Journal of Machine Learning Research 14 (2013) 3073-3104 Submitted 5/12; Revised 5/13; Published 10/13

Stationary-Sparse Causality Network Learning

Yuejia He YUEJIAHE@UFL.EDU
Department of Electrical and Computer Engineering
University of Florida
Gainesville, FL 32611-6130

Yiyuan She YSHE@STAT.FSU.EDU
Department of Statistics
Florida State University
Tallahassee, FL 32306-4330

Dapeng Wu WU@ECE.UFL.EDU
Department of Electrical and Computer Engineering
University of Florida
Gainesville, FL 32611-6130

Editor: Hui Zou

Abstract

Recently, researchers have proposed penalized maximum likelihood to identify network topology underlying a dynamical system modeled by multivariate time series. The time series of interest are assumed to be stationary, but this restriction is never taken into consideration by existing estimation methods. Moreover, practical problems of interest may have ultra-high dimensionality and obvious node collinearity. In addition, none of the available algorithms provides a probabilistic measure of the uncertainty for the obtained network topology which is informative in reliable network identification. The main purpose of this paper is to tackle these challenging issues. We propose the $\mathbf{S}^2$ learning framework, which stands for *stationary-sparse* network learning. We propose a novel algorithm referred to as the Berhu iterative sparsity pursuit with stationarity (BISPS), where the Berhu regularization can improve the Lasso in detection and estimation. The algorithm is extremely easy to implement, efficient in computation and has a theoretical guarantee to converge to a global optimum. We also incorporate a screening technique into BISPS to tackle ultra-high dimensional problems and enhance computational efficiency. Furthermore, a stationary bootstrap technique is applied to provide connection occurring frequency for reliable topology learning. Experiments show that our method can achieve stationary and sparse causality network learning and is scalable for high-dimensional problems.

Keywords: stationarity, sparsity, Berhu, screening, bootstrap

1. Introduction

There has been an increasing interest in identifying network dynamics and topologies in the emerging scientific discipline of network science (e.g., Newman and Watts, 2006; Lewis, 2009). In a dynamical network, the evolution of a node is controlled not only by itself, but also by other nodes. For example, in the gene regulatory network (Faith et al., 2007), the expression levels of genes influence each other, following some dynamic rules, which connect the genes together and form a dynamical system. If the topology and evolution rules of the network are known, we can analyze the regulation between genes or detect unusual behaviors to help diagnose and cure genetic diseases. Similarly, the modeling and estimation of dynamical networks are of great importance for various domains including stock market (Mills and Markellos, 2008), brain network (Bullmore and Sporns, 2009) and social network (Hanneke et al., 2010). To accurately identify the topology and dynamics underlying those networks, scientists are devoted to developing appropriate mathematical models and corresponding estimation methods.

In practice, we can obtain discrete observations of the network over a period of time, which can usually be modeled by multivariate time series from a statistical perspective. For example, the fMRI data of the human brain is taken every one minute during a two-hour experiment; stock prices are often recorded daily or weekly. These multivariate time series contain important information of the network topology and dynamics. Vector autoregressive (VAR) process (Sims, 1980) is one of the most commonly used models for characterizing the relations between the time series. In this model, the state of each node is characterized by a time series. The value of a node at a time point is a linear combination of the past values of itself and the nodes regulating it. This kind of regulation relationship is regarded as the *Granger causal connection* (Granger, 1969). By estimating the transition matrix of the model, we can understand the Granger causal relations between nodes.

In estimating the transition matrix, one must bare in mind two most important objectives: first, the estimate fitted on the training data should provide accurate prediction; second, a sparse topology that illustrates the most prominent network connections is desired. Recently, compressive sensing approaches based on penalized maximum likelihood (PML) are applied to achieve accurate prediction and sparse representation simultaneously (Donoho, 2006; Tsaig and Donoho, 2006; Songsiri and Vandenberghe, 2010). Different penalties and algorithms are proposed (Fan and Li, 2006, 2001; Zou, 2006; Zou and Hastie, 2005; Blumensath and Davies, 2010). The ℓ_1 penalty (Tibshirani, 1996) is popular for its computational efficiency and theoretical elegance. Nevertheless, the major problem of the ℓ_1 penalty for dynamical network learning is its incapability of handling collinearity, which typically exists in network data as a result of the interaction between nodes. The elastic net (Zou and Hastie, 2005) uses a linear combination of the ℓ_1 and ℓ_2 penalties to deal with collinearity and large noise. However, its ℓ_2 component may counteract sparsity and bring the so-called "double shrinkage" issue. To improve these drawbacks, we study a new '$\ell_1 + \ell_2$' variant—Berhu (Owen, 2007), which fuses the ℓ_1 and ℓ_2 penalties in a nonlinear fashion and thus can deal with collinearity as well as achieve sufficient sparsity. We propose a Berhu thresholding operator to efficiently solve the Berhu penalized problem.

In real-world problems, raw observations from a dynamical network are usually preprocessed and stationarized before the application of PML (Stock and Watson, 2012; Hsu et al., 2008). Nevertheless, as will be demonstrated in the experiment, it is possible for PML to end up with a nonstationary estimate, due to noise contamination and limited number of observations. Such nonstationary estimates may give unmeaningful prediction results, especially for **long-term** forecasting. Hence, our work, distinguished from the existing ones, focuses on enforcing the stationarity guarantee in network estimation and topology identification, which is of great importance but has never been properly addressed. The stationarity condition of the VAR model is its spectral radius being smaller than one (Reinsel, 1997). This constraint is nonconvex and extremely difficult to tackle directly (Burke et al., 2005; Curtis and Overton, 2012; Overton and Womersley, 1988). Hence, we use a convex relaxation and come up with a stationarity constrained PML problem. We propose an efficient algorithm, the *Berhu iterative sparsity pursuit with stationarity* (BISPS), to achieve *stationary-sparse* ($\mathbf{S}^2$) network learning. This algorithm is very easy to implement and theoretically guaranteed to converge to a global optimum. Experimentation demonstrates that our method can guarantee a stationary and sparse estimate. It not only gives satisfactory identification accuracy, but also outperforms the plain PML method significantly in prediction.

Another challenge in network identification lies in the high dimensionality of the data (Fan and Lv, 2010). For a network with p nodes and n observations, the number of unknown variables in the transition matrix is p^2, and we frequently face practical data sets with $p^2 \gg n$, for example, microarray data sets consisting of thousands of genes but fewer than a hundred observations. This so-called "ultra-high dimensional" problem (Fan and Lv, 2008) adds tremendous difficulties to the inference methods in terms of statistical estimation accuracy as well as computational complexity. To address this challenging issue, we propose two efficient techniques. First, the *quantile thresholding iterative screening* (QTIS) is designed to "preselect" connections for the BISPS algorithm in a supervised manner. QTIS differs from existing screening techniques such as the sure independence screening (Fan and Lv, 2008) in that it takes into account of collinearity in the data. Secondly, we propose the *stationary bootstrap* enhanced BISPS (SB-BISPS). Bootstrap is a nonparametric technique for approximating the distributions of statistics or constructing confidence intervals. Our work applies this powerful tool with stationarity guarantee to network identification and provides a confidence level for the occurrence of each possible connection in the network.

The remainder of the paper is organized as follows: Section 2 introduces the stationary and sparse network model and formulates the $\mathbf{S}^2$ learning framework. Section 3 proposes our algorithms, mainly the Berhu iterative sparsity pursuit with stationarity (BISPS) and the quantile thresholding iterative screening (QTIS), and provides theoretical proofs for their convergence. Section 4 describes the stationary bootstrap enhanced BISPS (SB-BISPS). In Section 5, we show experimental results on synthetic data. In Section 6, we apply the proposed method to the U.S. macroeconomic data. Section 7 concludes our work.

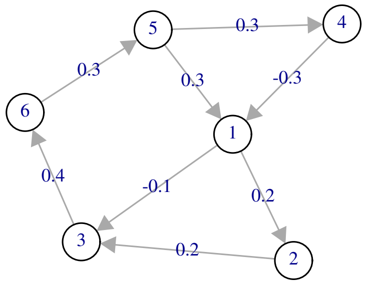
2. The Stationary-Sparse ($\mathbf{S}^2$) Network Learning Framework

Let x be a p-dimensional random vector with each component being a time series associated with one node in a dynamical network, where p is the number of nodes. We are interested in characterizing the observations of x at different time points using mathematical models, based on which we can conduct useful network analysis. In particular, we are interested in understanding the causal relations between nodes and making predictions for future. A commonly used model describes the current state x_t of the system as a linear transformation of its previous state x_{t-1}:

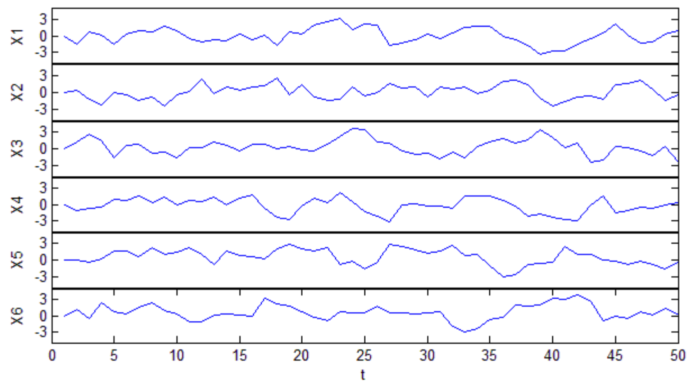
$$x_t = Bx_{t-1} + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \Sigma_\varepsilon), \tag{1}$$

where B is the *transition matrix* and ε_t is random noise. This corresponds to the first-order vector autoregressive (VAR) model (Sims, 1980). It can be generalized to a VAR model with order m, where the current state is a linear combination of the most recent m states. On the other hand, any mth-order VAR model can be converted to a first-order VAR model by appropriately redefining the node variables (Lütkepohl, 2007), and thus we focus on the former one with $m = 1$ in this paper.

In (1), the transition matrix $B = [b_{ij}]_{1 \le i,j \le p}$ describes a network that represents the dynamical system: if $b_{ij} \neq 0$, there is a *Granger causal connection* (Granger, 1969) from node j to node i with weight b_{ij}. In other words, node j Granger-causes node i. For example, for a network with 6 nodes



(a) causality network



(b) multivariate time series

Figure 1: Example of a network (1) with transition matrix (2).

and a transition matrix as

$$B = \begin{pmatrix} 0.6 & 0.2 & -0.1 & 0 & 0 & 0 \\ 0 & 0.6 & 0.2 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 & 0.4 \\ -0.3 & 0 & 0 & 0.7 & 0 & 0 \\ 0.3 & 0 & 0 & 0.3 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0.3 & 0.6 \end{pmatrix}, \tag{2}$$

Figure 1a shows its topology (self-connections are removed). The nodes evolve and interact with each other through the Granger causal connections, resulting in the random processes plotted in Figure 1b. Therefore, matrix B not only illustrates the dynamic rules that govern the evolution of the system, but also captures a linear *causality network* that describes the (Granger) casual relations between nodes.

2.1 Sparse Network Learning by Penalized Maximum Likelihood Estimation

Given n observations of the dynamical network $x_1, \cdots, x_n$, we wish to estimate B. Due to Markov-chain property, we can write the likelihood of B as

$$L(B|x_1, \cdots, x_n) = \prod_{t=2}^{n} f(x_t|x_{t-1}, \cdots, x_1, B) f(x_1|B) = \prod_{t=2}^{n} f(x_t|x_{t-1}, B) f(x_1|B).$$

The exact maximum likelihood (ML) estimate requires solving a nonlinear optimization problem. For simplicity, researchers often use the *conditional* likelihood where the initial state x_1 is assumed to be fixed. Due to normality, we have the conditional likelihood

$$L_c(B) = \prod_{t=2}^{n} f(x_t|x_{t-1}, B) = \prod_{t=2}^{n} (2\pi)^{-p/2} |\Sigma_\varepsilon|^{-1/2} \exp\{-\frac{1}{2}(x_t - Bx_{t-1})^\top \Sigma_\varepsilon^{-1}(x_t - Bx_{t-1})\}.$$

So the (conditional) ML estimate of B can be obtained by solving

$$\min_B \frac{1}{2} \sum_{t=2}^{n} \|x_t - Bx_{t-1}\|_2^2.$$

Letting $Y = [x_2^{\mathsf{T}}, x_3^{\mathsf{T}}, \cdots, x_n^{\mathsf{T}}]^{\mathsf{T}}$, $X = [x_1^{\mathsf{T}}, x_2^{\mathsf{T}}, \cdots, x_{n-1}^{\mathsf{T}}]^{\mathsf{T}}$ and $A = B^{\mathsf{T}}$, we can formulate the problem in matrix form:

$$A_{ML} = \arg\min_{A} l(A) = \frac{1}{2}\|Y - XA\|_F^2.$$

For convenience, we use A instead of B to represent the network in the remainder of the paper. Note that a_{ij} describes the directed connection strength from node i to node j. The estimate $\hat{A}_{ML}$ has been investigated and applied to many real-world data. For stationary process, the consistency and asymptotic efficiency of $\hat{A}_{ML}$ are analyzed in Reinsel (1997). The small-sample properties are discussed in Lütkepohl (2007).

Nevertheless, the plain ML estimation is not ideal for network learning. In practice, X usually demonstrates high collinearity, especially when some nodes have similar dynamical behaviors and when the number of observations is limited. Moreover, the ML estimation does not promote sparsity and consequently the resulting model is difficult to interpret. To improve prediction accuracy and obtain interpretable model, *shrinkage estimation* is necessary. It can be done by adding a penalty and/or constraint. For example, we can estimate A via penalized maximum likelihood (PML):

$$\hat{A}_{PML} = \arg\min_{A} l(A) + P(A; \lambda). \tag{3}$$

We consider only the additive penalties and denote $P(A;\lambda) = \sum_{i,j} P(a_{ij}; \lambda_{ij})$, where $P(\cdot)$ is a penalty function applied to each component of A, and λ_{ij} is the corresponding regularization parameter(s). Alternatively, constraints can also be used. See Section 2.3 and Section 3.4.

Different penalties have been proposed. The famous Lasso (Tibshirani, 1996) solves the ℓ_1 penalized problem. It is fast in computation. Nevertheless, Lasso suffers from some drawbacks such as selection inconsistency, estimation bias and incapability of dealing with collinearity, in particular. Zou and Hastie (2005) propose the elastic net (eNet for short in this paper) which adds an additional ridge regularization (Hoerl and Kennard, 1970) to deal with collinearity and large noise. However, the design counteracts sparsity to some extend and may bring the double shrinkage issue. Some nonconvex alternatives, including the '$\ell_0 + \ell_1$' SCAD (Fan and Li, 2001) and the '$\ell_0 + \ell_2$' hard-ridge (She, 2009, 2012), are advocated to promote more sparsity. However, due to nonconvexity, the convergent solution may be only locally optimal and depend on the choices of the initial point. They are also more computationally expensive than convex approaches. Therefore, we do not consider nonconvex regularizations hereinafter.

2.2 The S^2 Network Learning

Many real-world time series (possibly after proper transformations such as taking logarithm and/or differencing) are stationary. Stationary and nonstationary processes behave in fundamentally different manners. See Figure 2. For a stationary process, its probability distribution is invariant with respect to the shift in time. In the nonstationary process, however, we can clearly see drifting and trending behaviors. In practice, given raw observations sampled from a dynamical system, researchers first stationarize the time series and then input them to the ML/PML estimator. The resulting estimate $\hat{A}_{ML}/\hat{A}_{PML}$ is used for analysis and forecast. Unfortunately, however, the stationarity requirement may be violated by $\hat{A}_{ML}/\hat{A}_{PML}$ in practice. Figure 3 shows a real-data example. We apply ML and PML (adopting the ℓ_1 penalty) respectively to the U.S. macroeconomic data (Stock and Watson, 2012), which are stationary after proper transformations. The estimates are then used to forecast an index "GDP263". As shown in Figure 3, though the time series of GDP263

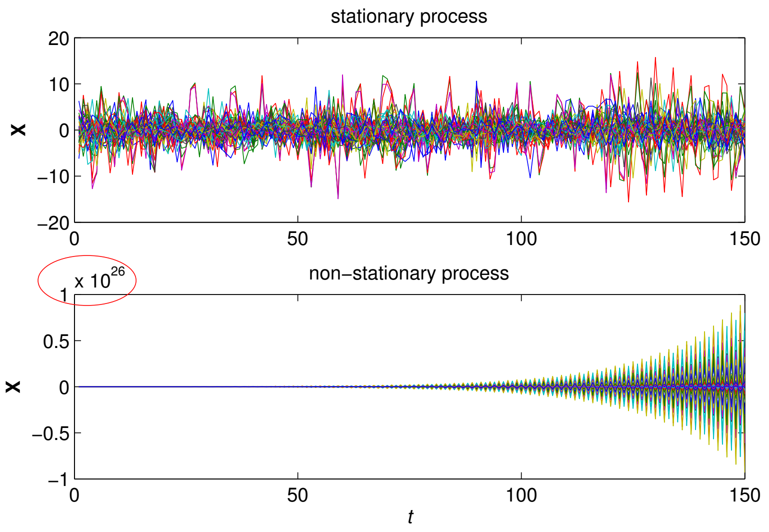


Figure 2: Example of stationary and nonstationary processes. The number of nodes is $p = 50$. The stationary process has $\rho(A) = 0.95$ and the nonstationary process has $\rho(A) = 1.05$.

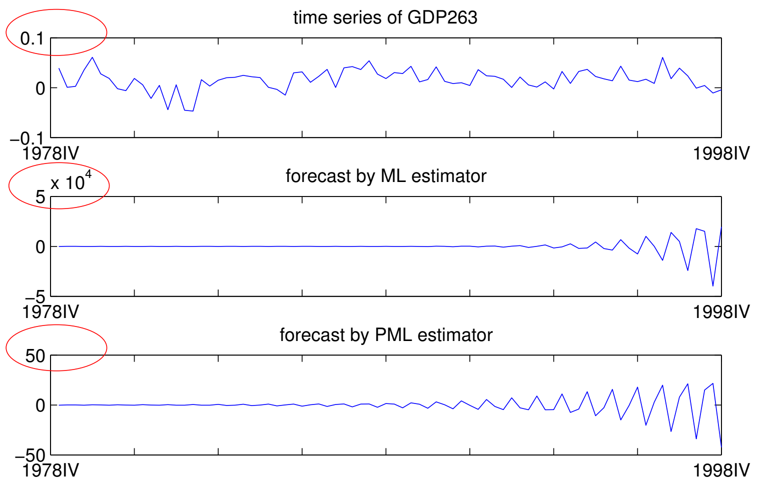


Figure 3: Forecasts of GDP263 given by ML and PML estimation. The time series of GDP263 is obtained from seasonal observations between 1978:IV and 1998:IV. It is a stationary process. However, the forecasts given by ML and PML exhibit nonstationary behaviors. $\rho(\hat{A}_{ML}) = 1.171$, $\rho(\hat{A}_{PML}) = 1.073$.

is stationary, the ML and PML forecasts clearly exhibit nonstationary behaviors and they fail to capture all characteristics of the original time series.

In this paper, we propose the framework of *stationary-sparse* ($\mathbf{S}^2$) network learning to address the limitation of PML to guarantee the stationarity property of the network. We invoke the stationarity condition and design an efficient algorithm to solve the optimization problem.

A random process as in (1) is stationary if and only if its spectral radius $\rho(A)$ satisfies the *stationarity condition*:

$$\rho(A) \overset{\Delta}{=} \max_i |\lambda_i| < 1, \tag{4}$$

where λ_i is the ith eigenvalue of A, possibly complex, and $|\cdot|$ is the complex norm (Reinsel, 1997). This leads to the following optimization problem:

$$\begin{aligned} \min_A \; f(A) = \frac{1}{2}\|Y - XA\|_F^2 + P(A;\lambda) \\ \text{s.t. } \rho(A) < 1. \end{aligned} \tag{5}$$

Nevertheless, problem (5) is extremely challenging due to the fact that $\rho(A)$ is a nonconvex and non-Lipschitz-continuous function of A. An optimization method proposed by Curtis and Overton (2012), which combines sequential quadratic programming and gradient sampling, sheds some light on solving (5). However, at each iteration, the gradient sampling needs to sample p^2 points and calculate the gradient of the spectral radius at each point. As discussed in Overton and Womersley (1988), calculating the gradient of spectral radius for a single point is already a challenging and computationally demanding problem. It is prohibitive to do so for p^2 points at each iteration in our problem. Moreover, this method only guarantees $\rho(A) \leq 1$, not $\rho(A) < 1$.

We consider a reasonable convex relaxation of (4) as the stationarity constraint:

$$\|A\|_2 \overset{\Delta}{=} \max_i |\nu_i| \leq 1,$$

where ν_i is the ith singular value of A and thus $\|A\|_2$ is the spectral norm. For an arbitrary square matrix, we have $\rho(A) \leq \|A\|_2$, where the equality holds when A is a symmetric matrix. In all our applications, we have $\rho(A) < 1$.

The $\mathbf{S}^2$ learning problem is given by

$$\begin{aligned} \hat{A}_{S^2} = \arg\min_A \; f(A) \\ \text{s.t. } \|A\|_2 \leq 1. \end{aligned} \tag{6}$$

Note that the stationarity constraint also has a "shrinking" effect on the estimate, which contributes to the shrinkage estimation we are seeking, as discussed in Section 2.1.

2.3 The "Berhu" Penalty for Sparsity Pursuit and Model Decorrelation

As discussed in Section 2.1, coherence is often observed in real-world network data, especially when some nodes have similar dynamical behaviors or strong influence between each other. In such cases, the conventional Lasso fails to handle collinearity and consequently gives unsatisfactory identification and forecasting performance. Hence, a more proper penalty is in need for $\mathbf{S}^2$ network learning. We adopt a new hybrid penalty "Berhu", which has a close relation with Huber's loss function for robust regression (Huber, 1981). The Huber function is quadratic at small values and linear at large ones, which makes it more robust to outliers than the squared-error criterion. Inspired by the Huber function, Owen (2007) designed a convex penalty function Berhu

$$P_{\mathcal{B}}(t;\lambda,M)=\begin{cases}\lambda|t| & \text{if } |t|\le M\\ \lambda\frac{t^2+M^2}{2M} & \text{if } |t|>M.\end{cases} \tag{7}$$

As implied by its name, Berhu reverses the composition of Huber: it is linear at small values and quadratic at large ones.

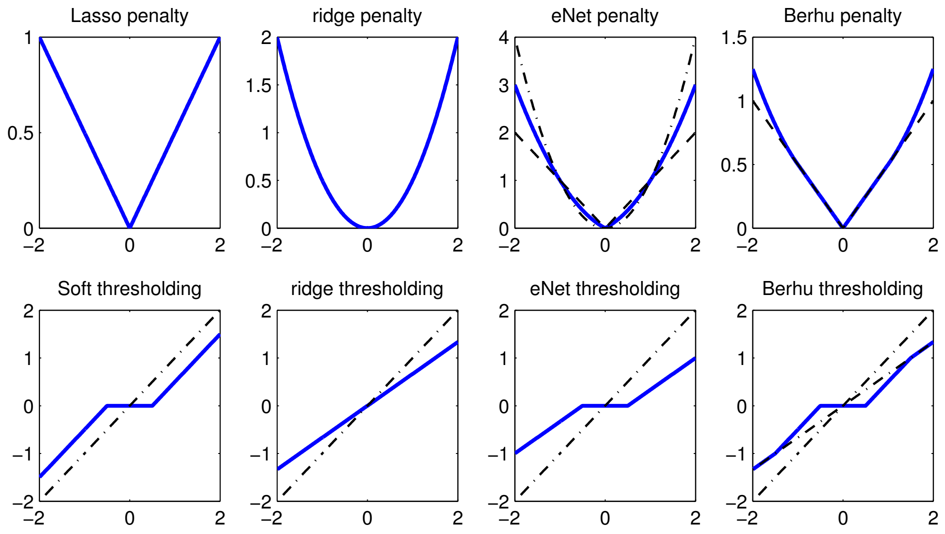


Figure 4: Penalty functions and corresponding solutions.

Figure 4 compares Berhu with the ridge penalty $P_R(t;\eta)=\frac{1}{2}\eta t^2$, Lasso $P_L(t;\lambda)=\lambda|t|$ and eNet $P_E(t;\lambda,\eta)=\lambda|t|+\frac{1}{2}\eta t^2$. The upper panel plots the functions, while the lower panel shows their corresponding solutions in the univariate and orthogonal case (see Section 3.2 for details). The ridge penalty shrinks the coefficients to compensate for collinearity. But it can not produce exact zero coefficients in the estimate. The Lasso soft-thresholds the coefficients to encourage sparsity. However, it does not shrink the large coefficients effectively and does not work well for correlated data. The eNet incorporates the ridge component into the ℓ_1. However, the singularity of the penalty function at zero is smoothed out to some extent by the ℓ_2 part, which may lead to an estimate not parsimonious enough. Also, it tends to over-shrink medium and large coefficients (Zou and Hastie, 2005). Berhu overcomes these drawbacks by using a nonlinear fusion of the ℓ_1 and ℓ_2 penalties: for small coefficients, the ℓ_1 regularization is enforced to achieve sparsity; for large coefficients, the ℓ_2 regularization is enforced to compensate collinearity (Hoerl and Kennard, 1970) and multi-dimensionality (James and Stein, 1961). As a result, Berhu not only preserves the singularity property of Lasso at zero but also inherits the advantage of ridge regression in model decorrelation. It is convex as well. The difference between Berhu and eNet is significant. For medium and large coefficients, eNet not only shifts but also shrinks, which results in the double shrinkage effect (Zou and Hastie, 2005). On the other hand, Berhu shifts only medium coefficients and shrinks large ones. It does selection and decorrelation separately, which better serves two important objectives of network learning: accurate prediction and parsimonious representation.

Substituting $P_{\mathcal{B}}(A;\lambda,M)$ into (6), we focus on solving

$$\begin{aligned} \arg\min_{A} \ f_{\mathcal{B}}(A) = \frac{1}{2}\|Y-XA\|_F^2 + P_{\mathcal{B}}(A;\lambda,M) \\ \text{s.t. } \|A\|_2 \le 1. \end{aligned} \tag{8}$$

In this problem, $P_{\mathcal{B}}(A;\lambda,M)$ is typically nondifferentiable at zero and piecewise. The stationarity constraint adds more difficulties to the problem. Moreover, in practice we are frequently confronted with large-scale networks. Hence, an efficient and scalable algorithm is desired for $\mathbf{S}^2$ network learning.

3. Computation of BISPS

In this section, we propose an algorithm named *Berhu iterative sparsity pursuit with stationarity* (BISPS) to effectively solve the $\mathbf{S}^2$ learning problem (8). Some algorithms based on conventional techniques will be developed first. They suffer from high computational complexity, poor numerical accuracy, and/or insufficient sparsity. We then propose the novel BISPS which is easy to implement and computationally efficient. Finally, to facilitate BISPS for ultra-high dimensional problems, we propose the *quantile thresholding iterative screening* (QTIS). Convergence proofs are provided. We assume the data matrices X,Y has been centered before all the computation. Precalculations $\Sigma_{XX} \triangleq X^\top X$ and $\Sigma_{XY} \triangleq X^\top Y$ help avoid repeated computation.

3.1 Algorithms Based on Conventional Techniques

Problem (8) can be reformulated and then solved by well-known optimization techniques, such as semidefinite programming, projected subgradient method, and alternating direction method of multipliers. We briefly discuss these algorithms before introducing BISPS.

3.1.1 SEMIDEFINITE PROGRAMMING

Problem (8) can be reformulated as a semidefinite programming (SDP) problem

$$\begin{aligned} \min_{A} f_{\mathcal{B}}(A) \\ \text{s.t. } \begin{bmatrix} I & A \\ A^\top & I \end{bmatrix} \succeq 0 \end{aligned}$$

and can be solved by general SDP solvers. However, since most of the SDP solvers use interior point methods, they suffer from extremely high space complexity. For example, we tried the popular SDP solvers SeDuMi (Sturm, 1998) and SDPT3 (Tütüncü et al., 2003) using MATLAB7.11.0 on a PC with 4GB memory; when the number of network nodes is larger than 100, both solvers ran out of memory.

3.1.2 PROJECTED SUBGRADIENT METHOD

Define the subgradient of $f_{\mathcal{B}}(A)$ at A as

$$\partial f_{\mathcal{B}}(A) = \nabla l(A) + \partial P_{\mathcal{B}}(A;\lambda,M),$$

where $\partial P_{\mathcal{B}}(A;\lambda,M)$ is the subdifferential (Alber et al., 1998) of $P_{\mathcal{B}}(\cdot)$, and $\nabla l(A)$ is the gradient of $l(A)$: $\nabla l(A) = \Sigma_{XX}A - \Sigma_{XY}$. The projected subgradient method (PSGM) for problem (8) computes a sequence of feasible points $\{A^k\}$ with the update rule

$$A^{k+1} = \Pi(A^k - \alpha_k U^k; 1), \text{ where } U^k \in \partial f_{\mathcal{B}}(A^k),$$

until A^k satisfies $0 \in \partial f_{\mathcal{B}}(A^k)$. The operator Π stands for spectral norm projection defined in Lemma 5.

PSGM is simple to implement. At each step, one performs subgradient evaluation and spectral norm projection. Nevertheless, due to the uncertainty of the subgradient at the non-differentiable point of the penalty function, it suffers from slow convergence and insufficiency of sparsity. For example, in an experiment where the number of nodes $p = 100$ and number of observations $n = 80$, it does not converge yet after 10^4 iterations.

3.1.3 ALTERNATING DIRECTION METHOD OF MULTIPLIERS

The basic idea of alternating direction method of multipliers (ADMM) is to split the objective function and variables and update them in an alternating fashion (Boyd et al., 2010). To apply ADMM for solving problem (8), we reformulate it as

$$\begin{gathered}\min_{A,B,C} \frac{1}{2}\|Y - XA\|_F^2 + P_{\mathcal{B}}(B;\lambda,M) \\ \text{s.t. } \begin{bmatrix} A \\ A \end{bmatrix} = \begin{bmatrix} B \\ C \end{bmatrix}, \\ \text{and } \|C\|_2 \le 1.\end{gathered}$$

The augmented Lagrangian can then be written as $L_{\rho_1,\rho_2}(A,B,C,\Gamma_1,\Gamma_2) = \frac{1}{2}\|Y - XA\|_F^2 + P_{\mathcal{B}}(B;\lambda,M) + \mathrm{tr}\{\Gamma_1^T(A-B)\} + \mathrm{tr}\{\Gamma_2^T(A-C)\} + \frac{\rho_1}{2}\|A-B\|_F^2 + \frac{\rho_2}{2}\|A-C\|_F^2$, where Γ_1 and Γ_2 are Lagrangian multipliers and ρ_1 and ρ_2 are the augmented Lagrangian parameters. The iteration of ADMM consists of the following steps

$$\begin{aligned}
A^{k+1} &= (\Sigma_{XX} + \rho_1 I + \rho_2 I)^{-1}(\Sigma_{XY} - \Gamma_1^k - \Gamma_2^k + \rho_1 B^k + \rho_2 C^k), \\
B^{k+1} &= \Theta_{\mathcal{B}}(A^{k+1} + \Gamma_1^k/\rho_1; \lambda/\rho_1, M), \\
C^{k+1} &= \Pi(A^{k+1} + \Gamma_2^k/\rho_2; 1), \\
\Gamma_1^{k+1} &= \Gamma_1^k + \rho_1(A^{k+1} - B^{k+1}), \\
\Gamma_2^{k+1} &= \Gamma_2^k + \rho_2(A^{k+1} - C^{k+1}),
\end{aligned}$$

where $\Theta_{\mathcal{B}}$ is the thresholding operator of Berhu—see Appendix A for detail. Note that matrix inversion is involved in updating A, which increases computational difficulty. The penalty parameters ρ_1 and ρ_2 have to be large enough; the choices of them have been shown to influence the number of iterations significantly. To speed up convergence in practice, one can replace the constants ρ_1 and ρ_2 with two sequences $\{\rho_1^k\}$ and $\{\rho_2^k\}$ respectively, where ρ_1^k and ρ_2^k vary along the iterations following some *ad hoc* adaptive rule (He et al., 2000). However, the algorithm is still slow when the problem is high dimensional. For example, when $p = 300, n = 100$, ADMM costs up to 10 times more computation time than BISPS (to be described in Section 3.2) to reach comparable accuracy. Also, the convergence property of ADMM with varying ρ is not clear.

In summary, although we have implemented some algorithms based on conventional techniques for the $\mathbf{S}^2$ network learning problem, they are unable to cope with large-scale networks. A more efficient and scalable algorithm is in great need.

3.2 The Berhu Thresholding Operator

Section 2.3 advocates the Berhu penalty for $\mathbf{S}^2$ network learning. However, in the original paper (Owen, 2007), Berhu was solved through cvx (Grant and Boyd, 2008). Practical applications call for the development of much faster algorithms. In this paper, we reparameterize Berhu and develop its coupled *thresholding rule*, which allows us to solve the Berhu sparsity pursuit—problem (3) with Berhu penalty—in a simple and efficient way. This formulation facilitates easy parameter tuning. Also, it helps us understand the essence of Berhu.

Let $\eta = \lambda/M$. Reformulate the Berhu penalty (7) as

$$P_{\mathcal{B}}(t;\lambda,\eta) = \begin{cases} \lambda|t| & \text{if } |t| \le \lambda/\eta \\ \frac{\eta^2 t^2+\lambda^2}{2\eta} & \text{if } |t| > \lambda/\eta. \end{cases} \tag{9}$$

Define a thresholding rule

$$\Theta_{\mathcal{B}}(t;\lambda,\eta) = \begin{cases} 0 & \text{if } |t| < \lambda \\ t-\lambda\mathrm{sgn}(t) & \text{if } \lambda \le |t| \le \lambda+\lambda/\eta \\ \frac{t}{1+\eta} & \text{if } |t| > \lambda+\lambda/\eta. \end{cases} \tag{10}$$

It can be verified that, as shown in Lemma 3, $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is the coupled thresholding rule for the Berhu penalty $P_{\mathcal{B}}(\cdot;\lambda,\eta)$:

$$P_{\mathcal{B}}(t;\lambda,\eta) = \int_0^{|t|} (\sup\{s : \Theta_{\mathcal{B}}(s;\lambda,\eta) \le u\} - u)du.$$

For the multivariate case, the Berhu thresholding operator is applied elementwise.

With $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$, we can solve the Berhu sparsity pursuit (without the stationarity constraint) using a simple iterative procedure:

$$A^{k+1} = \Theta_{\mathcal{B}}(A^k + \alpha_k(\Sigma_{XY} - \Sigma_{XX}A^k);\lambda,\eta). \tag{11}$$

Since $P_{\mathcal{B}}(\cdot;\lambda,\eta)$ is convex, algorithm convergence is guaranteed given $\alpha_k \le \sqrt{2}/\|X\|_2$ (She, 2012).

It is worth pointing out that, based on the construction rule (13), we can also define the thresholding operators for other penalties including Lasso, eNet and the ridge penalty, as shown in Figure 4. The Berhu thresholding operator $\Theta_{\mathcal{B}}(t;\lambda,\eta)$ offers a nonlinear fusion of the soft thresholding operator (coupled with Lasso) $\Theta_S(t;\lambda) = \mathrm{sgn}(t)(|t|-\lambda)1_{|t|\ge\lambda}$ and the ridge thresholding operator $\Theta_R(t;\eta) = \frac{t}{1+\eta}$. For the difference between the Berhu thresholding and the eNet thresholding $\Theta_E(t;\lambda,\eta) = \frac{1}{1+\eta}\mathrm{sgn}(t)(|t|-\lambda)1_{|t|\ge\lambda}$, see the discussion in Section 2.3.

3.3 The BISPS Algorithm

Based on the Berhu thresholding operator (10), we now propose BISPS as given in Algorithm 1. This algorithm contains only simple matrix operations in addition to the *spectral norm projection*

Algorithm 1 The Berhu iterative sparsity pursuit with stationarity (BISPS)

```
Input: data matrix $\Sigma_{XX}, \Sigma_{XY}$; regularization parameters $\lambda$, $\eta$; stopping criteria $\delta_1, \delta_2, M_1, M_2$; initial
  estimate $A^0$.
  {Let $\|\cdot\|_2$ denotes the spectral norm and $\|\cdot\|_{\max}$ denote the elementwise max-norm.}
  $k_0 \leftarrow$ any constant satisfying $k_0 > \|X\|_2$;
  $k \leftarrow 0$;
  repeat
    1) $B^0 \leftarrow A^k + \frac{1}{k_0^2}(\Sigma_{XY} - \Sigma_{XX}A^k)$;
    2) $j \leftarrow 0; P^0 \leftarrow 0; Q^0 \leftarrow 0$;
    repeat
      2.1) $C^j = \Theta_{\mathcal{B}}(B^j + P^j; \lambda/k_0^2, \eta/k_0^2)$;
      2.2) $P^{j+1} = B^j + P^j - C^j$;
      2.3) $B^{j+1} = \Pi(C^j + Q^j; 1)$;
      2.4) $Q^{j+1} = C^j + Q^j - B^{j+1}$;
      $j \leftarrow j+1$;
    until $\|B^j - B^{j-1}\|_{\max} \le \delta_2$ or $j \ge M_2$
    3) $A^{k+1} \leftarrow B^j$;
    $k \leftarrow k+1$;
  until $\|A^k - A^{k-1}\|_{\max} \le \delta_1$ or $k \ge M_1$
  $\hat{A} \leftarrow A^k$;
Output: $\hat{A}$.
```

Π. Parameter k_0 can be set to any constant that is larger than the spectral norm of X. No *ad hoc* algorithmic parameters, such as ρ_1, ρ_2 in ADMM and α_k in PSGM, are involved. The inner iteration of Step 2 often converges within 10 steps in practice, where matrices C, P, Q are auxiliary variables that contribute to fast convergence of the procedure. Step 2.1 is to enforce sparsity by Berhu thresholding and Step 2.3 is to project the estimate to the convex set $\{B : \|B\|_2 \le 1\}$. The outer iteration has only a simple update step and converges fast. As a result, the algorithm is computationally efficient as well as easy to implement.

The convergence of BISPS is theoretically guaranteed. For simplicity, we assume the inner iteration is run till convergence. Theorem 1 states that Algorithm 1 solves the $\mathbf{S}^2$ network learning problem.

Theorem 1 *Suppose* $\lambda \ge 0, \eta \ge 0$ *and* $\lambda\eta \ne 0$. *Given* $k_0 > \|X\|_2$, *for any initial value* A^0, *the sequence of iterates* $\{A^k\}$ *produced by Algorithm 1 converges to a* ***globally*** *optimal solution to problem* (8).

See Appendix A for the detailed proof.

BISPS has more flexibility and generality. Though it is designed with the Berhu penalty, by replacing $\Theta_{\mathcal{B}}$ with an appropriate thresholding operator in Step 2.1, the algorithm allows for any convex penalty for $\mathbf{S}^2$ learning. Moreover, if Step 2.2 to Step 2.4 are removed, Algorithm 1 reduces to the Berhu sparsity pursuit (11).

The most expensive computation of Algorithm 1 lies in the spectral norm projection (Step 2.3). In practice, we can apply some techniques to further improve computational efficiency. 1) We can first run Algorithm 1 without Step 2.2 to Step 2.4 and obtain an estimate. If it satisfies the stationarity condition (4), we accept and output this solution. Otherwise, we rerun Algorithm 1 with Step 2.2 to Step 2.4 included. 2) Moreover, we can take advantage of the fact that A^{k+1} is sparse and the number of singular values of A^{k+1} that are larger than 1 is much smaller than p. The singular value thresholding algorithm (Cai et al., 2010), among some other fast algorithms, calculates only the singular values that are above a threshold and their corresponding singular vectors, which is computationally efficient for large sparse matrix and thus fits our problem well. We use the package MODIFIED-PROPACK provided by Lin (2011) to calculate the partial SVD with threshold being 1. For $p = 500, n = 100$, the partial SVD, compared with the original SVD, can accelerate the calculation by up to 30 times.

3.4 Quantile Thresholding Iterative Screening

Nowadays, a great challenge for network identification and statistical learning comes from the large scale of the system. For example, for a network with $p = 1000$ nodes, the number of variables in the transition matrix is as large as $p^2 = 10^6$, which poses a great challenge for any estimation algorithm in scalability and stability. As a result, ultra-high dimensional learning has become a hot topic (Fan et al., 2009; Fan and Lv, 2010) . For regression problems, under the assumption that the number of nonzero coefficients is far smaller than n, *screening* techniques can be used to coarsely select the variables before finer estimation. This idea can be adopted in network identification: if one is sure that the average number of connections for each node is much less than $\lceil \mu n \rceil$ (say $\mu = 0.8$) or the total number of connections in the network is much less than $\lceil \mu pn \rceil$, one can first use fast screening techniques to select $m = \lceil \mu pn \rceil$ candidate connections, and then apply BISPS restricted on the candidate connections for further selection and estimation. If the screening technique can include all the true connections with high probability, dramatic computational gain can be attained with mild performance sacrifice.

Independence screening methods, such as the sure independence screening (SIS) (Fan and Lv, 2008) can be applied to preselect variables in a supervised manner. Applied to network learning, SIS sorts the elements of $W = X^\mathsf{T} Y$ by magnitude in a decreasing order and defines a reduced model

$$\mathcal{M}_\mu = \{(i, j) : |w_{ij}| \text{ is among the } m \text{ largest of all}, 1 \le i, j \le p\}.$$

This method is simple and fast, but it relies on the assumption that the predictors are *independent*, since it only studies the marginal correlation between Y and X and selects the variables accordingly. In network settings, the nodes are interacting dynamically with each other, so there is usually high collinearity in the data. In such cases, SIS is too greedy and misses many true connections.

To derive a new screening technique that can handle network data, we first observe that SIS corresponds to the first step of the iterative procedure in (11) with $A^0 = 0$ and hard thresholding $\Theta_H(t; \lambda) = t1_{|t| \ge \lambda}$ with a properly chosen λ. This inspires us to apply an iterative procedure for screening: starting from $A^0 = 0$, repeat

1) $A^{k+1} \leftarrow A^k - \frac{1}{k_0^2}(\Sigma_{XX} A^k - \Sigma_{XY})$;
2) $\lambda^{k+1} \leftarrow (m+1)$th largest element of A^{k+1} in magnitude;
3) $A^{k+1} \leftarrow \Theta_H(A^{k+1}; \lambda^{k+1})$;
4) $\mathcal{M}_\mu^{k+1} \leftarrow \{(i, j) : |a_{ij}^{k+1}| \ne 0, 1 \le i, j \le p\}$;

until $\mathcal{M}_\mu^k$ stops changing.

We call this screening procedure the *quantile thresholding iterative screening* (QTIS). As shown in Step 2 and Step 3, QTIS does not select variables using a fixed thresholding parameter λ. Instead, it uses a **dynamic** threshold to keep a fixed number m of nonzero elements at each iteration. The *quantile parameter* μ determines the number of variables to be selected. In comparison to SIS which ranks the connections based on $X^{\top}Y$, the iterative nature of QTIS lessens the greediness by repeatedly updating the importance of each candidate connection with a theoretical guarantee of convergence.

Theorem 2 *Given $k_0 > \|X\|_2$, for any $0 < \mu \leq 1$, the sequence of iterates $\{A^k\}$ generated by QTIS has the function value decreasing property that $l(A^{k+1}) \leq l(A^k)$, where $l(A) = \frac{1}{2}\|Y - XA\|_F^2$, and A^k satisfies $\|A^k\|_0 \leq m$, where $\|\cdot\|_0$ denotes the number of nonzero elements.*

See Appendix B for the detailed proof.

In practice, $\mathcal{M}_\mu$ usually stops changing after less than a hundred iterations. The number of unknowns is reduced from p^2 to $\lceil \mu pn \rceil$ effectively by a small amount of computation. Then, more involved and sophisticated estimation, for example, BISPS, can be performed to the reduced model. It is much faster than applying BISPS directly if $p^2 \gg n$. In addition, QTIS provides BISPS with a sparse pattern, which facilitates the fast computation of partial SVD.

To apply BISPS on $\mathcal{M}_\mu$, we use element-wise penalty parameters λ_{ij}'s and set

$$\lambda_{ij} = \infty \text{ if } (i,j) \notin \mathcal{M}_\mu.$$

This simple modification guarantees that only elements in $\mathcal{M}_\mu$ will be selected by BISPS.

3.5 Two-dimensional Selective Cross Validation for Tuning

The reparameterization of Berhu (9) separates the roles of ℓ_1 and ℓ_2 regularizations; each of them is associated with a regularization parameter, namely λ for ℓ_1 and η for ℓ_2. This provides important guidelines for parameter tuning. Based on our experience, the estimate is not very sensitive to η, so a full two-dimensional grid search is not necessary. Instead, we search along several one-dimensional solution paths including one η-path and three λ-paths:

- *Step 1*: Run the η-path ($\lambda = 0$). Do ridge regression with a grid of values for η, and choose the optimal η^* using AIC (Akaike, 1974).

- *Step 2*: Run 3 λ-paths with $\eta = 0.5\eta^*, 0.05\eta^*, 0.005\eta^*$ respectively. For each value of η, run BISPS with a grid of values for λ, and find the optimal one λ^o using the K-fold selective cross-validation (SCV) (She, 2012). This results in three λ^o's, one from each path. Choose the optimal one from them and let it be the optimal thresholding parameter λ^*. The pair (λ^*, η^*) is our final choice of the two parameters.

The SCV cross-validates different sparsity patterns instead of the regularization parameters. It is more computationally efficient than the plain cross validation since it runs the sparse algorithm only once and globally (instead of K times locally). To calculate the SCV error associated with λ, we first apply BISPS to the whole data set and obtain the solution $\hat{A}(\lambda)$. Then, for $k = 1, \cdots, K$, we apply ridge regression restricted to the variables that are picked by $nz(\hat{A}(\lambda))$ on the data without the kth data piece, and evaluate its validation error using the kth data piece. The sum of the K validation errors is defined as the SCV error. See She (2012) for more details.

4. Stationary Bootstrap (SB) Enhanced Network Learning

The $\mathbf{S}^2$ learning framework proposed in Section 3 is an effective technique to identify stationary and sparse network. Nevertheless, a "one-time" estimate, without any p-value or confidence interval, provides only limited guidance in identifying the true network topology. In fact, whatever inference method is used, there will be uncertainty underlying the variable selection procedure. It would be greatly helpful if one could provide some kind of uncertainty measure for such an estimate. In our case, we would like to find a certain confidence measure for the estimated topology. This can be done by assigning a probability for the existence of each connection. Hence, we use bootstrap (Efron, 1979). In this section, we propose the *stationary bootstrap* enhanced BISPS (SB-BISPS) which provides a confidence level about whether a connection exists in the network by measuring the frequency with which it is chosen by the BISPS algorithm.

4.1 The SB-BISPS Framework

The SB-BISPS procedure completes the BISPS (or QTIS+BISPS) algorithm with a stationary bootstrap resampling step. The SB-BISPS is described as follows.

- *Step 1*: Run BISPS over the original data set $\mathcal{X}$. Record the pattern of $\hat{A}$, which is a $p \times p$ binary matrix $\Phi = [\phi_{ij}]_{1\le i,j\le p}$ defined as:

$$\phi_{ij} = \begin{cases} 1 & \text{if } \hat{a}_{ij} \neq 0 \\ 0 & \text{if } \hat{a}_{ij} = 0. \end{cases}$$

- *Step 2*: Draw B **stationary** bootstrap samples from $\mathcal{X}$. Repeat Step 1 for each sample. Record Φ_j^* for the jth sample.

- *Step 3*: Compute the matrix $F = [f_{ij}]_{1\le i,j\le p}$ of *connection occurring frequency* (COF) by adding up all the patterns Φ_j^*'s and normalizing the result by B:

$$F = \frac{1}{B}\sum_{j=1}^{B} \Phi_j^*.$$

Given a sufficiently large B, f_{ij} is a good approximation of the probability for BISPS to select connection a_{ij}, which serves as a measure of how confident we are with the existence of this connection. For example, if $f_{ij} = 80\%$, it means that in 80% of the bootstrap samples, a connection from node i to node j is detected. So we can say the probability for the existence of this connection is (approximately) 80%. We can use a cutoff value f^* to threshold the COF matrix, and choose only the connections with $f_{ij} \ge f^*$ for further analysis. This renders us a sparse topology that shows the most significant connections within the network.

4.2 Stationary Bootstrap

There are different resampling schemes to draw bootstrap samples from the original data in Step 2. If the observations are independent and identically distributed, we can resample the data randomly with replacement. When the observations are time series, the problem is more complicated, since the observations are largely dependent on each other, and we would like to preserve the specific dependency structure in bootstrapping. Techniques such as resampling blocks of consecutive observations or resampling "blocks of blocks" can be used (Kunsch, 1989). The basic idea is that, despite the dependence of individual observations, blocks of observations can be approximately independent with each other given a proper block size l.

When a time series is stationary, it is natural to maintain this property in the bootstrap samples. The stationary bootstrap (Politis and Romano, 1994) is a method with this property. It is based on resampling blocks of random lengths, where the length of each block follows a geometric distribution with mean $1/\gamma$. We apply a simple approach to conduct such resampling. Given that x_i^* is chosen to be the Jth observation x_J in the original time series, we choose x_{i+1}^* based on the following rule:

$$x_{i+1}^* = \begin{cases} x_{J+1} \text{ with probability } 1-\gamma \\ \text{picked randomly from } x_1, \cdots, x_n \text{ with probability } \gamma. \end{cases}$$

Similar with block bootstrap, where the block size l has to be determined, the value of γ should be chosen properly. Fortunately, the sensitivity of γ in stationary bootstrap is less than that of l in block bootstrap.

5. Experiments

In this section, we present the experimental results on synthetic data and demonstrate the effectiveness of the proposed $\mathbf{S}^2$ network learning framework.

5.1 Performance Measures and Experiment Settings

To examine the performance of the proposed methods, we define the following measures.

- Stationarity violation percentage (P_v): In N repeated experiments, if there are N_v experiments in which the estimate $\hat{A}$ violates the stationarity condition (4), then the stationarity violation percentage is defined as $P_v = N_v/N$.
- Miss rate (P_m): If $a_{ij} \neq 0, \hat{a}_{ij} = 0$, we say there is a miss. Denote C_m as the total number of misses and C_{nz} as the number of nonzero entries in A. The miss rate is defined as $P_m = C_m/C_{nz}$.
- False alarm rate (P_f): If $a_{ij} = 0, \hat{a}_{ij} \neq 0$, we say there is a false alarm. Denote C_f as the total number of false alarms and C_z as the number of zero entries in A. The false alarm rate is defined as $P_f = C_f/C_z$.
- Testing error (TE): The testing error is defined as $TE = \frac{1}{n_t}\|Y^t - X^t\hat{A}\|_F^2$, where Y^t and X^t are testing data, and n_t is their length. For time series, the testing data are collected right after the training data.
- Computation time: The averaged running time of an algorithm. All the algorithms are run in MATLAB7.11.0 on a PC with 4GB memory.

Particularly, to evaluate the prediction/forecasting performances of the algorithms, we adopt the so-called rolling MSE, a conventional measure in econometrics (Stock and Watson, 2012). Suppose we have T observations: $x_1, \cdots, x_T$. Let the rolling window size be W and the horizon be h. Standing at time t, we use the most recent W observations to estimate A, denoting the estimate as $\hat{A}_t$. Then we use this estimate to forecast x_{t+h}, denoting the forecast as $\hat{x}_{t+h}$ and the forecasting error as $e_t^h = \|x_{t+h} - \hat{x}_{t+h}\|_2^2$. This process is repeated for $t = W, \cdots, W+N-1$ as we shift the window. N is the number of window shifting that satisfies $1 \le N \le T - W - h - 1$. Then the rolling MSE for horizon h is defined as $MSE_{rolling}^h = \frac{1}{N}\sum_{t=W}^{W+N-1} e_t^h$. When using $\hat{A}_t$ to forecast x_{t+h}, we should do pseudo out-of-sample forecasting. That is, we assume the observations after t are not available and consequently we need do h-step-ahead forecast. For our model (1), this should be done as: $\hat{x}_{t+h} = \hat{A}_t^{\mathsf{T}} \hat{x}_{t+h-1}$ for $h \ge 1$, where $\hat{x}_t \overset{\Delta}{=} x_t$ when $h = 1$. The testing error defined above corresponds to the rolling MSE for $h = 1$.

We generate the $p \times p$ transition matrix A with both sparsity and stationarity properties. First, the topology is generated from a directed random graph $G(p, \xi)$, where the edge from one node to another node occurs independently with probability ξ. Then the strength of the edges is generated independently from a Gaussian distribution. This process is repeated until we obtain a matrix A that has a desired spectral radius $0.9 < \rho(A) < 1$. We set $\xi = 10/p, \Sigma_\varepsilon = \sigma^2 I, \sigma^2 = 10$.

The regularization parameters are chosen by SCV as described in Section 3.5. For a λ-path, we use a grid of 100 values for λ, which is picked from the interval $[0, \|A^0 + X^{\mathsf{T}}Y - X^{\mathsf{T}}XA^0\|_{\max}]$. The initial estimate is simply set as $A^0 = 0$. For an η-path, we use a grid of 76 values for η, which is picked from the interval $[2^{-10}, 2^5]$. The number of folds for SCV is set to be $K = 5$. All the statistics collected are averaged over $N = 100$ times of window shifting. The length of testing data $n_t = 200$.

5.2 Performance of BISPS

We compare the performance of BISPS with Lasso, eNet and Berhu—we use the penalty name to denote the corresponding PML estimation. The number of observations is $n = 80$. Table 1 shows the experiment results for different network sizes, namely $p = 100, 200, 300$. Recall that for a network with size p, the number of unknown parameters is p^2. For example, for the network with 300 nodes, the number of parameters to be estimated is 9×10^4, which is extremely high compared with the number of observations 80.

Among the three penalties, the Lasso solution gives higher miss rates. This is because when some predictors are correlated, Lasso tends to choose only a part, or even none, of them. As a result, Lasso sometimes “over-shrinks” the estimate. The eNet and Berhu, in such cases, tend to include all the correlated predictors, thanks to the ℓ_2 part in the penalties. However, the sparsity of the eNet solution is affected by the ℓ_2 regularization, so it gives high false alarm rates. On the other hand, Berhu has improved eNet to some extend by enforcing the ℓ_2 regularization only to large coefficients. As a result, Berhu achieves the smallest testing errors ($h = 1$) among the three penalties.

As shown by P_v, no matter what penalty is used, it is possible for PML to give a nonstationary estimate, whereas the proposed $\mathbf{S}^2$ learning and BISPS can guarantee the stationarity property of $\hat{A}$. This indicates that adding the stationarity constraint into the sparsity pursuit does effectively prevent the estimate from becoming nonstationary. Meanwhile, the $\mathbf{S}^2$ estimate can achieve a comparable estimation and detection accuracy with the PML estimate. Table 1 gives the rolling MSEs for different horizons h to illustrate both the short term and long term forecasting performance. For PML estimates, the rolling MSEs grow *explosively* with h due to the existence of nonstationary estimates, while those of BISPS accumulate much more slowly.

To further illustrate the disadvantages of a nonstationary estimate, we find one run where Lasso gives a nonstationary estimate $\hat{A}_{Lasso}$. Starting from a time point t, we generate $\hat{x}_{t+h}(\hat{A})$

p	method	P_v	$(P_m, P_f)(\%)$	$h=1$	$h=64$	$h=128$
100	Lasso	5%	(13.6, 22.3)	23.4	1058.3	7798.9
	eNet	5%	(12.3, 25.9)	23.5	1238.1	10842.8
	Berhu	5%	(<u>12.3</u>, <u>24.9</u>)	<u>23.0</u>	1329.4	12219.6
	BISPS	**0%**	(<u>12.4</u>, <u>24.8</u>)	<u>23.1</u>	195.9	**209.2**
200	Lasso	3%	(15.8, 29.3)	41.8	332.8	10998.7
	eNet	2%	(14.5, 26.9)	36.5	207.8	405.3
	Berhu	2%	(<u>14.2</u>, <u>24.6</u>)	<u>32.2</u>	201.4	391.5
	BISPS	**0%**	(<u>14.2</u>, <u>24.4</u>)	<u>32.1</u>	131.9	**200.5**
300	Lasso	2%	(18.1, 14.9)	19.9	61.6	111.2
	eNet	3%	(13.5, 26.1)	20.4	65.5	123.2
	Berhu	3%	(<u>13.4</u>, <u>22.9</u>)	<u>18.8</u>	66.5	121.3
	BISPS	**0%**	(<u>13.7</u>, <u>22.1</u>)	<u>19.3</u>	63.3	**65.5**

Table 1: Performance comparison of BISPS with Lasso, eNet and Berhu. $n = 80$.

for $h = 1, \cdots, 100$ using $\hat{A}_{Lasso}$ and $\hat{A}_{BISPS}$ respectively and compare them with x_{t+h} observed from the true model. The results are plotted in Figure 5. We can easily see that $\hat{x}(\hat{A}_{BISPS})$ gives a reasonable imitation of the true system. The nonstationary estimate $\hat{x}(\hat{A}_{Lasso})$, however, *blows up* quickly and behaves completely differently from the true model. This indicates that ensuring a stationary estimate is indeed crucial.

5.3 Performance of QTIS

To examine the performance of QTIS for connection screening, we first compare it with the sure independence screening (SIS) (Fan and Lv, 2008) by examining their ability to include all the true connections, which can be measured by the miss rate. Simulation is done for networks with different sizes, namely $p = 300, 400, 500$. The sample size $n = 80$. Independence screening methods, including SIS, are very popular in ultra-high dimensional problems for dimension reduction and variable selection. However, our finding is that such methods can perform very poorly for network learning. As shown in Figure 6, it is possible for SIS to miss even more than half of the true connections. One possible reason is that, because of the evolving processes, correlation exits ubiquitously in dynamical networks. As a result, independence screening is not appropriate for network learning. On the other hand, the proposed QTIS algorithm considers the correlation issue and thus can obtain much smaller miss rates than SIS. Also, its performance is more robust to the choice of the quantile parameter μ.

We then run BISPS with and without QTIS and check the difference of their performances. Denote "QTIS+BISPS" as the procedure that first applies QTIS to screen the connections and then applies BISPS to the reduced model. Networks with sizes up to $p = 800$ are considered (the number of unknown parameters is $p^2 = 640,000$). The sample size $n = 80$. The quantile parameter $\mu = 0.8$. Figure 7 compares the performance of QTIS+BISPS and BISPS. When p/n ratio is large, adding QTIS not only improves the estimation and identification accuracy, but also saves up to **80%** of the computation time. As the p/n ratio becomes larger, the improvement becomes more remarkable. Therefore, QTIS is a helpful tool to facilitate BISPS for ultra-high dimensional network learning.

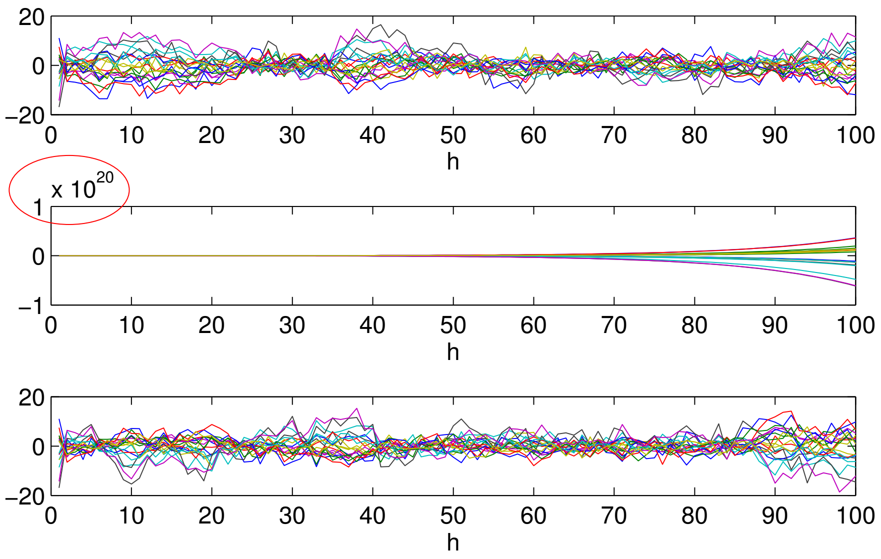


Figure 5: Comparison of BISPS and Lasso in terms of forecasting performance. Top: sample from the true model; Middle: forecast from the nonstationary estimate $\hat{A}_{Lasso}$; Bottom: forecast from the stationary estimate $\hat{A}_{BISPS}$.

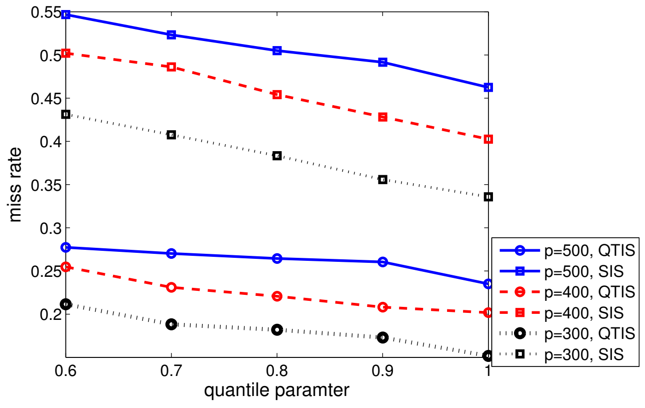
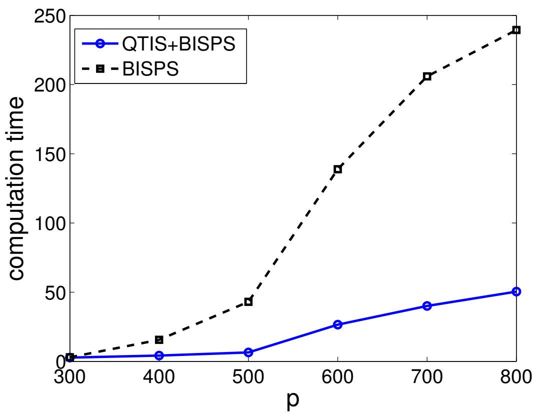


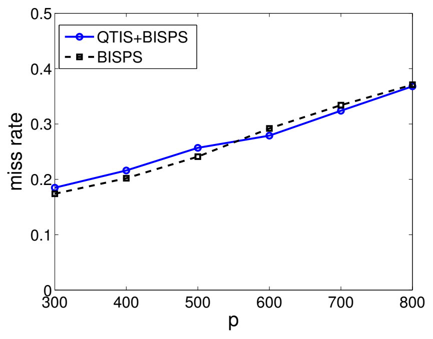
Figure 6: Miss rate comparison for QTIS and SIS. $n = 80$.

6. Application to U.S. Macroeconomic Data

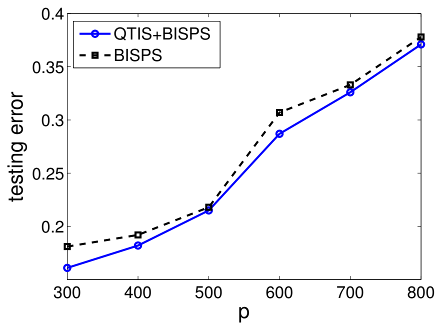
We apply the proposed learning framework to the U.S. macroeconomic data. The data set consists of quarterly observations on 108 macroeconomic variables from 1960:I to 2008:IV, which belong to 12 categories. A complete description of the data can be found at "`http://www.princeton.edu/~mwatson/wp.html`". There are in total 109 macroeconomic variables from 13 categories. We remove Category 13 (consumer expect) since it has only one variable while we are interested



(a) computation time



(b) miss rate



(c) testing error

Figure 7: Performance of QTIS+BISPS, compared with applying BISPS to a full model. $n = 80$.

Category	Lasso	**BISPS**	Category	Lasso	**BISPS**
1. GDP	0.589	0.445	7. Prices	1.971	1.874
2. IP	0.846	0.576	8. Wages	0.552	0.207
3. Employment	0.936	0.711	9. Interest rate	1.443	0.738
4. Unempl. rate	0.289	0.165	10. Money	0.114	0.065
5. Housing	0.071	0.033	11. Exchange rates	0.370	0.107
6. Inventories	0.506	0.217	12. Stock prices	0.254	0.100

Table 2: Normalized Rolling MSE of Lasso and BISPS for each category.

in multivariate time series. The data have been preprocessed so that each time series is a stationary process. We use the $\mathbf{S}^2$ network model (1) to detect Granger causal relations between these macroeconomic indices.

h	1	2	4	8	16	32
Lasso	0.017	0.021	0.029	0.365	329.9	3.1×10^8
BISPS	0.017	0.018	0.019	0.020	**0.020**	**0.018**

Table 3: Rolling MSE of Lasso and BISPS for different horizons.

6.1 Comparison of Rolling MSE

We first study the data set by category, considering that multiple time series explain the interactions of the indices in each category. To each of the 12 categories, we apply Lasso and BISPS respectively with the horizon $h = 1$ and the rolling window size $W = 0.8 \times p$, where p is the number of time series (network size). Table 2 shows the rolling MSEs of the Lasso and BISPS, normalized by that of the AR(4) model, which is a conventional benchmark of macroeconomic forecasting.

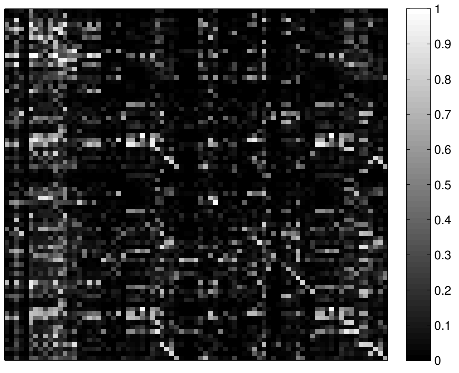
Compared with the AR(4) model, both Lasso and BISPS, based on a VAR model, have attained much smaller forecasting errors, except for Category 7 (prices). Therefore, by introducing the Granger causal interactions between different indices, we can build a multivariate network model that is more accurate than the univariate AR model in capturing the evolution of the U.S. macroeconomics, given the same amount of observations. The exception of Category 7 may be due to the higher lag order used in the univariate AR model.

Moreover, we note that BISPS gives smaller forecasting errors than Lasso for all the 12 categories of macroeconomic time series. It indicates that adopting a fusion of ℓ_1 and ℓ_2 penalties and imposing the stationarity constraint can capture the network dynamics more accurately and achieve a stronger capability of forecasting. To further support this conclusion, we apply Lasso and BISPS respectively to all the 108 variables with $W = 0.8 \times p$ and different horizons h. The rolling MSEs for $h = 1, 2, 4, 8, 16, 32$ is recorded in Table 3. As the horizon increases, the rolling MSE of Lasso grows exponentially, which clearly indicates that some estimates of the Lasso are nonstationary and thus fail to forecast for large horizons. On the other hand, the rolling MSE of BISPS stays stable for different horizons. This phenomenon is similar to what is shown in Figure 5. They have illustrated the fundamental difference of the $\mathbf{S}^2$ learning from the plain PML estimation.

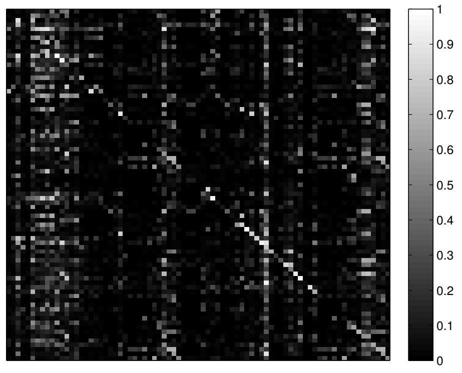
6.2 Bootstrap Analysis

In this experiment, we apply the SB-BISPS to the macroeconomic data before and after the "Great Moderation" (Davis and Kahn, 2008) and analyze the changes in their Granger causal connections. As the economic structure of U.S. has gone through huge changes in the Great Moderation in mid-1980, we expect to see significantly different causality networks before and after mid-1980. Hence, we divide the time series into two periods, the pre-Great Moderation period and the post-Great Moderation period, and apply SB-BISPS separately to the two periods.

For the pre-Great Moderation period, we use the data from 1960:I to 1979:IV as training set (80 observations); for the post-Great Moderation period, we use the data from 1985:I to 2004:IV (80 observations). The stationary bootstrap samples are obtained using the R function *tsboot* (Dalgaard, 2008) with default parameter values. The number of stationary bootstrap samples is set to be $B =$ 100. Figure 8 shows the COF (connection occurring frequency) matrices given by SB-BISPS for the pre-Great Moderation and the post-Great Moderation periods. We notice that the COF matrix of the pre-Great Moderation period has a higher energy level than that of the post-Great Moderation

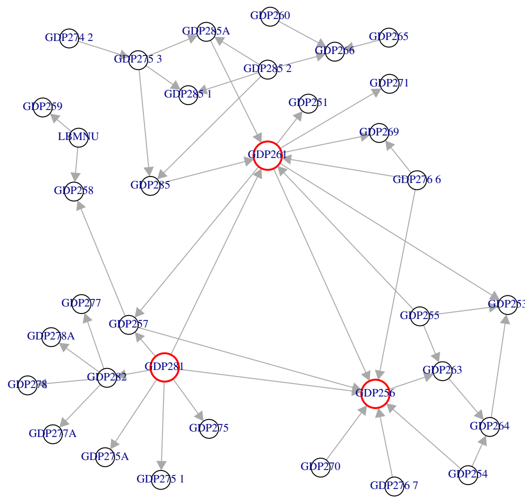


(a) pre-Great Moderation

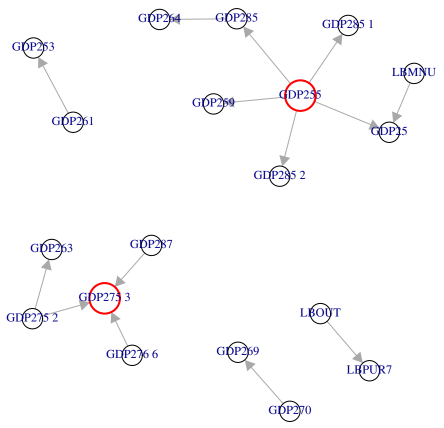


(b) post-Great Moderation

Figure 8: COFs of the pre-Great Moderation period and the post-Great Moderation period.



(a) pre-Great Moderation



(b) post-Great Moderation

Figure 9: Topologies of the macroeconomic network in the pre-Great Moderation period and the post-Great Moderation period. $f^* = 80\%$. Self-loops are not shown.

period. This indicates that the nodes are more actively interacting with each other in the pre-Great Moderation period than in the post-Great Moderation period, which effectively reflects the reduction in volatility of the business cycle fluctuations since the Great Moderation.

To illustrate the idea more clearly, we set the cutoff value for COF to be $f^* = 80\%$ and identify the most significant connections. The topologies obtained for the pre-Great Moderation period and the post-Great Moderation period are shown in Figure 9. Isolated nodes are removed. In the pre-Great Moderation period, the macro variables actively interact and form a complex dynamical network. There are three prominent variables, namely GDP281 (durable goods index), GDP256 and GDP261 (gross private domestic investment indices), which act like "hub" variables. They interact not only with many non-hub variables but also with each other. Therefore, there are no independent clusters. After the Great Moderation, on the other hand, the interactions have been remarkably reduced and most of the variables seem only self-regulated. This makes it easier for the network to stay stable. There are two hub variables, GDP255 (real personal consumption expenditure) and GDP275-3 (energy goods price index). The increasing importance of these two variables agrees with the observation that environmental regulations and energy policies have begun to influence the economic growth since the Great Moderation period (Jorgenson and Wilcoxen, 1990; Halkos and Tzeremes, 2011).

7. Conclusion

We have proposed the stationary-sparse ($\mathbf{S}^2$) learning of causality networks described in Granger's sense. Distinguished from the existing works, we explicitly incorporated the stationarity concern in a possibly ultra-high dimensional scenario and provided a probabilistic measure for the occurrence of any causal connection. We added a relaxed stationarity constraint in the penalized maximum likelihood estimation and proposed the BISPS algorithm which is easy to implement and computationally efficient. We must point out that although the algorithm is designed for the Berhu penalty, the framework extends to any convex penalties and their coupled thresholding rules. In network modeling, the number of unknown variables p^2 is often much larger than the number of observations n, which confronts us with an ultra-high dimensional problem. Therefore, we implanted the quantile thresholding iterative screening (QTIS) into the BISPS algorithm to improve scalability and computational efficiency. Furthermore, the stationary bootstrap enhanced BISPS (SB-BISPS) was proposed to provide a confidence measure for each possible connection in the network. The method has been successfully applied to the U.S. macroeconomic data, which leads to some interesting discoveries.

Our current work assumes multivariate Gaussian noise and focuses on learning the transition matrix. We will pursue the network learning in more general settings in the future. One particular problem of interest is to jointly capture the structure of the transition matrix and the concentration matrix, which may provide more comprehensive descriptions of the network. Also, we will consider the situations where the noise follows other distributions other than Gaussian, for example, the heavy-tail distribution. Finally, we will proceed to nonlinear time series models for network identification to handle more complex network data.

Acknowledgments

We would like to thank the editor and the anonymous referees for their careful comments and useful suggestions that significantly improve the quality of the paper. This work is partially supported by NSF under grants CCF-1117012 and CCF-1116447.

Appendix A. Proof of Theorem 1

We begin the proof by introducing some lemmas. Throughout the proof, we assume $\tau \geq 0, \lambda \geq 0, \eta \geq 0$, and $\lambda\eta \neq 0$.

Lemma 3 *Given the Berhu penalty* (9)*, the minimization problem*

$$\min_B \frac{1}{2}\|B-\Xi\|_F^2 + P_{\mathcal{B}}(B;\lambda,\eta) \tag{12}$$

has a unique optimal solution given by $\hat{B} = \Theta_{\mathcal{B}}(\Xi;\lambda,\eta)$*, where* $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ *is the Berhu thresholding rule defined as* (10)*.*

Proof It is easy to verify that $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is an odd, nondecreasing, shrinkage function that satisfies the definition of a thresholding rule given in She (2009). Following the construction procedure

$$P(t;\lambda,\eta) = \int_0^{|t|} (\sup\{s : \Theta(s;\lambda,\eta) \le u\} - u)du, \tag{13}$$

the Berhu penalty $P_{\mathcal{B}}(\cdot;\lambda,\eta)$ can be constructed from $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$. So $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is the coupled thresholding rule for $P_{\mathcal{B}}(\cdot;\lambda,\eta)$. By Lemma 1 in She (2012), $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ is the global minimizer of (12). ∎

Lemma 4 *The Berhu thresholding operator* $\Theta_{\mathcal{B}}(\cdot;\lambda,\eta)$ *is nonexpansive, that is,* $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)| \le |t-\tilde{t}|$ *for any* $t,\tilde{t} \in \mathbb{R}$*.*

The conclusion directly extends to the multivariate case: $\|\Theta_{\mathcal{B}}(A;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{A};\lambda,\eta)\| \le \|A-\tilde{A}\|$ for any $A,\tilde{A} \in \mathbb{R}^{p\times q}$.

Proof It is sufficient to show that the univariate Berhu thresholding operator $\Theta_{\mathcal{B}}$ is nonexpansive. Define $\Delta = |t-\tilde{t}|^2 - |\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)|^2$ and $a = |t|, b = |\tilde{t}|$.

a) Suppose $a \le \lambda, b \le \lambda$. Then $\Theta_{\mathcal{B}}(t;\lambda,\eta) = \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta) = 0$. So $\Delta = |t-\tilde{t}|^2 \ge 0$.

b) Suppose $a \le \lambda, \lambda < b \le \lambda+\lambda/\eta$. Then $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)|^2 = |\tilde{t} - \lambda sng(\tilde{t})|^2 = b^2+\lambda^2 - 2\lambda b$. So $\Delta = a^2+b^2-2ab-(b^2+\lambda^2-2\lambda b) = (\lambda-a)(2b-a-\lambda) \ge 0$.

c) Suppose $a \le \lambda, b \ge \lambda+\lambda/\eta$. Then $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)|^2 = |\frac{\tilde{t}}{1+\eta}|^2$. So $\Delta = a^2+b^2-2ab-\frac{b^2}{(1+\eta^2)} = a^2+[\frac{\eta^2+2\eta}{(1+\eta)^2}b-2a]b \ge a^2+[\frac{\eta^2+2\eta}{(1+\eta)^2}(\lambda+\lambda/\eta)-2a]\lambda = (a-\lambda)^2+\frac{1}{1+\eta}\lambda^2 \ge 0$.

d) Suppose $\lambda < a \le \lambda+\lambda/\eta, \lambda < b \le \lambda+\lambda/\eta$. Then $\Delta = 2\lambda(1-\mathrm{sgn}(t\tilde{t}))(a+b-\lambda) \ge 0$.

e) Suppose $\lambda < a \le \lambda+\lambda/\eta, b \ge \lambda+\lambda/\eta$. Then $\Delta = |t-\tilde{t}|^2 - |t-\lambda\mathrm{sgn}(t) - \frac{\tilde{t}}{\eta+1}|^2 = \frac{\eta(\eta+2)}{(\eta+1)^2}b^2 - 2b\frac{\eta a+\lambda}{\eta+1}\mathrm{sgn}(t\tilde{t}) + \lambda(2a-\lambda) = b[\frac{\eta(\eta+2)}{(\eta+1)^2}b - 2\frac{\eta a+\lambda}{\eta+1}\mathrm{sgn}(t\tilde{t})] + \lambda(2a-\lambda) \ge (\lambda+\lambda/\eta)[\frac{\eta(\eta+2)}{(\eta+1)^2}(\lambda+\lambda/\eta) - 2\frac{\eta a+\lambda}{\eta+1}\mathrm{sgn}(t\tilde{t})] + \lambda(2a-\lambda) = \frac{\lambda}{\eta}[\eta\lambda+2\lambda-2(\eta a+\lambda)\mathrm{sgn}(t\tilde{t})] + \lambda(2a-\lambda) \ge 0$.

f) Suppose $a \ge \lambda+\lambda/\eta, b \ge \lambda+\lambda/\eta$. Then $\Delta = |t-\tilde{t}|^2 - |\frac{t}{1+\eta} - \frac{\tilde{t}}{1+\eta}|^2 = \frac{\eta^2+2\eta}{(1+\eta)^2}|t-\tilde{t}|^2 \ge 0$.

Therefore, $|\Theta_{\mathcal{B}}(t;\lambda,\eta) - \Theta_{\mathcal{B}}(\tilde{t};\lambda,\eta)| \le |t-\tilde{t}|$ for any $t,\tilde{t} \in \mathbb{R}$. So the Berhu thresholding operator is nonexpansive. ∎

Lemma 5 *Let the SVD of B be* $B = USV^{\mathsf{T}}$*, where* $S = diag(\nu_1,\nu_2,\cdots,\nu_p)$ *with* $\nu_1,\nu_2,\cdots,\nu_p$ *being the singular values. Then* $\Pi(B;\tau)$ *defined by*

$$\Pi(B;\tau) = U diag(\min(\nu_1,\tau),\cdots,\min(\nu_p,\tau))V^{\mathsf{T}}$$

gives the projection of B into the convex set $\{B : \|B\|_2 \le \tau\}$*.*

Proof Let C be the projection of B into the convex set $\{B : \|B\|_2 \leq \tau\}$. Then C can be solved by

$$\min_C \|B - C\|_F^2,$$
$$\text{s.t. } \|C\|_2 \leq \tau.$$

To prove the lemma, we introduce von Neumann's trace inequality (von Neumann, 1937), which states that for any $p \times p$ matrices A and B with singular values $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_p$ and $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_p$ respectively,

$$|\mathrm{tr}\{AB\}| \leq \sum_{i=1}^{p} \alpha_i \beta_i, \tag{14}$$

where equality holds if and only if it is possible to find unitary matrices U and V that simultaneously singular value decompose A and B.

Let $B = U_0 S_0 V_0^\top$ and $C = USV^\top$ be the singular value decompositions of B and C respectively, where $S_0 = \mathrm{diag}(\nu_{0,1}, \nu_{0,2}, \cdots, \nu_{0,p})$ and $S = \mathrm{diag}(\nu_1, \nu_2, \cdots, \nu_p)$ with $\nu_{0,1} \geq \nu_{0,2} \geq \cdots \geq \nu_{0,p}$ and $\nu_1 \geq \nu_2 \geq \cdots \geq \nu_p$. Then,

$$\begin{aligned} \|B - C\|_F^2 &= \|B\|_F^2 + \|C\|_F^2 + 2\mathrm{tr}\{B^\top C\} \\ &\geq \|S_0\|_F^2 + \|S\|_F^2 + 2\mathrm{tr}\{S_0 S\} \\ &= \sum_{i=1}^{p} (\nu_{0,i} - \nu_i)^2. \end{aligned}$$

Equality holds if and only if $U = U_0$ and $V = V_0$. Optimality is achieved at $\nu_i = \min(\nu_{0,i}, \tau), i = 1, \cdots, p$. The proof is complete. ■

Lemma 6 *The projection operator* $\Pi(\cdot;\tau)$ *defined in Lemma 5 is nonexpansive, that is,* $\|\Pi(A) - \Pi(\tilde{A})\|_F \leq \|A - \tilde{A}\|_F$ *for any* $A, \tilde{A} \in \mathbb{R}^{p \times p}$.

Proof For simplicity, we denote $\Pi(\cdot;\tau)$ as $\Pi(\cdot)$. Let the SVDs for $p \times p$ matrices A and $\tilde{A}$ be $A = UDV^\top$ and $\tilde{A} = \tilde{U}\tilde{D}\tilde{V}^\top$ respectively. Define $\Delta = \|A - \tilde{A}\|_F^2 - \|\Pi(A) - \Pi(\tilde{A})\|_F^2$. Then,

$$\begin{aligned} \Delta &= \|UDV^\top - \tilde{U}\tilde{D}\tilde{V}^\top\|_F^2 - \|U\Pi(D)V^\top - \tilde{U}\Pi(\tilde{D})\tilde{V}^\top\|_F^2 \\ &= \|D\|_F^2 + \|\tilde{D}\|_F^2 - \|\Pi(D)\|_F^2 - \|\Pi(\tilde{D})\|_F^2 - 2\mathrm{tr}\{VDU^\top \tilde{U}\tilde{D}\tilde{V}^\top\} + 2\mathrm{tr}\{V\Pi(D)U^\top \tilde{U}\Pi(\tilde{D})\tilde{V}^\top\}. \end{aligned}$$

Applying von Neumann's trace inequality (14) again, we have

$$\begin{aligned} &-2\mathrm{tr}\{VDU^\top \tilde{U}\tilde{D}\tilde{V}^\top\} + 2\mathrm{tr}\{V\Pi(D)U^\top \tilde{U}\Pi(\tilde{D})\tilde{V}^\top\} \\ &= -2\mathrm{tr}\{V(D - \Pi(D))U^\top \tilde{U}\tilde{D}\tilde{V}^\top + V\Pi(D)U^\top \tilde{U}(\tilde{D} - \Pi(\tilde{D}))\tilde{V}^\top\} \\ &\geq -2\{\sum_{i=1}^{p} (d_i - \Pi(d_i))\tilde{d}_i + \sum_{i=1}^{p} (\tilde{d}_i - \Pi(\tilde{d}_i))\Pi(d_i)\}. \end{aligned}$$

Therefore,

$$\Delta \geq \sum_{i=1}^{p} \{(d_i - \tilde{d}_i)^2 - (\Pi(d_i) - \Pi(\tilde{d}_i))^2\}.$$

It is easy to verify that $(d_i - \tilde{d}_i)^2 - (\Pi(d_i) - \Pi(\tilde{d}_i))^2 \geq 0, i = 1, \cdots, p$. So $\Delta \geq 0$. The projection Π is a nonexpansive mapping. ■

Lemma 7 *Let $P^0 = Q^0 = 0$. The sequence $\{B^j\}$ of iterative procedure*

$$\begin{aligned} C^j &= \Theta_{\mathcal{B}}(B^j + P^j; \lambda, \eta), \\ P^{j+1} &= B^j + P^j - C^j, \\ B^{j+1} &= \Pi(C^j + Q^j; \tau), \\ Q^{j+1} &= C^j + Q^j - B^{j+1} \end{aligned} \tag{15}$$

converges to a globally optimal solution to

$$\begin{aligned} &\min_B \frac{1}{2}\|B - B^0\|_2^2 + P_{\mathcal{B}}(B; \lambda, \eta), \\ &\quad s.t.\ \|B\|_2 \leq \tau. \end{aligned} \tag{16}$$

Procedure (15) is designed for a penalized minimization problem with a convex constraint based on Dykstra's projection algorithm (Dykstra, 1983; Boyle and Dykstra, 1986).
Proof First, we rewrite the problem as

$$\min_B \frac{1}{2}\|B - B^0\|_2^2 + f(B) + g(B), \tag{17}$$

where $f(B) = P_{\mathcal{B}}(B; \lambda, \eta)$ and $g(B) = \mathbb{1}_{\|B\|_2 \leq \tau}$ is an indicator function for $\|B\|_2 \leq \tau$, defined as

$$\mathbb{1}_{\|B\|_2 \leq \tau} = \begin{cases} 0 & \text{if } \|B\|_2 \leq \tau \\ +\infty & \text{otherwise.} \end{cases}$$

It is easy to show that $g(B)$ is a proper lower semicontinuous convex function, $f(B)$ is a proper continuous (hence lower semicontinuous) convex function (Rockafellar, 1970) and they satisfy

$$\text{dom} f \cap \text{dom} g \neq \emptyset.$$

Lemma 3 and Lemma 5 imply that $\Theta_{\mathcal{B}}(\cdot; \lambda, \eta)$ and $\Pi(\cdot; \tau)$ are the proximity operators (Moreau, 1962) of $f(B)$ and $g(B)$ respectively:

$$\text{prox}_f B = \Theta_{\mathcal{B}}(B; \lambda, \eta) \text{ and } \text{prox}_g B = \Pi(B; \tau).$$

Therefore, by Theorem 3.2 and Theorem 3.3 in Bauschke and Combettes (2008), it holds that

$$B^j \to \text{prox}_{f+g} B^0.$$

Hence, the sequence $\{B^j\}$ converges to a globally optimal solution to problem (17). ■

Now we can establish Theorem 1. Recall that Algorithm 1 is to solve

$$\min_A f(A;\lambda,\eta) = \frac{1}{2}\|Y - XA\|_F^2 + P_{\mathcal{B}}(A;\lambda,\eta),$$
$$\text{s.t. } \|A\|_2 \le 1.$$

We use Opial's conditions (Opial, 1967) to prove the convergence of $\{A^k\}$. To be specific, we show that the iteration of Step 1 to Step 3 in Algorithm 1 is a nonexpansive asymptotically regular mapping with a nonempty set of fixed points.

Lemma 8 *For the sequence $\{A^k\}$ generated by Algorithm 1,*

$$f(A^k;\lambda,\eta) - f(A^{k+1};\lambda,\eta) \ge \frac{1}{2}(k_0^2 - \|X\|_2^2)\|A^{k+1} - A^k\|_F^2.$$

Proof First define

$$g(A,B;\lambda,\eta) = \frac{1}{2}\|Y - XB\|_F^2 + P_{\mathcal{B}}(B;\lambda,\eta) + \frac{1}{2}\text{tr}\{(B-A)^\top(k_0^2 I - X^\top X)(B-A)\}.$$

Given A, minimizing g over B is equivalent to

$$\min_B \frac{1}{2}\|B - A - \frac{1}{k_0^2}(X^\top Y - X^\top XA)\|_F^2 + P_{\mathcal{B}}(B;\lambda/k_0^2,\eta/k_0^2)$$
$$\text{s.t. } \|B\|_2 \le 1.$$

We can obtain its globally optimal solution by performing the iterative procedure (15) in Lemma 7, substituting $\tau \leftarrow 1, B^0 \leftarrow A + \frac{1}{k_0^2}(X^\top Y - X^\top XA), \lambda \leftarrow \lambda/k_0^2, \eta \leftarrow \eta/k_0^2$. Therefore, we have

$$f(A^k;\lambda,\eta) = g(A^k,A^k;\lambda,\eta) \ge g(A^k,A^{k+1};\lambda,\eta) = f(A^{k+1};\lambda,\eta) + \frac{1}{2}(k_0^2 - \|X\|_2^2)\|A^{k+1} - A^k\|_F^2.$$

The proof is complete. ■

Given $k_0 > \|X\|_2$, Lemma 8 implies that the sequence $\{A^k\}$ is asymptotically regular (Browder and Petryshyn, 1966).

Lemma 9 *The sequence $\{A^k\}$ generated by the iteration of Algorithm 1 is uniformly bounded.*

Proof First, based on Lemma 8 we have

$$P_{\mathcal{B}}(A^k;\lambda,\eta) \le f(A^k;\lambda,\eta) \le f(A^0;\lambda,\eta) \triangleq C.$$

This implies that $P_{\mathcal{B}}(a_{ij}^k;\lambda,\eta) \le C$, $\forall 0 \le i,j \le p$.

If $|a_{ij}^k| \le \lambda/\eta$, we have $\lambda|a_{ij}^k| \le C$, which implies $(a_{ij}^k)^2 \le \max(\lambda^2/\eta^2, C^2/\lambda^2)$. If $|a_{ij}^k| > \lambda/\eta$, we have $\frac{\eta^2 t^2 + \lambda^2}{2\eta} \le C$, which implies $(a_{ij}^k)^2 \le \frac{2\eta C - \lambda^2}{\eta^2}$. Given $\lambda\eta \ne 0$,

$$(a_{ij}^k)^2 \le \max\left(\lambda^2/\eta^2, C^2/\lambda^2, \frac{2\eta C - \lambda^2}{\eta^2}\right) \triangleq C_2,\ 1 \le i,j \le p.$$

Hence,

$$\|A^k\|_F^2 \le p^2 C_2.$$

The sequence $\{A^k\}$ is uniformly bounded. ■

Lemma 10 *The iteration of Step 1 to Step 3 in Algorithm 1 is a nonexpansive mapping.*

Proof From Lemma 4 and Lemma 6, $\Theta_{\mathcal{B}}$ and Π are nonexpansive mappings. In fact, the composition of nonexpansive mappings is also nonexpansive. So the inner iteration given by Step 2 in Algorithm 1 is nonexpansive. When $k_0 \ge \|X\|_2$, Step 1 in Algorithm 1 is nonexpansive. Again, the composition of Step 1 and Step 2 is nonexpansive. Hence, the iteration of Algorithm 1 is nonexpansive. ■

Lemma 9 and Lemma 10 imply that the mapping of Algorithm 1 is a nonexpansive mapping into a bounded closed convex subset. By Theorem 1 in Browder (1965), it has a fixed point. Then, with all Opial's conditions satisfied, the sequence $\{A^k\}$ has a unique limit point, denoted as A^*, and it is a fixed point of Algorithm 1.

Next, we prove that A^* must be a global minimizer of problem (8). Denote $h(A) = \|A\|_2 - 1$. By Lemma 7 and Lemma 8, A^* satisfies the KKT conditions (Boyd and Vandenberghe, 2004) of problem (16) with $\tau = 1$:

$$\begin{cases} 0 \in A^* - B^0 + \partial P_{\mathcal{B}}(A^*; \lambda/k_0^2, \eta/k_0^2) + \nu^* \partial h(A^*), \\ h(A^*) \le 0, \\ \nu^* \ge 0, \\ \nu^* h(A^*) = 0. \end{cases}$$

Substituting $B^0 = A^* + \frac{1}{k_0^2}(\Sigma_{XY} - \Sigma_{XX}A^*)$, we have

$$\begin{cases} 0 \in \Sigma_{XY} - \Sigma_{XX}A^* + \partial P_{\mathcal{B}}(A^*; \lambda, \eta) + \tilde{\nu}^* \partial h(A^*), \\ h(A^*) \le 0, \\ \tilde{\nu}^* \ge 0, \\ \tilde{\nu}^* h(A^*) = 0. \end{cases} \tag{18}$$

Note that problem (8) is convex and its KKT conditions are given by (18). Hence, A^* is a global minimizer of problem (8). The proof is complete.

Appendix B. Proof of Theorem 2

First, we introduce a quantile thresholding rule $\Theta^{\#}(\cdot; m)$ as a variant of the hard thresholding rule. Given $1 \le m \le pq$: $A \in \mathbb{R}^{p\times q} \to B \in \mathbb{R}^{p\times q}$ is defined as follows: $b_{ij} = a_{ij}$ if $|a_{ij}|$ is among the m largest in the set of $\{|a_{ij}| : 1 \le i \le p, 1 \le j \le q\}$, and $b_{ij} = 0$ otherwise.

To prove the function value decreasing property, we introduce the following lemma.

Lemma 11 *$\hat{B} = \Theta^{\#}(A;m)$ is a globally optimal solution to*

$$\min_B l(B) = \frac{1}{2}\|A-B\|_F^2$$
$$s.t.\ \|B\|_0 \le m.$$

Proof Let $I \subset \{(i,j)|1 \le i \le p, 1 \le j \le q\}$ with $|I| = m$. Assuming $B_{I^c} = 0$, we get the optimal solution $\hat{B}$ with $\hat{B} = A_I$. It follows that $l(\hat{B}) = \frac{1}{2}\|A\|_F^2 - \frac{1}{2}\sum_{i,j\in I} a_{ij}^2$. Therefore, the quantile thresholding $\Theta^{\#}(A;m)$ yields a global minimizer. ■

Define a surrogate function

$$\tilde{l}(A,B) = \frac{1}{2}\|Y - XB\|_F^2 + \frac{1}{2}\mathrm{tr}\{(B-A)^\top(k_0^2 - X^\top X)(B-A)\}.$$

Based on Lemma 11 and $k_0 \ge \|X\|_2$, the function value decreasing property can be proved following the lines of Lemma 8. So we have

$$l(A^k) = \tilde{l}(A^k,A^k) \ge \tilde{l}(A^k,A^{k+1}) = l(A^{k+1}) + \frac{1}{2}(k_0^2 - \|X\|_2^2)\|A^{k+1} - A^k\|_F^2 \ge l(A^{k+1}).$$

The proof is complete.

References

H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, 1974.

Ya. I. Alber, A. N. Iusem, and M. V. Solodov. On the projected subgradient method for nonsmooth convex optimization in a hilbert space. *Mathematical Programming*, pages 23–35, 1998.

H. H. Bauschke and P. L. Combettes. A dykstra-like algorithm for two monotone operators. *Pacific Journal of Optimization*, 4(3):383–391, Sep. 2008.

T. Blumensath and M. E. Davies. Normalized iterative hard thresholding: Guaranteed stability and performance. *IEEE Journal of Selected Topics in Signal Processing*, 4(2), Apr. 2010.

S. Boyd, N. Parikh, E. Chu, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Information Systems Journal*, 3(1):1–118, 2010.

S.P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. ISBN 9780521833783.

J. P. Boyle and R. L Dykstra. A method for finding projections onto the intersection of convex sets in hilbert spaces. In *Advances in Order Restricted Statistical Inference*, pages 28–47. Springer, 1986.

F. E. Browder. Nonexpansive nonlinear operators in a banach space. *Proceedings of the National Academy of Sciences of the United States of America*, 54(4):1041, 1965.

F. E. Browder and W. V. Petryshyn. The solution by iteration of nonlinear functional equations in banach spaces. *Bull. Amer. Math. Soc.*, 72:571–575, 1966.

E. Bullmore and O. Sporns. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, 10:186–198, Mar. 2009.

J. V. Burke, A. S. Lewis, and M. L. Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM J. on Optimization*, 15(3):751–779, Mar. 2005. ISSN 1052-6234.

J. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956–1982, Mar. 2010.

F. Curtis and M. Overton. A sequential quadratic programming algorithm for nonconvex, nonsmooth constrained optimization. *SIAM Journal on Optimization*, 22(2):474–500, 2012.

P. Dalgaard. *Introductory Statistics with R, Statistics and Computing*. Springer, Aug. 2008.

S. J. Davis and J. A. Kahn. Interpreting the great moderation: Changes in the volatility of economic activity at the macro and micro levels. *Journal of Economic Perspectives*, 22(4):155–180, 2008.

D.L. Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, Apr. 2006.

R. L. Dykstra. An algorithm for restricted least squares regression. *Journal of the American Statistical Association*, 78(384):837–842, 1983.

B. Efron. Bootstrap methods: another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

J. J. Faith, B. Hayete, J. T. Thaden, I. Mogno, J. Wierzbowski, G. Cottarel, S. Kasif, J. J. Collins, and T. S. Gardner. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS Biol*, 5(1):e8, Jan. 2007.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96:1348–1360, Dec. 2001.

J. Fan and R. Li. Statistical challenges with high dimensionality: feature selection in knowledge discovery. In *International Congress of Mathematicans*, Aug. 2006.

J. Fan and J. Lv. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5):849–911, 2008.

J. Fan and J. Lv. A selective overview of variable selection in high dimensional feature space. *Stat Sin.*, 20(1):101–148, Jan. 2010.

J. Fan, R. Samworth, and Y. Wu. Ultrahigh dimensional feature selection: Beyond the linear model. *Journal of Machine Learning Research*, 10:2013–2038, Dec. 2009. ISSN 1532-4435.

C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37(3):424–438, 1969.

M. C. Grant and S. P. Boyd. Graph implementations for nonsmooth convex programs. In *Recent Advances in Learning and Control*, pages 95–110. Springer, 2008.

G. E. Halkos and N. G. Tzeremes. Oil consumption and economic efficiency: A comparative analysis of advanced, developing and emerging economies. *Ecological Economics*, 70(7):1354–1362, May 2011.

S. Hanneke, W. Fu, and E. P. Xing. Discrete temporal models of social networks. *Electronic Journal of Statistics*, 4:585–605, 2010.

B.S. He, H. Yang, and S.L. Wang. Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *Journal of Optimization Theory and Applications*, 106 (2):337–356, 2000. ISSN 0022-3239.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

N. Hsu, H. Hung, and Y. Chang. Subset selection for vector autoregressive processes using lasso. *Computational Statistics and Data Analysis*, 52(7):3645–3657, 2008.

P. J. Huber. *Robust Statistics*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 1981.

W. James and C. Stein. Estimation with quadratic loss. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 361–379, 1961.

D. W. Jorgenson and P. J. Wilcoxen. Environmental regulation and U.S. economic growth. *RAND Journal of Economics*, 21(2):314–340, Summer 1990.

H. R. Kunsch. The jackknife and the bootstrap for general stationary observations. *The Annals of Statistics*, 17(3):1217–1241, 1989.

T. G. Lewis. *Network Science: Theory and Applications*. Wiley Publishing, 2009.

Z. Lin. Some software packages for partial SVD computation. *CoRR*, abs/1108.1548, 2011.

H. Lütkepohl. *New Introduction to Multiple Time Series Analysis*. Springer, 2nd edition, 2007.

T.C. Mills and R.N. Markellos. *The Econometric Modelling of Financial Time Series*. Cambridge University Press, 2008.

J. J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Reports of the Paris Academy of Sciences, Series A*, 255:2897–2899, 1962.

M. E. J. Newman and D. J. Watts. *The Structure and Dynamics of Networks*. Princeton University Press, 2006.

Z. Opial. Weak convergence of the sequence of successive approximations for nonexpansive mappings. *Bull. Am. Math. Soc.*, 73:591–597, 1967.

M. L. Overton and R. S. Womersley. On minimizing the spectral radius of a nonsymmetric matrix function - optimality conditions and duality theory. *SIAM J. Matrix Anal. Appl.*, 9(4):473–498, Oct. 1988.

A. B. Owen. A robust hybrid of lasso and ridge regression. *Prediction and Discovery (Contemporary Mathematics)*, 443:59–71, 2007.

D. N. Politis and J. P. Romano. The stationary bootstrap. *Journal of the American Statistical Association*, 89(428):1303–1313, 1994.

G. C. Reinsel. *Elements of Multivariate Time Series Analysis*. New York, Springer, 2nd edition, 1997.

R.T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

Y. She. Thresholding-based iterative selection procedures for model selection and shrinkage. *Electron. J. Statist*, 3:384–415, 2009.

Y. She. An iterative algorithm for fitting nonconvex penalized generalized linear models with grouped predictors. *Computational Statistics and Data Analysis*, 56(10):2976–2990, Oct. 2012.

C. A. Sims. Macroeconomics and reality. *Econometrica*, 48(1):1–48, Jan. 1980.

J. Songsiri and L. Vandenberghe. Topology selection in graphical models of autoregressive processes. *Journal of Machine Learning Research*, 9999:2671–2705, Dec. 2010. ISSN 1532-4435.

J. H. Stock and M. W. Watson. Generalized shrinkage methods for forecasting using many predictors. *Journal of Business & Economic Statistics*, 30(4):481–493, Oct. 2012.

J. F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones, 1998.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.

Y. Tsaig and D. L. Donoho. Extensions of compressed sensing. *Signal Processing*, 86(3):549–571, 2006.

R. H. Tütüncü, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, 95:189–217, 2003.

J. von Neumann. Some matrix inequalities and metrization of matrix space. *Tomsk. Univ. Rev.*, 1: 153–167, 1937.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.