicates one target gene. X-axis represents the test RMSE of the Lasso model, whereas Y-axis represents the test RMSE of the linear low-rank model. The figure shows that the low-rank model yields lower testing error than Lasso for most target genes.
doi:10.1371/journal.pone.0082146.g001

states with a total of 211 experiment conditions (will also be referred to as samples). We used a list of 523 candidate regulators manually complied in [33] based on biological annotations (including important transcriptional regulators or signalling molecules previously implicated in hematopoietic differentiation) as our regulator genes. Among the remaining non-regulator genes, we removed genes with low variance across the samples, and kept only the top 1000 genes with highest variance. These 1000 genes will be called target genes, and our goal is to learn the regulatory relationship between the 523 regulators and the 1000 target genes. We rescaled the expression of each gene (both regulators and targets) to be mean 0 and variance 1 across the samples. Similar to the yeast dataset, we benchmarked the performance of our and control models on this dataset using ten-fold cross-validation, and RMSE was used to measure both training and testing errors.

The training and test performance of four models - SiMoNe [16], linear low-rank (3), nonlinear low-rank (10), and Lasso (2), on the human hematopoietic gene expression dataset is summarized in Table 2. The linear low-rank model (3) reduces training error by 20.0% and testing error by 3.2% when compared to the Lasso model. Similar to the yeast dataset, the regression-based models outperform SiMoNe by a large margin. If we look specifically at the prediction accuracy of each target gene, we find that 70% of the targets are predicted more accurately by the linear low-rank model than by Lasso (Figure 2). Similar to the yeast dataset, we used an ANOVA kernel to train the nonlinear low-rank model. The training error of the nonlinear model is 5.0% smaller than that of the linear low-rank model, but their testing performance is similar. The optimal rank returned by the linear model is 112 and the one returned by the nonlinear model is 109, suggesting that approximately 109–112 regulatory modules are active in this hematopoietic gene expression dataset.

Table 1. Root-mean-squared error (RMSE) comparison among different models on the yeast gene expression data.

Model	Training error	Testing error
SiMoNe	—	1.0074±0.0650
Lasso	0.3897±0.0045	0.6124±0.0583
Linear low-rank	0.3488±0.0014	0.5750±0.0053
Nonlinear low-rank	0.3249±0.0063	0.5752±0.0054

"Lasso" represents model (2), "Linear low-rank" represents model (3), and "Nonlinear low-rank" represents model (6) with ANOVA kernel. SiMoNe is the model described by Chiquet et al. [16]. Both training and testing errors are measured in terms of RMSE averaged over all target genes. Shown here are mean ± standard deviation values of RMSEs in ten different runs.
doi:10.1371/journal.pone.0082146.t001

Connectivity map data

The third dataset we have experimented with is the connectivity map data provided by Lamb et al. [2], which contains gene expression measurements of human cells responding to diverse treatments with chemical compounds and genetic reagents. The connectivity map data contains microarray measurements of human genes in thousands of profiles (will be referred to as samples). For regulator genes, we used 978 "landmark" genes determined by the connectivity map project as the set of genes that are most predictive of the expression of the other genes (Aravind Subramanian, personal communication). Among the remaining non-regulator genes, we used the top 10,000 genes with highest variances across samples as our target genes. We randomly selected 1000 samples from this dataset to benchmark the performance of our and other models (we were unable to use all samples due to computational constraints.) Our goal is to learn the regulatory relationship between the 10,000 target genes and the 978 landmark genes based on these 1000 samples.

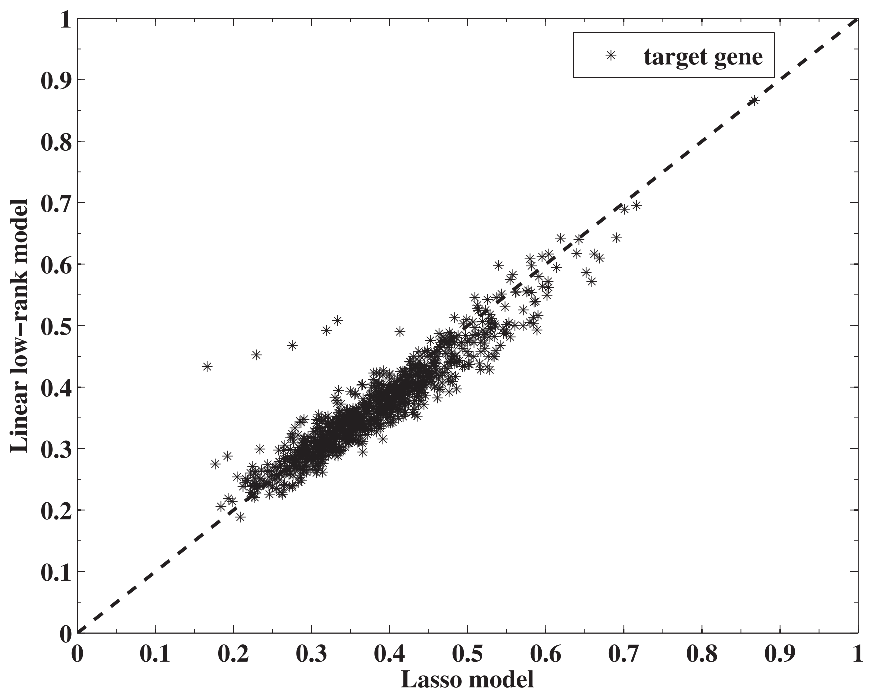

Figure 2. Comparison of the testing performance of the linear low-rank regularization model vs. Lasso on the human hematopoietic gene expression dataset. Each * indicates one target gene. X-axis represents the test RMSE of the Lasso model, whereas Y-axis represents the test RMSE of the linear low-rank model. The figure shows that the low-rank model yields lower testing error than Lasso for most target genes.
doi:10.1371/journal.pone.0082146.g002

We benchmarked the performance of our and control models using ten-fold cross-validation. More specifically, we randomly partitioned the samples into ten nonoverlapping subsets. Each model was trained using nine of the ten subsets, and the test performance of the learned model was then measured using the remaining subset. We used root-mean-square error (RMSE) to measure the differences between values predicted by each model and the the values actually observed. The average RMSE over all target genes measured on the training and test data will be called training and testing error, respectively.

We were unable to obtain SiMoNe results after hours of running the program, which might be due to the large number of genes contained in this dataset, and the fact that the inverse covariance matrix between these genes is too large to be handled by SiMoNe. So next we will focus on comparing the performance of our models to Lasso. The performance of the linear low-rank (3), nonlinear low-rank (10), and Lasso (2), on the connectivity map data is summarized in Table 3. The nonlinear low-rank model achieves the lowest testing error in this dataset. The testing performances of linear low-rank model and Lasso are similar, with the testing errors of both models 4.31% higher than the nonlinear low-rank model. If we compare the prediction performance of each target gene, 88% of the target genes are predicted more accurately by the nonlinear low-rank model than by Lasso.

Our algorithms were implemented in Matlab and run on the platform of Intel Xeon E5-4617 - 2.9 GHz 1-Core CPU with 128 GB memory. The CPU times of running our algorithms on the three datasets are shown in Table 4. The time complexity of Algorithm 1 and 2 is mainly determined by the singular value decomposition step. Exact singular value decomposition of a $m \times n$

Table 2. Root-mean-squared error (RMSE) comparison among different models on the human hematopoietic gene expression data.

Model	Training error	Testing error
SiMoNe	—	0.9987±0.0400
Lasso	0.2345±0.0024	0.3881±0.0265
Linear low-rank	0.1877±0.0030	0.3758±0.0265
Nonlinear low-rank	0.1783±0.0005	0.3767±0.0265

"Lasso" represents model (2), "Linear low-rank" represents model (3), and "Nonlinear low-rank" represents model (6) with ANOVA kernel. SiMoNe is the model described by Chiquet et al. [16]. Both training and testing errors are measured in terms of RMSE averaged over all target genes. Shown here are mean ± standard deviation values of RMSEs in ten different runs.
doi:10.1371/journal.pone.0082146.t002

Table 3. Root-mean-squared error (RMSE) comparison among different models on the connectivity map gene expression data.

Model	Training error	Testing error
Lasso	0.4943±0.0008	0.7077±0.0134
Linear low-rank	0.5157±0.0004	0.7000±0.0123
Nonlinear low-rank	0.4025±0.0005	0.6772±0.0125

"Lasso" represents model (2), "Linear low-rank" represents model (3), and "Nonlinear low-rank" represents model (6) with ANOVA kernel. Both training and testing errors are measured in terms of RMSE averaged over all target genes. Shown here are mean ± standard deviation values of RMSEs in ten different runs.
doi:10.1371/journal.pone.0082146.t003

Table 4. CPU time of running the linear and nonlinear low-rank models.

Dataset	Linear low-rank (min)	Nonlinear low-rank (min)
Yeast gene expression	1.6	0.8
Human hematopoietic gene expression	1.0	0.4
Connectivity map gene expression	1067	869

doi:10.1371/journal.pone.0082146.t004

matrix has the time complexity of $\mathcal{O}(\min(m^2n, mn^2))$. In our algorithms, n corresponds to the number of targets. However, m corresponds to the number of regulators in the linear model, and the number of samples in the nonlinear model. So when the number of samples is smaller than the number of regulators, the nonlinear model actually runs fasters than the linear model (See Table 4).

Discussion

Gene expression program learning is an important problem in both basic research as well as practical and applied settings of biotechnology development. In this paper, we formulate the gene expression program learning as a multi-target (more specifically L-target) regression problem and use Tikhonov regularization to constrain the space of the L-target mapping. We propose a new form of regularization that constrains the number of independent connectivity patterns between regulator genes and target genes. We use the rank of the connectivity matrix from regulators to targets to represent the number of independent connectivity patterns, and approximate the rank of the matrix using its nuclear norm. The resulting low-rank regularization problem is convex, and we provide an efficient algorithm to find its globally optimal solution.

Previously, in gene expression program learning each target gene is usually treated separately. Because the expression of many genes are highly correlated, it would be beneficial to learn their expression regulation jointly instead of separately. However, it was unclear before on how to model the regulatory relationship from regulators to target genes jointly such that the resulting model is both computationally efficient and able to take the constraints between targets into account. The low-rank regularization provides an effective and yet computationally efficient framework for considering all target genes simultaneously. Experiments on two real gene expression datasets demonstrate that the low-rank model outperforms the Lasso model, one of the most widely used regularization method in gene expression program learning, in terms of prediction accuracy in both datasets.

We showed that the low-rank model can also be generalized to nonlinear settings, where we constrain the dimension of the hypothesis space of the L-target regression function. We proved that the resulting problem is still convex and derived an efficient algorithm to find its globally optimal solution. We tested the nonlinear low-rank model on the gene expression datasets. The nonlinear low-rank model produces better testing performance than the linear low-rank model in some datasets, but is comparable to the linear model in other datasets. The lack of improvement comparing to the linear one in some datasets might be due to a) the fact that the number of samples used in these two datasets might be too small to fit a more complex model, and b) the kernel we have tried (ANOVA, Gaussian, and polynomial) might not be a good fit for the gene expression program learning. We expect that the nonlinear model will improve when a larger number of samples become available. So finding or designing the right kernel specifically for gene expression program learning will be the key to improving the nonlinear model.

The two forms of regularization, low-rank and sparsity, described in this paper are complementary to each other, considering two different aspects of gene regulation. A future direction is to combine these two regularizations into a single framework to constrain the connectivity matrix to be simultaneously sparse and low rank.

Supporting Information

Material S1 For proving Theorems 1 and 2.
(PDF)

Acknowledgments

We thank A Subramanian for providing us the connectivity map data and for helpful discussions.

Author Contributions

Conceived and designed the experiments: GY XX. Performed the experiments: GY MT JC QN XX. Analyzed the data: GY MT JC QN XX. Wrote the paper: GY QN XX.

References

1. Bansal M, Belcastro V, Ambesi-Impiombato A, Di Bernardo D (2007) How to infer gene networks from expression profiles. Molecular systems biology 3.
2. Lamb J, Crawford E, Peck D, Modell J, Blat I, et al. (2006) The connectivity map: using gene-expression signatures to connect small molecules, genes, and disease. Science 313: 1929.
3. Friedman N (2004) Inferring cellular networks using probabilistic graphical models. Science 303: 799–805.
4. Friedman J, Hastie T, Tibshirani R (2008) Sparse inverse covariance estimation with the graphical lasso. Biostatistics 9: 432–441.
5. Grzegorczyk M, Husmeier D (2011) Improvements in the reconstruction of time-varying gene regulatory networks: dynamic programming and regularization by information sharing among genes. Bioinformatics 27: 693–699.
6. Faith JJ, Hayete B, Thaden JT, Mogno I, Wierzbowski J, et al. (2007) Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. PLoS biology 5: e8.
7. Margolin AA, Wang K, Lim WK, Kustagi M, Nemenman I, et al. (2006) Reverse engineering cellular networks. Nature Protocols 1: 662–671.
8. Gustafsson M, Hornquist M, Lombardi A (2005) Constructing and analyzing a large-scale geneto-gene regulatory network lasso-constrained inference and biological validation. Computational Biology and Bioinformatics, IEEE/ACM Transactions on 2: 254–261.
9. Gardner T, Di Bernardo D, Lorenz D, Collins J (2003) Inferring genetic networks and identifying compound mode of action via expression profiling. Science 301: 102.
10. De Jong H (2002) Modeling and simulation of genetic regulatory systems: a literature review. Journal of computational biology 9: 67–103.
11. Christley S, Nie Q, Xie X (2009) Incorporating existing network information into gene network inference. PloS one 4: e6799.
12. Lee S, Dudley A, Drubin D, Silver P, Krogan N, et al. (2009) Learning a prior on regulatory potential from eqtl data. PLoS genetics 5: e1000358.

13. Ye GB, Wang Y, Chen Y, Xie X (2011) Efficient latent variable graphical model selection via split bregman method. arXiv preprint arXiv:11103076.
14. Tibshirani R (1996) Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B (Methodological) : 267–288.
15. Segal E, Shapira M, Regev A, Pe'er D, Botstein D, et al. (2003) Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nature genetics 34: 166–176.
16. Chiquet J, Smith A, Grasseau G, Matias C, Ambroise C (2009) Simone: Statistical inference for modular networks. Bioinformatics 25: 417–418.
17. Cai J, Candès E, Shen Z (2010) A singular value thresholding algorithm for matrix completion. SIAM Journal on Optimization 20: 1956–1982.
18. Chen J, Liu J, Ye J (2012) Learning incoherent sparse and low-rank patterns from multiple tasks. ACM Transactions on Knowledge Discovery from Data (TKDD) 5: 22.
19. Candès E, Li X, Ma Y, Wright J (2011) Robust principal component analysis? Journal of the Association for Computing Machinery 58: 1–37.
20. Ying Y, Huang K, Campbell C (2009) Sparse metric learning via smooth optimization. Advances in Neural Information Processing Systems 22.
21. Hoerl A, Kennard R (1970) Ridge regression: Biased estimation for nonorthogonal problems. Technometrics : 55–67.
22. Candès E, Recht B (2009) Exact matrix completion via convex optimization. Foundations of Computational Mathematics 9: 717–772.
23. Donoho D (2006) Compressed sensing. IEEE Transactions on Information Theory 52: 1289–1306.
24. Cristianini N, Shawe-Taylor J (2004) An introduction to support Vector Machines: and other kernel-based learning methods. Cambridge university press.
25. Schölkopf B, Smola A (2002) Learning with kernels: Support vector machines, regularization, optimization, and beyond. MIT press.
26. Vapnik VN (1998) Statistical learning theory. Adaptive and Learning Systems for Signal Processing, Communications, and Control. New York: John Wiley & Sons Inc., xxvi+736 pp. A Wiley-Interscience Publication.
27. Aronszajn N (1950) Theory of reproducing kernels. Trans Amer Math Soc 68: 337–404.
28. Nesterov Y (2005) Smooth minimization of non-smooth functions. Mathematical Programming 103: 127–152.
29. Beck A, Teboulle M (2009) A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J Imaging Sci 2: 183–202.
30. Nesterov Y (2007) Gradient methods for minimizing composite objective function. Center for Operations Research and Econometrics (CORE), Catholic University of Louvain, Tech Rep 76.
31. Gasch A, Spellman P, Kao C, Carmel-Harel O, Eisen M, et al. (2000) Genomic expression programs in the response of yeast cells to environmental changes. Science 11: 4241.
32. Thomas Hofmann BS, Smola AJ (2008) Kernel method in machine learning. The Annals of Statistics 36: 1171.
33. Novershtern N, Subramanian A, Lawton L, Mak R, Haining W, et al. (2011) Densely interconnected transcriptional circuits control cell states in human hematopoiesis. Cell 144: 296–309.