Sparse Multivariate Factor Regression

Milad Kharratzadeh, Mark Coates
Electrical and Computer Engineering Department, McGill University
Montreal, Quebec, Canada
milad.kharratzadeh@mail.mcgill.ca, mark.coates@mcgill.ca

Abstract

We consider the problem of multivariate regression in a setting where the relevant predictors could be shared among different responses. We propose an algorithm which decomposes the coefficient matrix into the product of a long matrix and a wide matrix, with a separate ℓ_1 penalty on each. The first matrix linearly transforms the predictors to a set of latent factors, and the second one regresses the responses on these factors. Our algorithm simultaneously performs dimension reduction and coefficient estimation and automatically estimates the number of latent factors from the data. Our formulation results in a non-convex optimization problem, which despite its flexibility to impose effective low-dimensional structure, is difficult, or even impossible, to solve exactly in a reasonable time. We specify an optimization algorithm based on alternating minimization to solve this non-convex problem and provide theoretical results on its convergence and local optimality. Finally, we demonstrate the effectiveness of our algorithm via experiments on simulated and real data.

I. Introduction

Multivariate regression analysis, also known as multiple–output regression, is concerned with modelling the relationships between a set of real–valued output vectors, known as responses, and a set of real–valued input vectors, known as predictors or features. The multivariate responses are measured over the same set of predictors and are often correlated. Hence, the goal of multivariate regression is to exploit these dependencies to learn a predictive model of responses based on an observed set of input vectors paired with corresponding outputs. Multiple–output regression can also be seen as an instance of the problem of multi–task learning, where each task is defined as predicting individual responses based on the same set of predictors. The multivariate regression problem is encountered in numerous fields including finance [1], computational biology [2], geostatistics [3], chemometrics [4], and neuroscience [5].

In this paper, we are interested in multivariate regression tasks where it is reasonable to believe that the responses are related to factors, each of which is a sparse linear combination of the predictors. Our model further assumes that the relationships between the factors and the responses are sparse. This type of structure occurs in a number of applications and we provide two examples in later sections.

Given p–dimensional predictors $\mathbf{x_i} = (x_{i1}, \ldots, x_{ip})^T \in \mathbb{R}^p$ and q–dimensional responses $\mathbf{y_i} = (y_{i1}, \ldots, y_{iq})^T \in \mathbb{R}^q$ for the i-th sample, we assume there is a linear relationship between the inputs and outputs as follows:

$$\mathbf{y_i} = \mathbf{D}^T\mathbf{x_i} + \boldsymbol{\epsilon}_i, \qquad i = 1, \ldots, N, \tag{1}$$

where $\mathbf{D}_{p\times q}$ is the regression coefficient matrix and $\boldsymbol{\epsilon}_i = (\epsilon_{i1}, \ldots, \epsilon_{iq})$ is the vector of errors for the i-th sample. We can combine these N equations into a single matrix formula:

$$\mathbf{Y} = \mathbf{XD} + \mathbf{E}, \tag{2}$$

where $\mathbf{X}$ denotes the $n \times p$ matrix of predictors with $\mathbf{x_i}^T$ as its i-th row, $\mathbf{Y}$ denotes the $n \times q$ matrix of responses with $\mathbf{y_i}^T$ as its i-th row, and $\mathbf{E}$ denotes the $n \times q$ matrix of errors with $\boldsymbol{\epsilon}_i^T$ as its i-th row. For $q = 1$, this multivariate linear regression model reduces to the well–known, univariate linear regression model.

We assume that the columns of $\mathbf{X}$ and $\mathbf{Y}$ are centred and hence the intercept terms are omitted. We also assume that the error vectors for N samples are iid Gaussian random vectors with zero mean and covariance $\boldsymbol{\Sigma}$, i.e. $\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}), i = 1, \ldots, N$.

In the absence of additional structure, many standard procedures for solving (2), such as linear regression and principal component analysis, are not consistent unless $p/n \to 0$. Thus, in a high-dimensional setting where p is comparable to or greater than n, we need to impose some low-dimensional structure on the coefficient matrix. For instance, element-wise sparsity can be imposed by constraining the ℓ_1 norm of the coefficient matrix, $\|\mathbf{D}\|_{1,1}$ [6], [7]. This regularization is equivalent to solving q separate univariate lasso regressions for every response; thus we consider tasks separately. Another way to introduce sparsity is to consider the mixed ℓ_1/ℓ_γ norms ($\gamma > 1$). In this approach (sometimes called group lasso), the mixed norms impose a block-sparse structure where each row is either all zero or mostly zeros. Particular examples, among many other works, include results using the ℓ_1/ℓ_∞ norm [8], [9], and the ℓ_1/ℓ_2 norm [10], [11]. Also, there are the so-called "dirty" models which are superpositions of simpler low-dimensional structures such as element-wise and row-wise sparsity [12], [13] or sparsity and low rank [14], [15].

Another approach is to impose a constraint on the rank of the coefficient matrix. In this approach, instead of constraining the regression coefficients directly, we can apply penalty functions on the rank of $\mathbf{D}$, its singular values and/or its singular vectors [16]–[20]. These algorithms belong to a broad family of dimension-reduction methods known as *linear factor regression*, where the responses are regressed on a set of *factors* achieved by a linear transformation of the predictors. The coefficient matrix is decomposed into two matrices: $\mathbf{D} = \mathbf{A}_{p\times m}\mathbf{B}_{m\times q}$. Matrix $\mathbf{A}$ transforms the predictors into m latent factors, and matrix $\mathbf{B}$ determines the factor loadings.

Our contributions

Here, we propose a novel algorithm which performs sparse multivariate factor regression (SMFR). We jointly estimate matrices $\mathbf{A}$ and $\mathbf{B}$ by minimizing the mean-squared error, $\|\mathbf{Y}-\mathbf{XAB}\|_F^2$, with ℓ_1 penalties on both matrices. We provide a formulation to estimate the number of effective latent factors, m. To the best of our knowledge, our work is the first to strive for low-dimensional structure by imposing sparsity on both factoring and loading matrices. This can result in a set of interpretable factors and loadings with high predictive power; however, these benefits come at the cost of a non-convex objective function. Most current approaches for multivariate regression solve a convex problem (either through direct formulation or by relaxation of a non-convex problem) to impose low-dimensional structures on the coefficient matrix. Although non-convex formulations, such as the one introduced here, can be employed to achieve very effective representations in the context of multivariate regression, there are few theoretical performance guarantees for optimization schemes solving such problems. We formulate our problem in Section II. In Section III, we propose an optimization procedure based on alternating minimization and provide theoretical guarantees for its convergence and local optimality. We show that under mild conditions on the predictor matrix, every limit point of the minimization algorithm is a local minimum of the objective function. Through analysis of simulations on synthetic datasets in Section V and two real-world datasets in Section VI, we show that compared to other multivariate regression algorithms, our proposed algorithm can provide a more effective representation of the data, resulting in a higher predictive power.

Related Methods

Many multivariate regression techniques impose a low-dimensional structure on the coefficient matrix. Element-wise sparsity, here noted as LASSO, is the most common approach where the cost function is defined as $\|\mathbf{D}\|_{1,1}$ [6], [7]. An extension of LASSO to the multivariate case is the row-wise sparsity with the ℓ_1/ℓ_2 norm as the cost function: $\|\mathbf{D}\|_{1,2}$ [10], [11]. Peng et al. proposed a method, called RemMap [13], which imposes both element-wise and row-wise sparsity and solves the following problem:

$$\min_{\mathbf{D}} \|\mathbf{Y}-\mathbf{XD}\|_F^2 + \lambda_1\|\mathbf{D}\|_{1,1} + \lambda_2\|\mathbf{D}\|_{1,2}.$$

In an alternative approach, [18] extended the partial least squares (PLS) framework by imposing an additional sparsity constraint and proposed Sparse PLS (SPLS).

Another common idea is to employ dimensionality reduction techniques to find the underlying latent structure in the data. One of the most basic algorithms in this class is an approach called Reduced Rank Regression (RRR) [21] where the sum-of-squares error is minimized under the constraint that rank$(\mathbf{D}) \leq r$ for some $r \leq \min\{p,q\}$. It is easy to show that one can find a closed-form solution for $\mathbf{D}$ based on the singular value decomposition of $\mathbf{Y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$. However, similar to least-squares, the solution of this problem without appropriate regularization exhibits poor predictive performance and is not suitable for high-dimensional settings. Another popular approach is to use the trace norm as the penalty function:

$$\min_{\mathbf{D}} \|\mathbf{Y}-\mathbf{XD}\|_F^2 + \lambda \sum_{j=1}^{\min\{p,q\}} \sigma_j(\mathbf{D}), \tag{3}$$

where $\sigma_j(\mathbf{D})$ denotes the j'th singular value of $\mathbf{D}$. The trace norm regularization has been extensively studied in the literature [19], [22]–[24]. It imposes sparsity in the singular values of $\mathbf{D}$ and therefore, results in a low-dimensional solution (higher values of λ correspond to achieving solutions of lower rank).

Many papers study problems of the following form:

$$\min_{\mathbf{D}} \|\mathbf{Y}-\mathbf{X}\mathbf{D}\|_F^2 + g(\mathbf{D}) \quad \text{s.t.} \quad \text{rank}(\mathbf{D}) \le r \le \min\{p,q\}, \tag{4}$$

where $g(\mathbf{D})$ is a regularization function over $\mathbf{D}$. For instance, in [25], a ridge penalty is proposed with $g(\mathbf{D}) = \lambda\|\mathbf{D}\|_F^2$. Often it is assumed that $\mathbf{D} = \mathbf{A}_{p\times r}\mathbf{B}_{r\times q}$ (and thus $\text{rank}(\mathbf{A}) \le r$, $\text{rank}(\mathbf{B}) \le r$, and consequently $\text{rank}(\mathbf{D}) \le r$) and the problem is formulated in terms of $\mathbf{A}$ and $\mathbf{B}$. In [26], $g(\mathbf{A},\mathbf{B}) = \lambda_1\|\mathbf{A}\|_{1,1} + \lambda_2\|\mathbf{B}\|_F^2$. An algorithm called Sparse Reduced Rank Regression (SRRR) is proposed in [17], where $g(\mathbf{A},\mathbf{B}) = \lambda\|\mathbf{A}\|_{1,2}$ with an extra constraint that $\mathbf{B}\mathbf{B}^T = \mathbf{I}$. In [22], $g(\mathbf{A},\mathbf{B}) = \lambda\|\mathbf{B}\|_{2,1}^2$ with an extra constraint that $\mathbf{A}^T\mathbf{A} = \mathbf{I}$, and it is assumed that $p \le q$ and $r = p$. Dimension reduction is achieved by the constraint on $\mathbf{B}$ which forces many rows to be zero which consequently cancels the effects of the corresponding columns in $\mathbf{A}$.

Our problem formulation differs in two important ways: (i) sparsity constraints are imposed on *both* $\mathbf{A}$ and $\mathbf{B}$; and (ii) the number of factors is determined directly, without the need for cross-validation. We will discuss the latter aspect in detail in the next section. The former difference has substantial consequences; when decomposing the coefficient matrix into two matrices, the first matrix has the role of aggregating the input signals to form the latent factors and the second matrix performs a multivariate regression on these factors. Imposing sparsity on $\mathbf{A}$ enhances the variable selection as well as the interpretability of the achieved factors. Also, as originally motivated by LASSO, we would like to impose the sparsity constraint on $\mathbf{B}$ in order to improve the interpretability and prediction performance.

Imposing sparsity on both $\mathbf{A}$ and $\mathbf{B}$ means that for our model to make sense for a specific problem, the outputs should be related in a sparse way to a common set of factors that are derived as a sparse combination of the inputs. For example, in analyzing the S&P 500[1] stocks, it is well-known that returns exhibit much stronger correlations for companies that belong to the same industry sector. The memberships of each sector are not always so clear, because a company may have several diverse activities that generate revenue. So if we are using just concurrent stock returns to try to predict those of other companies, it is reasonable to assume that factors representing industry sectors should appear [27]. Since we do not expect the main sectors to overlap much, a company will not be present in many factors; so, it is reasonable to assume that $\mathbf{A}$ is sparse. Moreover, most companies will only be predicted by one or two such factors, so it makes sense that $\mathbf{B}$ is also sparse.

There is a connection between Sparse PCA [28] and our algorithm, in the case where $\mathbf{Y}$ is replaced by $\mathbf{X}$ (i.e., $\mathbf{X}$ is regressed on itself). We investigate this relationship in Section IV.

II. Problem Setup

In this work, we introduce a novel low-dimensional structure where we decompose $\mathbf{D}$ into the product of two sparse matrices $\mathbf{A}_{p\times m}$ and $\mathbf{B}_{m\times q}$ where $m < \min(p,q)$. This decomposition can be interpreted as first identifying a set of m factors which are derived by some linear transformation of the predictors (through matrix $\mathbf{A}$) and then identifying the transformed regression coefficient matrix $\mathbf{B}$ to estimate the responses from these m factors. We provide a framework to find m, the number of effective latent factors, as well as the transforming and regression matrices, $\mathbf{A}$ and $\mathbf{B}$. For a fixed m, define:

$$\widehat{\mathbf{A}}_m, \widehat{\mathbf{B}}_m = \underset{\mathbf{A}_{p\times m},\mathbf{B}_{m\times q}}{\text{argmin}} \; f(\mathbf{A},\mathbf{B}), \tag{5}$$

where

$$f(\mathbf{A},\mathbf{B}) = \frac{1}{2}\|\mathbf{Y}-\mathbf{X}\mathbf{A}\mathbf{B}\|_F^2 + \lambda_1\|\mathbf{A}\|_{1,1} + \lambda_2\|\mathbf{B}\|_{1,1}. \tag{6}$$

Then, we solve the following optimization problem:

$$\widehat{m} = \max(m) \le r \quad \text{s.t} \; \text{rank}(\widehat{\mathbf{A}}_m) = \text{rank}(\widehat{\mathbf{B}}_m) = m, \tag{7}$$

where r is a problem-specific bound on the number of factors. We then choose $\widehat{\mathbf{A}}_{\widehat{m}}$ and $\widehat{\mathbf{B}}_{\widehat{m}}$ as solutions. Thus, we find the maximum number of factors such that the solution of (5) has full rank factor and loading matrices. In other words, we find the maximum m such that the best possible regularized reconstruction of responses, i.e., the solution of (5), results in a model where the factors (columns of of $\widehat{\mathbf{A}}$) and their contributions to the responses (rows of $\widehat{\mathbf{B}}$) are linearly independent. To achieve this, we initialize $\mathbf{A}$ to have r columns, $\mathbf{B}$ to have r rows, and set $m = r$, solve the problem (3)-(4), check for the full rank condition; if not satisfied, set $m = m - 1$, and repeat the process until we identify an m that satisfies the rank condition.

[1]Standard and Poors index of 500 large-cap US equities http://ca.spindices.com/indices/equity/sp-500

In choosing m, we want to avoid both overfitting (large m) and lack of sufficient learning power (small m). In general, we only require $m \le \min(p, q)$; however, in practical settings where p and q are very large, we impose an upper bound on m to have a reasonable number of factors and avoid overfitting. This upper bound, denoted by r, is problem-specific and should be chosen by the programmer. For instance, continuing our example of analysing the S&P 500 stocks, most economists identify 10-15 primary financial sectors, so a choice of $r = 15$ or 20 is reasonable since we expect the factors representing industries to appear, but we allow the algorithm to find the optimal m from the data. In order to have the maximum learning power, we find the maximum $m \le r$ for which the solutions satisfy our rank conditions. This motivates starting with $m = r$ and decreasing it until the conditions hold (as opposed to starting with $m = 1$ and increasing the dimension).

The full rank conditions are employed to guarantee a good estimate of the number of "effective" factors. An effective factor explains some aspect of the response data but cannot be constructed as a linear combination of other factors (otherwise it is superfluous). We therefore require the estimated factors to be linearly independent. In addition, we require that the rows of $\mathbf{B}$, which determine how the factors affect the responses, are linearly independent. If we do not have this latter independence, we could reduce the number of factors and still obtain the same relationship matrix $\mathbf{D}$, so at least one of the factors is superfluous. By enforcing that $\mathbf{A}$ and $\mathbf{B}$ are full rank, we make sure that the estimated factors are linearly independent in both senses, and thus $\widehat{m}$ is a good estimate of the number of effective factors.

In estimating the number of factors, we differ fundamentally from the common approach in literature. Setting aside the differences in the choice of regularization, most algorithms minimize a cost function as in (5) for a fixed m *without* the rank condition in (7). They then use cross-validation to find the optimal $\widetilde{m}$ [13], [17], [19], [20]. The criterion in choosing $\widetilde{m}$ is thus the cross-validation error. In contrast, we strive to find the largest $\widehat{m}$ such that the solutions of (7) have full rank and the estimated factors are linearly independent. Finding $\widetilde{m}$ via cross-validation may result in non-full rank solutions with linearly dependent factors. Therefore, some factors can be expressed as linear combinations of other factors and can be viewed as redundant. Including redundant factors (via a non-full rank $\mathbf{A}$) could help to improve the sparsity of $\mathbf{B}$, but our goal is to have the minimum necessary set of factors, not to have a sparse $\mathbf{B}$ at any cost. Later, with experiments on synthetic and real data, we show that our approach towards choosing the number of factors results in better predictive performance as well as more interpretable factors compared to other techniques that apply cross-validation.

III. Optimization Technique and Theoretical Results

The optimization problem defined in (5–7) is not a convex problem and it is difficult, if not impossible, to solve exactly (i.e., to find the global optimum) in polynomial time. Therefore, we have to employ heuristic algorithms, which may or may not converge to a locally or globally optimal solution. In this section, we propose an algorithm based on alternate minimization, and show that it is locally well-posed: under certain conditions, it converges to a local minimum. We present our results through a series of theorems with detailed proofs in the Appendix.

For a fixed m, the objective function in (5) is biconvex in $\mathbf{A}$ and $\mathbf{B}$; it is not convex in general, but is convex if either $\mathbf{A}$ or $\mathbf{B}$ is fixed. Let us define $\mathbf{C} = (\mathbf{A}, \mathbf{B})$. To solve (5) for a fixed m, we perform Algorithm 1, with an arbitrary, non-zero starting value $\mathbf{A}_0$ (see Section V for a discussion on the choice of the starting point). The stopping criterion is related to the convergence of the value of function f, not the convergence of its arguments. In our experiments, we assume f has converged, if $\frac{|f_i - f_{i+1}|}{f_i} < \epsilon$ where the default value of the tolerance parameter, ϵ, is 0.001; i.e., the algorithm stops if the changes in f are less than 0.1%.

Algorithm 1 Solving problem (5) for fixed m

```
$\mathbf{A} \leftarrow \mathbf{A}_0$, $i \leftarrow 0$
while value of $f(\mathbf{A}, \mathbf{B})$ not converged do
    $\mathbf{B}_{i+1} \leftarrow \underset{\mathbf{B}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\mathbf{A}_i\mathbf{B}\|_F^2 + \lambda_2\|\mathbf{B}\|_{1,1}$    (8)
    $\mathbf{A}_{i+1} \leftarrow \underset{\mathbf{A}}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\mathbf{A}\mathbf{B}_{i+1}\|_F^2 + \lambda_1\|\mathbf{A}\|_{1,1}$    (9)
    $i \leftarrow i + 1$
end while
$\widehat{\mathbf{A}}, \widehat{\mathbf{B}} \leftarrow$ values of $\mathbf{A}$ and $\mathbf{B}$ at convergence
```

Definition 1. $\mathbf{C}^* = (\mathbf{A}^*, \mathbf{B}^*)$ is called a *partial optimum* of f if

$$\begin{aligned} f(\mathbf{A}^*, \mathbf{B}^*) &\leq f(\mathbf{A}^*, \mathbf{B}), \ \forall \mathbf{B} \in \mathbb{R}^{m \times q} && (10) \\ \text{and} \quad f(\mathbf{A}^*, \mathbf{B}^*) &\leq f(\mathbf{A}, \mathbf{B}^*), \ \forall \mathbf{A} \in \mathbb{R}^{n \times m}. && (11) \end{aligned}$$

Definition 2. A point $\mathbf{C}^*$ is an *accumulation point* or a *limit point* of a sequence $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$, if for any neighbourhood V of $\mathbf{C}^*$, there are infinitely many $j \in \mathbb{N}$ such that $\mathbf{C}_j \in V$. Equivalently, $\mathbf{C}^*$ is the limit of a subsequence of $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$.

Main Results: We show that the sequence $f(\mathbf{C}_i)$ monotonically converges. We also show that the solution of (8) is unique if $\mathbf{A}_i$ is full rank and similarly, the solution of (9) is unique if $\mathbf{B}_{i+1}$ is full rank. Then, we prove that for any given starting point, the sequence of solutions, $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$, has at least one accumulation point and if $\mathbf{A}$ or $\mathbf{B}$ for all accumulation points are full rank, then all the accumulation points are partial optima and have the same function value. Moreover, if both $\mathbf{A}$ and $\mathbf{B}$ for all accumulation points are full rank, the difference between successive solutions of the algorithm converges to zero, i.e., $\lim_{i \to \infty} \|\mathbf{C}_{i+1} - \mathbf{C}_i\| = 0$. Finally, we show that all the partial optima of f are indeed local optima (i.e., not saddle points).

Proposition 1. *The sequence $f(\mathbf{A}_i, \mathbf{B}_i)$ generated by Algorithm 1 converges monotonically.*

The value of f is always positive and is reduced in each of the two main steps of Algorithm 1. Thus, it is guaranteed that the stopping criterion of Algorithm 1 will be reached.

Theorem 1. *If the entries of $\mathbf{X} \in \mathbb{R}^{n \times p}$ are drawn from a continuous probability distribution on $\mathbb{R}^{np}$, then:*
(i) The solution of (8) is unique if $\mathbf{A}_i$ is full rank.
(ii) The solution of (9) is unique if $\mathbf{B}_{i+1}$ is full rank.

In classical LASSO, the condition on the entries of $\mathbf{X}$ is sufficient to achieve solution uniqueness [29]. For LASSO, the continuity is used to argue that the columns of $\mathbf{X}$ are in *general position* with probability 1 (see Appendix B, Definition 3 for a formal definition of general position). The affine span of the columns of $\mathbf{X}$, $\{\mathbf{X}_1, \ldots, \mathbf{X}_{k+1}\}$, has Lebesgue measure 0 in $\mathbb{R}^n$ for a continuous distribution on $\mathbb{R}^n$, so there is zero probability of drawing $\mathbf{X}_{k+2}$ in their span. If we multiply $\mathbf{X}$ by a matrix $\mathbf{A}$ with full column rank, we retain the same property, and thus the solution of (8) is unique if $\mathbf{A}_i$ is full rank. It is also shown in [29] that given the assumption on the elements of $\mathbf{X}$, a *sufficient* condition for the solution uniqueness of (9) is that the function $g(\mathbf{U}) = \|\mathbf{Y} - \mathbf{U}\mathbf{B}_{i+1}\|_F^2$ be strictly convex (the function argument is defined as $\mathbf{U} = \mathbf{X}\mathbf{A}$). We can show that g is strictly convex if $\mathbf{B}_{i+1}$ has full row rank (refer to the Appendix for detailed proofs). Next, we study the properties of $\widehat{\mathbf{C}} = (\widehat{\mathbf{A}}, \widehat{\mathbf{B}})$ at convergence.

Theorem 2. *Assume that the entries of $\mathbf{X} \in \mathbb{R}^{n \times p}$ are drawn from a continuous probability distribution on $\mathbb{R}^{np}$. For a given starting point $\mathbf{A}_0$, let $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$ denote the sequence of solutions generated by Algorithm 1. Then:*
(i) $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$ has at least one accumulation point.
(ii) If $\mathbf{A}$ or $\mathbf{B}$ for all accumulation points of $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$ is full rank, then all the accumulation points of $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$ are partial optima and have the same function value.
(iii) If both $\mathbf{A}$ and $\mathbf{B}$ are full rank for all accumulation points of $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$, then:

$$\lim_{i \to \infty} \|\mathbf{C}_{i+1} - \mathbf{C}_i\| = 0, \tag{12}$$

Part (i) follows from the fact that the solutions produced by Algorithm 1 are contained in a bounded, closed (and hence compact) set. Although Algorithm 1 converges to a specific value of f, this value can be achieved by different values of $\mathbf{C}$. Thus, the sequence $\mathbf{C}_i$ can have many accumulation points. Part (ii) of Theorem 2 shows that any accumulation point is a partial optimum, provided that $\mathbf{A}$ or $\mathbf{B}$ for all accumulation points of $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$ is full rank and the entries of $\mathbf{X} \in \mathbb{R}^{n \times p}$ are drawn from a continuous probability distribution on $\mathbb{R}^{np}$. Proposition 1 implies that for any given starting point, all the associated accumulation points have the same f value. Under the stronger assumption that *both* $\mathbf{A}$ and $\mathbf{B}$ are full rank for all accumulation points of $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$, part (iii) provides a guarantee that the difference between successive solutions of the algorithm converges to zero, for both the factor and loading matrices. Although the condition in (12) does not guarantee the convergence of the sequence $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$, it is close enough for practical purposes.

Theorem 3. *Assume that the entries of $\mathbf{X} \in \mathbb{R}^{n \times p}$ are drawn from a continuous probability distribution on $\mathbb{R}^{np}$. Then, any partial optimum of f, say $\mathbf{C}^* = (\mathbf{A}^*, \mathbf{B}^*)$, for which either $\mathbf{A}^*$ or $\mathbf{B}^*$ is full rank, is also a local minimum of f. Therefore, if $\mathbf{A}$ or $\mathbf{B}$ for all accumulation points of $\{\mathbf{C}_i\}_{i \in \mathbb{N}}$ is full rank, any accumulation point of Algorithm 1 is a local minimum.*

Partial optimality of $(\mathbf{A}^*, \mathbf{B}^*)$ implies that $f(\mathbf{A}^*, \mathbf{B}^*) \leq f(\mathbf{A}, \mathbf{B}^*)$ and $f(\mathbf{A}^*, \mathbf{B}^*) \leq f(\mathbf{A}^*, \mathbf{B})$ for any $\mathbf{A}$ and $\mathbf{B}$. We show that if either $\mathbf{A}^*$ or $\mathbf{B}^*$ is full rank (and thus the solution of either (8) or (9) is unique), we get a stronger result that $f(\mathbf{A}^*, \mathbf{B}^*) \leq f(\mathbf{A}, \mathbf{B})$ for any $\mathbf{A}$ and $\mathbf{B}$ in the neighborhood of $(\mathbf{A}^*, \mathbf{B}^*)$. Combining this with part (ii) of Theorem 2, proves that any accumulation point of Algorithm 1 is a local minimum.

Now, we propose Algorithm 2 to solve the optimization problem described in (5–7) to find the number of latent factors as well as the factor and loading matrices. Optimal values of λ_1 and λ_2 are found via 5-fold cross-validation.

Algorithm 2 Sparse Multivariate Factor Regression (SMFR) via Alternating Minimization

```
1: Input: Training Set $\mathbf{X}_{n\times p}, \mathbf{Y}_{n\times q}, \lambda_1, \lambda_2$
2: Output: Solution of problem (5–7): $\widehat{\mathbf{A}}, \widehat{\mathbf{B}}, \widehat{m}$
3: $m \leftarrow r$                                  ▷ $r$: upper bound on the number of factors
4: while true do
5:     $\mathbf{A} \leftarrow \mathbf{A}_0 \in \mathbb{R}^{p\times m}$, $i \leftarrow 0$
6:     while value of $f(\mathbf{A}, \mathbf{B})$ not converged do
7:         $\mathbf{B}_{i+1} \leftarrow \mathrm{argmin}_{\mathbf{B}} \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\mathbf{A}_i\mathbf{B}\|_F^2 + \lambda_2\|\mathbf{B}\|_{1,1}$
8:         $\mathbf{A}_{i+1} \leftarrow \mathrm{argmin}_{\mathbf{A}} \frac{1}{2}\|\mathbf{Y} - \mathbf{X}\mathbf{A}\mathbf{B}_{i+1}\|_F^2 + \lambda_1\|\mathbf{A}\|_{1,1}$
9:         $i \leftarrow i + 1$
10:    end while
11:    $\widehat{\mathbf{A}}, \widehat{\mathbf{B}} \leftarrow$ values of $\mathbf{A}$ and $\mathbf{B}$ at convergence
12:    if rank($\widehat{\mathbf{A}}$) $< m$ or rank($\widehat{\mathbf{B}}$) $< m$ then
13:        $m \leftarrow m - 1$
14:    else
15:        break
16:    end if
17: end while
```

IV. Fully Sparse PCA

In [28], Zou, Hastie, and Tibshirani propose a sparse PCA, arguing that in regular PCA "each principal component is a linear combination of all the original variables, thus it is difficult to interpret the results". Assume that we have a data matrix $\mathbf{X}_{n\times p}$ with the following SV decomposition: $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$. The principal components are defined as $\mathbf{Z} = \mathbf{U}\mathbf{D}$ with the corresponding columns of $\mathbf{V}$ as the loadings. In [30], Zou et al. show that solving the following optimization problem leads to exact PCA:

$$(\widehat{\mathbf{A}}, \widehat{\mathbf{B}}) = \underset{\mathbf{A},\mathbf{B}}{\mathrm{argmin}} \|\mathbf{X} - \mathbf{X}\mathbf{A}\mathbf{B}\|_F^2 + \lambda \sum_i \|\mathbf{A}_i\|_2^2 \text{ s.t. } \mathbf{B}\mathbf{B}^T = \mathbf{I},$$

where $\mathbf{A}_i$ denotes the i'th column of $\mathbf{A}$. Thus, we have $\widehat{\mathbf{A}}_i \propto \mathbf{V}_i$, and $\mathbf{X}\widehat{\mathbf{A}}_i$ corresponds to the i'th principal component. Sparse PCA (SPCA) is introduced by adding an ℓ_1 penalty on matrix $\mathbf{A}$ to the objective function. Zou et al. propose an alternating minimization scheme to solve this problem [28].

The ordinary principal components are uncorrelated and their loadings are orthogonal. SPCA imposes sparsity on the construction of the principal components. Here sparsity means that each component is a combination of only a few of the variables. By enforcing sparsity, the principal components become correlated and the loadings are no longer orthogonal. On the other hand, SPCA assumes, like regular PCA, that the contributions of these components are orthonormal ($\mathbf{B}\mathbf{B}^T = \mathbf{I}$). In our algorithm, if we replace $\mathbf{Y}$ with $\mathbf{X}$, i.e., regressing $\mathbf{X}$ on itself, we get a similar algorithm. However, our work differs in two ways. First, we also impose sparsity on the contributions of principal components. This comes at the expense of higher computational costs, but results in more interpretable results. Also, the contributions will not be orthonormal anymore. However, by the full rank constraint we impose on the two matrices, we make sure that the principal components and their contributions are linearly independent. Moreover, the algorithm provides a mechanism for learning from the data how many principal components are sufficient to explain the data.

V. Simulation Study

In this section, we use synthetic data to compare the performance of our algorithm with several related multivariate regression methods reviewed in the introduction.

A. Simulation Setup

We generate the synthetic data in accordance with the model described in (2), $\mathbf{Y} = \mathbf{X}\mathbf{D} + \mathbf{E}$, where $\mathbf{D} = \mathbf{A}\mathbf{B}$. First, we generate an $n \times p$ predictor matrix, $\mathbf{X}$, with rows independently drawn from $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_X)$, where the (i, j)-th element of $\mathbf{\Sigma}_X$ is defined as $\sigma^X_{i,j} = 0.7^{|j-i|}$. This is a common model for predictors in the literature [13], [19], [31]. The rows of the $n \times q$ error matrix are sampled from $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_N)$, where the (i, j)-th element of $\mathbf{\Sigma}_N$ is defined as $\sigma^X_{i,j} = \sigma_n^2 \cdot 0.4^{|j-i|}$. The value of σ_n^2 is varied to attain different levels of signal to noise ratio (SNR). Each row of the $p \times m$ matrix $\mathbf{A}$ is chosen by first randomly selecting m_0 of its elements and sampling them from $\mathcal{N}(0, 1)$ and then setting the rest of its elements to zero. Finally, we generate the $m \times q$ matrix $\mathbf{B}$ by the element-wise product of $\mathbf{B} = \mathbf{U} \circ \mathbf{W}$, where the elements of $\mathbf{U}$ are drawn independently from $\mathcal{N}(0, 1)$ and elements of $\mathbf{W}$ are drawn from Bernoulli distribution with success probability s.

We evaluate the performance of a given algorithm with three different metrics. We evaluate the predictive performance over a test set $(\mathbf{X}_{test}, \mathbf{Y}_{test})$, separate from the training set, in terms of the mean-squared error:

$$\text{MSE} = \frac{\|\mathbf{X}_{test}\widehat{\mathbf{D}} - \mathbf{Y}_{test}\|_F^2}{nq}, \tag{13}$$

where $\widehat{\mathbf{D}}$ is the estimated coefficient matrix. In our case, we have $\widehat{\mathbf{D}} = \widehat{\mathbf{A}}\widehat{\mathbf{B}}$. We also compare different algorithms based on their signed sensitivity and specificity of the support recognition:

$$\text{Signed Sensitivity} = \frac{\sum_{i,j} \mathbf{1}[d_{i,j} \cdot \widehat{d}_{i,j} > 0]}{\sum_{i,j} \mathbf{1}[d_{i,j} \neq 0]},$$

$$\text{Specificity} = \frac{\sum_{i,j} \mathbf{1}[d_{i,j} = 0] \cdot \mathbf{1}[\widehat{d}_{i,j} = 0]}{\sum_{i,j} \mathbf{1}[d_{i,j} = 0]},$$

where $\mathbf{1}$ represents the indicator function.

We compare the performance of our algorithm, SMFR, with many other algorithms reviewed in Section I as well as a baseline algorithm with a simple ridge penalty. We consider three different regimes: (i) high-dimensional problems with few instances (50) compared to the number of predictors or responses (50, 100, or 150); (ii) problems with increased number of instances ($p, q < n$); and (iii) problems where the structural assumption of our technique is violated. In the first regime, which is of most interest to us due to high-dimensionality, we explore different parameter settings. The values of σ_n and s affect the SNR—lower values of σ_n and higher values of s correspond to higher values of SNR (e.g., $\sigma_n = 5, s = 0.1$ corresponds to a very low SNR). In regime (iii), we violate the assumption about the structure of the coefficient matrix, i.e., $\mathbf{D} = \mathbf{A}\mathbf{B}$, in two ways. In the first case, $\mathbf{D}$ has an element-wise sparsity with a density of 20%; in the second, it has row-wise sparsity where 60% of the rows are all zeros and the rest have 30% non-zero elements. We consider these cases to compare our algorithm with others in an unfavourable setting.

Algorithms Implementation: For LASSO, we use the SPAMS [32] package which is coded in C++ with a Matlab interface. We use this toolbox for solving the first part of the alternating minimization in our algorithm. For the second part of the alternating minimization, we use the L1General package in Matlab [33]. For SPLS [34] and RemMap [35], we use the R packages provided by the authors. For ℓ_1/ℓ_2, we use the RemMap package and set $\lambda_1 = 0$. Trace norm and ridge regularizations are implemented by the MALSAR package in Matlab [36]. Finally, since there is not any code available online for SRRR, we implemented it directly in Matlab. See [37] for more details.

B. Results

a) Predictive performance: The means and standard deviations of different algorithms are presented in Table I. We use five-fold cross-validation to find the tuning parameters of all algorithms. We set r, the maximum number of factors, to 20. For the first two simulation regimes, our algorithm outperforms the other algorithms and results in lower MSE means and standard deviation. The improvements are more significant in the high-dimensional settings with high SNR. However, in settings with low SNR ($\sigma_n = 5, s = 0.1$) or high number of instances ($n = 500$),

Parameters							MSE over test set							
n	p	q	m	m_0	σ_n	s	SMFR	LASSO	ℓ_1/ℓ_2 [11]	SRRR [17]	RemMap [13]	SPLS [18]	Trace [23]	Ridge
50	150	50	10	1	3	0.2	0.074 (0.003)	0.083 (0.005)	0.090 (0.005)	0.084 (0.005)	0.083 (0.005)	0.091 (0.007)	0.088 (0.004)	0.089 (0.004)
			10	1	3	0.4	0.083 (0.006)	0.104 (0.008)	0.105 (0.007)	0.099 (0.007)	0.104 (0.008)	0.110 (0.008)	0.110 (0.007)	0.111 (0.006)
			10	1	5	0.2	0.110 (0.004)	0.118 (0.005)	0.133 (0.004)	0.117 (0.005)	0.123 (0.004)	0.122 (0.007)	0.115 (0.005)	0.122 (0.006)
			15	2	3	0.2	0.096 (0.005)	0.108 (0.006)	0.112 (0.007)	0.109 (0.008)	0.107 (0.006)	0.114 (0.008)	0.109 (0.006)	0.110 (0.008)
50	100	100	10	1	5	0.1	0.069 (0.002)	0.070 (0.002)	0.092 (0.002)	0.071 (0.002)	0.075 (0.002)	0.073 (0.002)	0.071 (0.002)	0.074 (0.002)
500	150	50	10	1	3	0.2	0.0173 (0.0001)	0.0180 (0.0001)	0.0198 (0.0001)	0.0176 (0.0001)	0.0184 (0.0001)	0.0216 (0.0007)	0.0183 (0.0002)	0.0187 (0.0001)
500	100	100	10	1	5	0.3	0.0202 (0.0001)	0.0209 (0.0002)	0.0222 (0.0001)	0.0204 (0.0001)	0.0214 (0.0001)	0.0222 (0.0003)	0.0208 (0.0001)	0.0213 (0.0002)
50	100	100	*element-wise sparsity*		5	—	0.084 (0.001)	0.078 (0.001)	0.096 (0.002)	0.080 (0.001)	0.085 (0.001)	0.081 (0.001)	0.078 (0.002)	0.081 (0.001)
50	150	50	*row-wise sparsity*		3	—	0.083 (0.004)	0.076 (0.003)	0.080 (0.003)	0.079 (0.003)	0.075 (0.003)	0.083 (0.003)	0.081 (0.001)	0.102 (0.004)

TABLE I
COMPARISON OF SIX ALGORITHMS FOR DIFFERENT SETUPS. WE REPORT MEAN AND STANDARD DEVIATIONS OF THE MSE OVER THE TEST SETS, BASED ON 20 SIMULATION RUNS.

we still observe lower errors for SMFR. On average, our algorithm reduces the test error by 8.3% compared with LASSO, 16.7% compared with ℓ_1/ℓ_2, 7.4% compared with SRRR, 10.2% compared with RemMap, and 14.8% compared with SPLS.

In the last two simulations, where the assumed factor structure is abandoned, our algorithm has no advantage over simpler methods with no factor structure (such as LASSO) and gives a higher error.

b) Variable selection: In Figure 1, we compare the average signed sensitivity and specificity of different algorithms (based on 20 simulation runs) as the number of instances increase (other parameters kept fixed). We observe that our algorithm has higher sensitivity and specificity. This effect for specificity is reduced as the number of instances increases. This shows that our algorithm is more advantageous in high-dimensional settings where the number of instances is comparable to or less than the number of predictors/responses. Although we only show the plots for a specific parameter setting, the results are similar for other parameters. In Figure 2, we provide a similar comparison but for the case where our assumed problem structure is violated (element-wise sparse coefficient matrix with density 20%). Our algorithm performs worse compared to those that have model assumptions that better match the underlying data, achieving sensitivity and specificity values 20-30% lower than LASSO and RemMap.

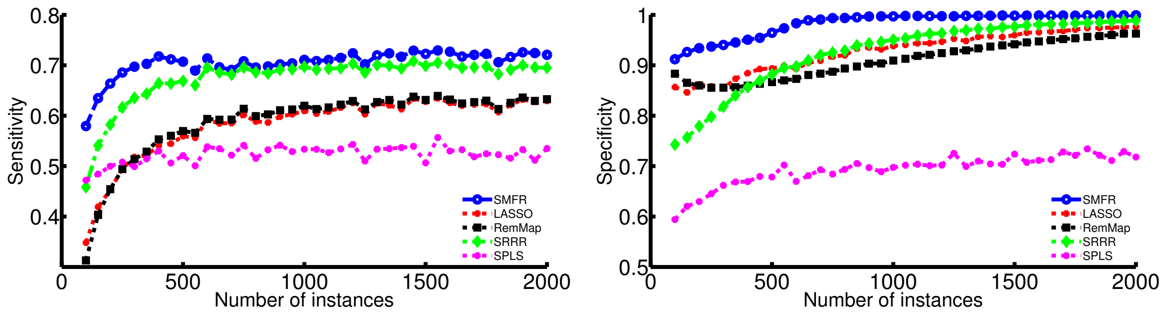


Fig. 1. Sensitivity and specificity comparison of different algorithms as the number of instances increases. ($p = 150, q = 50, m = 10, m_0 = 1, \sigma_n = 3, s = 0.3$)

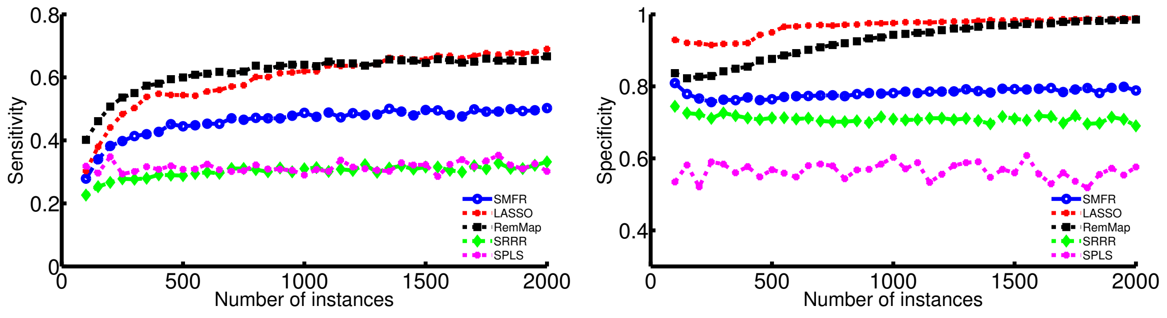


Fig. 2. Sensitivity and specificity comparison of different algorithms as the number of instances increases for the case where the structural assumption is violated (element-wise sparsity with density of 20% and $p = 50, q = 50, \sigma_n = 5$)

c) Number of latent factors: In Table II, we compare the number of estimated factors for the three algorithms that perform dimensionality reduction. For SRRR and SPLS, we find the number of factors by 5-fold cross-validation. The true number of factors is 10 in all parameter settings. We observe that except for the first set of parameters, our algorithm provides as good or better estimates of the number of factors.

n	p	q	m	m_0	σ_n	s	SMFR	SRRR	SPLS
50	150	50	10	1	3	0.3	14, 14.1, 2.3	8, 9.0, 1.0	15, 14.9, 3.8
50	150	50	10	1	5	0.3	11, 10.6, 2.0	8, 7.8, 2.2	9, 9.3, 3.7
50	150	50	10	1	5	0.2	10, 9.8, 2.9	7, 6.7, 2.4	7, 8.2, 2.9
50	100	100	10	1	5	0.1	7, 7.4, 2.1	6, 6.1, 3.0	4, 4.4, 1.3
500	150	50	10	1	3	0.2	10, 10.6, 1.1	10, 9.9, 0.2	15, 15, 0.0
500	100	100	10	1	5	0.3	10, 9.9, 0.3	10, 9.9, 0.2	15, 15, 0.0

TABLE II
MEDIAN, MEAN, AND STANDARD DEVIATION OF ESTIMATED NUMBER OF FACTORS (BASED ON 20 RUNS).

d) Computation time: We also compare the computation time of different algorithms for the parameter settings corresponding to the first row of Table I. We report the median computation time (on a PC with 16GB RAM and quad-core CPU at 3.4GHz) of each algorithm (excluding the cross-validation part) over 20 runs; SMFR: 4.0s, SRRR: 5.6s, RemMap: 0.3s, SPLS: 0.3s, LASSO: 0.02s, and ℓ_1/ℓ_2: 0.33s. Since the algorithms are implemented using different programming languages and stopping criteria vary slightly, care must be taken when interpreting these results. The main message is that SMFR and SRRR have considerably larger computation times, since they solve more complicated problems which involve estimating the number of factors, and the loading and factoring matrices. This extra information has value in itself and provides more insight about the structure of data. There are potential approaches for improving the speed of our algorithm, but that is beyond the scope of this paper.

e) SMFR initialization: In all the experiments above, and the ones in the next section, we use a random initialization for our algorithm where the elements of $\mathbf{A}_0$ are sampled independently from $\mathcal{N}(0,1)$. We compare this initialization with two other methods which are based the matrix factorization of other solutions: (i) using the SVD of LASSO solution, $\mathbf{D}_{\text{LASSO}} = \mathbf{USV}^T$, setting $\mathbf{A}_0 = \mathbf{U}_{1:m}\mathbf{S}_{1:m}$, where the subscript $1\!:\!m$ indicates choosing the first m columns; and (ii) using the SVD of trace-norm solution, $\mathbf{D}_{\text{Trace}} = \mathbf{USV}^T$, setting $\mathbf{A}_0 = \mathbf{U}_{1:m}\mathbf{S}_{1:m}$. We compare these three initializations for the case where the data is generated according to the first regime of Table I, $p = 150, q = 50, n = 50$, and $m = 10$, and vary the level of noise and sparsity. The results are shown in Table III. The procedure to find the number of effective factors (i.e., solving, checking rank condition, reducing m by one otherwise) is the same for all three initializations. Also, as a baseline comparison, we show the results for the usual LASSO regression in the last row. Our experiments illustrate that a random initialization is the most effective.

The results reported in Table III are based on 20 runs (i.e., 20 different realizations of the data). It would be interesting to examine how the results change based on different random initializations for a fixed realization of the data. In Table IV, we show the results for 5 realizations of the data. For each realization, we run our algorithm with 20 different random initializations and report the mean, standard deviation, and maximum of MSE over the test set. For comparison, we also include the MSE achieved by the decomposition of LASSO and trace norm solutions. The second row for each of these initializations indicates the number of random initializations (out of 20) with a higher error. For example, for instance #1, only two of the random initializations result in a higher error than LASSO initialization. The results show that the variation of MSE over different initializations is not significant and most random initializations perform better than using the decomposition of other solutions as the starting point.

	$\sigma_n = 3, s = 0.2$	$\sigma_n = 3, s = 0.3$	$\sigma_n = 5, s = 0.2$	$\sigma_n = 5, s = 0.3$
Random Initialization	0.074 (0.003)	0.081 (0.004)	0.110 (0.004)	0.116 (0.005)
LASSO Initialization	0.076 (0.003)	0.084 (0.006)	0.112 (0.004)	0.115 (0.005)
Trace Norm Initialization	0.076 (0.004)	0.085 (0.008)	0.112 (0.004)	0.120 (0.006)
LASSO	0.083 (0.005)	0.096 (0.006)	0.118 (0.005)	0.128 (0.006)

TABLE III
COMPARING DIFFERENT INITIALIZATION TECHNIQUES. ($p = 150, q = 50, n = 50$, AND $m = 10$)

Initialization		Instance #1	Instance #2	Instance #3	Instance #4	Instance #5
Random	Mean (std)	0.066 (0.002)	0.068 (0.003)	0.070 (0.003)	0.076 (0.003)	0.069 (0.002)
	Max	0.073	0.074	0.076	0.084	0.073
LASSO	Value	0.068	0.073	0.074	0.077	0.075
	# lower than random	2	1	1	6	0
Trace Norm	Value	0.070	0.071	0.074	0.077	0.074
	# lower than random	1	2	1	6	0

TABLE IV
MSE VARIATION OVER 20 DIFFERENT RANDOM INITIALIZATIONS FOR THE SAME PROBLEM.

VI. APPLICATION TO REAL DATA

We apply the proposed algorithm to real-world datasets and show that it exhibits better or similar predictive performance compared to state-of-the-art algorithms. We show that the factoring identified by our algorithm provides valuable insight into the underlying structure of the datasets.

A. Montreal's bicycle sharing system (BIXI)

The first dataset we consider provides information about Montreal's bicycle sharing system called BIXI. The data contains the number of available bikes in each of the 400 installed stations for every minute. We use the data collected for the first four weeks of June 2012. From this dataset we form the set of predictors and responses as follows. We allocate two features to each station corresponding to the number of arrivals and departures of bikes to or from that station for every hour. The learning task is to predict the number of arrivals and departures for all the stations from the number of arrivals and departures in the last hour (i.e., a vector autoregressive model). The choice of this model is a compromise between accuracy and complexity. Mathematically, we want to estimate $\mathbf{D}$ such that $\mathbf{Y} \simeq \mathbf{XD}$, where $\mathbf{X}_{t,j}$ and $\mathbf{X}_{t,400+j}$ respectively show the number of arrivals and departures in hour t at station j and $\mathbf{Y}_{t,j}$ and $\mathbf{Y}_{t,400+j}$ respectively show the number of arrivals and departures in hour $t+1$ at station j.

We perform the prediction task on each of the four weeks. For each week, we take the data for the first 5 days (120 data points; Friday to Tuesday) as the training set (with the fifth day data as the validation set), and the last two days as the test set (48 data points; Wednesday and Thursday). We compare the algorithms performing dimensionality reduction in terms of their predictive performance on the test sets and the number of chosen factors in Table V. We also include LASSO and ℓ_1/ℓ_2 as baseline algorithms. To avoid showing very small numbers, we present the value of MSE, defined in (13), times nq. In terms of the prediction performance, we observe that our algorithm outperforms the others in all 4 weeks.

We can also compare the algorithms in terms of the number of chosen factors. SMFR always chooses more factors compared to SRRR resulting in a better representation of data according to MSE. So, SRRR seems to underestimate the number of factors. SPLS chooses almost a constant number of factors, showing that it is not adaptive to the changes in the data over the four weeks. These changes arise because he pattern of bike use can be affected by different public events, weather, road constructions, and many other factors that can be present in one week and not in the others. For instance, in Figure 3, we show the total number of arrivals for three stations over the four weeks. We observe that for the first station, the level of activity in the second week is more than 2.5 times the level of activity in the first week. Or for the second station, the activity level decreases around 40% from the second week to the fourth week. Given that the average number of arrivals over all stations is almost the same for the four weeks with mean= 360 and std= 25 (i.e., overall network activity is the same over the four weeks), these individual changes seem significant.

week		SMFR	SRRR	SPLS	LASSO	ℓ_1/ℓ_2
1	error	560.1	570.0	1661	580.4	591.0
	factors	7	3	7	—	—
2	error	589.1	602.2	1888	610.9	623.7
	factors	9	3	6	—	—
3	error	639.7	641.9	2159	643.4	657.8
	factors	14	7	5	—	—
4	error	556.5	594.6	1621	594.9	588.0
	factors	4	3	6	—	—

TABLE V
TOTAL SQUARED ERROR (MSE$\times nq$) AND THE NUMBER OF FACTORS FOR BIXI DATASET

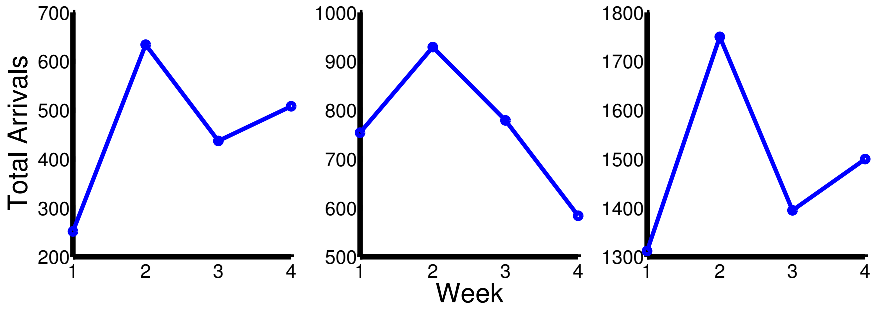


Fig. 3. The total number of arrivals in each of the four weeks for three selected stations. For these stations, the activity varies significantly for different weeks.

To gain more insight into the quality of predictions, we randomly choose a feature in week 4 and plot the predictions made by different algorithms over the test set in Figure 4; the results for other features are similar. The y-axis represents the cumulative number of bikes. We observe that SMFR provides a better fit to the data.

We are using the data from the weekends as well as three weekdays to learn the prediction model, so it is reasonable to ask whether it is sensible to combine weekends and weekdays. Our preliminary data analysis indicated that the data on weekends are not particularly different from the other days, and thus we can include weekends in our training sets. For example, the correlation between the number of arrivals on weekends and the number of arrivals on weekdays across stations, is very high (around 0.9) for all four weeks, showing that the relative level of activity in a station (compared to other stations) is similar for weekends and weekdays; i.e, if a station has relatively high number of arrivals on the weekdays, it is highly likely that it will have a relatively high number of arrivals on the weekends too. We have repeated the experiment after removing the weekends and obtained similar results.

To further investigate the variable selection of our algorithm, we run it on the whole data and examine the resulting factors. Three of these are shown in Figure 5. In these figures, all the bike stations are shown with green plus signs. For each factor, we show its constituent features; red crosses and blue circles correspond respectively to the departure and arrival features of each station that are present in that factor. Examining these factors provide useful insight into the data. For instance, the factor in Figure 5(a) shows that the departures from populated residential areas (The Plateau, Mile End, Outremont) and arrivals at downtown (Ville Marie) are combined together to form a factor. This agrees with the intuition that many people are taking bikes to go from their homes to downtown where universities and businesses are located. The factor in Figure 5(b) shows another strong effect which corresponds to the flow from the peripheries of downtown to more central locations (Place des Arts, Old Port). Many hotels and several universities are situated at the edge of downtown; numerous restaurants, cafes and tourist sites are located in the

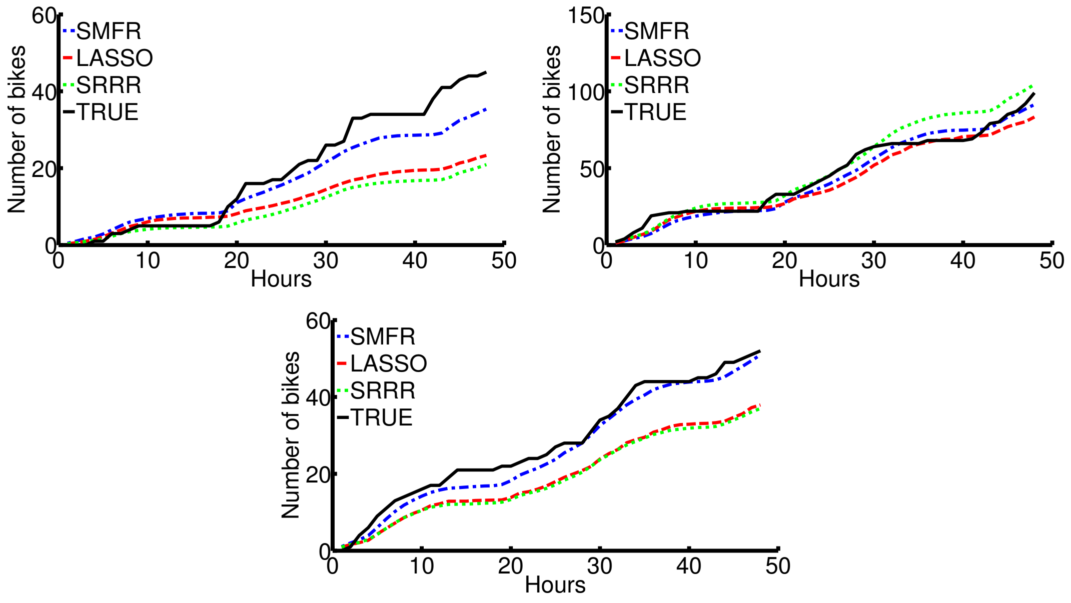
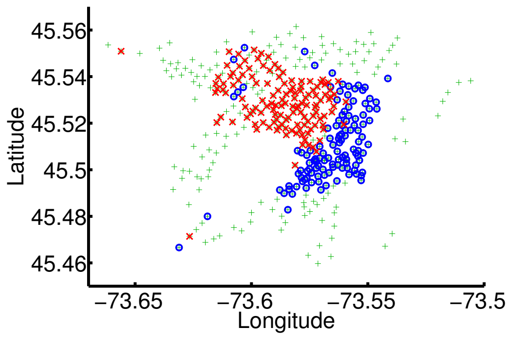


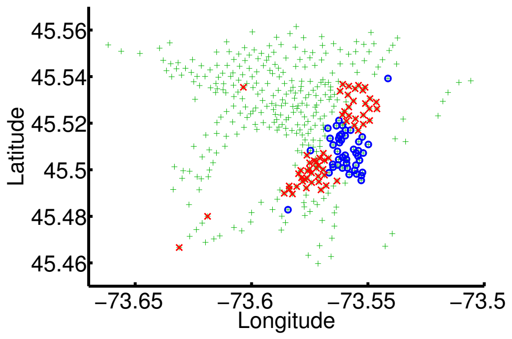
Fig. 4. Comparing fit of different algorithms for three sample cumulative features. Proposed algorithm performs better than others for most stations.

centre, and several festivals occurred there during June. The third factor represents the flow within this central part.

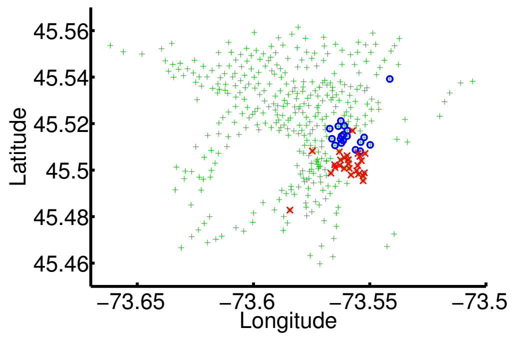
Our model is expected to provide a better fit to the BIXI data since its assumed structure matches the underlying



(a) Factor 1: populated residential areas to downtown



(b) Factor 2: peripheral parts of downtown to central, popular parts



(c) Factor 3: flow inside the central downtown

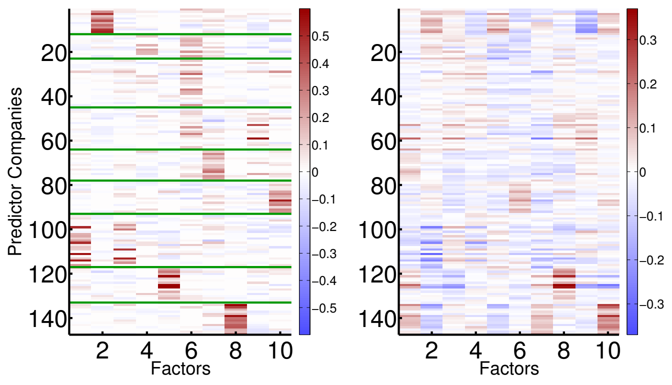
Fig. 5. Three of the factors identified in the BIXI dataset by our algorithm. Green plus signs show the stations, red crosses show the departure features and blue circles show the arrival features of a station chosen in the factor.

structure of bike movements. It is expected that generally, people ride from certain areas of the city to others. The factors capture this aggregated movement (similar to wavelets in the sense of smoothing over multiple individual sites). We expect the factors (matrix $\mathbf{A}$) to be sparse because they capture movement from one region to another, and any stations outside these regions do not participate. We expect the loadings (matrix $\mathbf{B}$) to be sparse because each station should only be predicted by those factors that involve it in terms of arrivals or departures. Therefore, as also confirmed by predictive performance, our proposed sparse, low-dimensional structure is a good fit to the data.

B. S&P 500 stocks

We consider 294 companies from the S&P 500 [38] and collect their daily returns (percentage change in value from one day to the next) between March 1992 and December 2013 for a total of 5500 days. These returns are volatility adjusted using a GARCH model [39] and market adjusted by subtracting the market's average return for each day. We use the global industry classification standard [40] which categorizes all major public companies into 10 sectors. Since there are very few companies in the Telecom sector, we ignore this sector altogether. We divide the companies into two equal groups such that the number of companies from a specific sector is the same in each of the two groups. Our learning task is to predict the daily returns of the second group of companies from the first group. Although this is not a prediction task that would be of most interest in practice (where we want to predict *future* returns), it is a good test to examine the ability of an algorithm to extract the underlying factors and existing structure in the data.

We divide the 5500 day period into 10 intervals of 550 days. For each interval, we choose the first 400 days as the training set (with the last 100 days as validation set) and the last 150 days as the test set. The average and standard deviation of MSE for SMFR is 129.3 (16.2), for SRRR is 129.2 (15.8), and for LASSO is 129.4 (17.1). There is minimal difference in the predictive performance of these algorithms on this dataset; however, we gain significant insight into the nature of data of by looking at the factors created by our algorithm. Figure 6 compares the resulting factors of SRRR and SMFR run over a period of 3000 days (this length is chosen to have clearer factors). We place companies from the same sectors next to each other in predictor and response matrices, and separate them with green lines. The sectors, from top to bottom, are Energy, Materials, Industrials, Consumer Discretionary, Consumer Staples, Health Care, Financials, IT, and Utilities. The proposed algorithm, SMFR, captures the sector factors to a good extent, whereas the structure of factors identified by SRRR is less clear.



(a) SMFR factors (b) SRRR factors

Fig. 6. Factor matrices for SMFR and SRRR.

C. Sparse PCA

We compare our proposed fully sparse PCA with the well-known SPCA of Zou et al. introduced in [30]. We use our BIXI dataset again. Remember that there are 800 features in this dataset. We use the first 200 data points of the dataset to simulate a high-dimensional setting. Thus, our data matrix, $\mathbf{X}$, is 200×800. We compute the first 6 principal components and compare them using two metrics. First, we compute the *adjusted* explained variance; since the sparse principal components are not uncorrelated as in regular PCA, computing their explained variance separately is not correct. Before computing the explained variance of the k'th principal component, regression projection is used to remove its linear dependence to components 1 to $k-1$. See [28] for more details on how to compute the adjusted explained variance.

We also compare the loading sparsity. To have sparse components, we set the regularization parameters such that each component receives contributions from at most 10% of the features. As a benchmark, we also consider the simple thresholding where the values of the regular principal components with absolute value smaller than a threshold are set to zero (here, we keep the top 80).

The results are summarized in Table VI. Compared to SPCA our algorithm explains more variance in the data (a total of 63.9% over the first 6 components compared to 18.6%) with sparser components. Also, we achieve a higher adjusted total variance compared to the simple thresholding (63.9% compared to 28.3%).

	Adjusted Var (%)			loading sparsity ($\|\mathbf{A}\|_{1,1}$)	
PC	SMFR	SPCA	Thresholding	SMFR	SPCA
1	14.5	8.7	15.2	78	80
2	12.4	2.9	4.2	35	79
3	12.3	2.0	2.7	66	80
4	11.5	1.9	2.3	43	78
5	8.0	1.7	2.1	80	77
6	5.2	1.4	1.8	71	79

TABLE VI
ADJUSTED EXPLAINED VARIANCE AND LOADING SPARSITY.

VII. CONCLUSION

We introduced a new sparse multivariate regression algorithm which imposes a low-dimensional structure on the coefficient matrix by first decomposing it into the product of a long factor matrix and a wide loading matrix, and then imposing ℓ_1 penalties on both these matrices. We also provided a formulation to infer the number of latent factors in a more effective way than current techniques. Although the problem formulation leads to a non-convex optimization problem, we showed that the proposed alternating minimization scheme converges and is locally well-posed. Through experiments on simulated and real datasets, we demonstrated that the proposed algorithm is able to exploit the existing structure in the data to improve predictive performance and model selection.

APPENDIX

A. *Proof of Proposition 1*

Proof. For any given $\mathbf{A}$ and $\mathbf{B}$, we have $f(\mathbf{A},\mathbf{B}) \geq 0$. In both minimization steps of Algorithm 1 (i.e., problems (8) and (9)), the value of function f is being decreased. Since f is bounded from below, the sequence of $f(\mathbf{A}_i,\mathbf{B}_i)$ converges to a limit value $f^* \in \mathbb{R}$. □

B. *Proof of Theorem 1*

Proof of part (i):
For problem (8), decomposing $\mathbf{B}$ into its columns, we can rewrite the minimization as follows:

$$\widehat{\mathbf{B}} = \underset{\mathbf{B}}{\operatorname{argmin}} \sum_{j=1}^{q} \left\{ \frac{1}{2}\|\mathbf{Y}^{(j)} - \mathbf{X}\mathbf{A}_i\mathbf{B}^{(j)}\|_2^2 + \lambda_2\|\mathbf{B}^{(j)}\|_1 \right\}, \tag{14}$$

where for any matrix, the superscript of (j) denotes its j'th column. Therefore, problem (8) is equivalent to q separate Lasso problems, one for each response.

Definition 3. A matrix $\mathbf{X}_{n\times p}$ has its columns in *general position* if for any $k<n$, $\mathbf{X}_j \notin \text{aff}\{\mathbf{X}_{i_1},\ldots,\mathbf{X}_{i_{k+1}}\}, \forall j \notin \{i_1,\ldots,i_{k+1}\}$, where $\mathbf{X}_i$ denotes the i'th column of $\mathbf{X}$ and 'aff' denotes the affine span.

In [29], Tibshirani shows that:

Lemma 1 ([29], Lemmas 3 and 4)**.** *Assume that we have the following Lasso problem:*

$$\min_{b} \|\mathbf{y} - \mathbf{H}\mathbf{b}\|_2^2 + \lambda \|\mathbf{b}\|_1.$$

If the columns of $\mathbf{H}$ are in general position, then for any $\mathbf{y}$ and λ, the Lasso solution is unique with probability one. Moreover, if the entries of $\mathbf{H} \in \mathbb{R}^{n\times m}$ are drawn from a continuous probability distribution on $\mathbb{R}^{nm}$, then its columns are in general position and thus, for any $\mathbf{y}$ and λ, the Lasso solution is unique with probability one.

In problem (8), we have $\mathbf{H} = \mathbf{XA}$. If the elements of $\mathbf{X}$ are drawn from a continuous distribution, then its columns are in general position with probability one. In the following Lemma, we show that multiplying $\mathbf{X}$ by $\mathbf{A}$ with full column rank does not change this property and thus, the columns of $\mathbf{H}$ are also in general position. Therefore, according to Lemma 1, the solution of (8) is unique with probability one.

Lemma 2. *If the columns of $\mathbf{X}_{n\times p}$ are in general position with probability one, and $\mathbf{A}_{p\times m}$ has full column rank, then the columns of $\mathbf{XA}$ are also in general position with probability one.*

Proof. Assume that for $\{i_1, \ldots, i_{k+1}\}$ and a j not in that set, we have $\mathbf{XA}_j \in \text{aff}\{\mathbf{XA}_{i_1}, \ldots, \mathbf{XA}_{i_{k+1}}\}$. Then, for some $\alpha_l, l = 1, \ldots, k+1$, we have:

$$\mathbf{X}\left(\mathbf{A}_j + \sum_{l=1}^{k+1} \alpha_l \mathbf{A}_{i_l}\right) = \mathbf{0}. \tag{15}$$

Since $\mathbf{X}$ has its columns in general position, (15) holds with a non-zero probability iff $\mathbf{A}_j + \sum_{l=1}^{k+1} \alpha_l \mathbf{A}_{i_l} = \mathbf{0}$, which is not possible because $\mathbf{A}$ has full column rank. □

Proof of part (ii):
As shown in Lemma 5 of [29], a *sufficient* condition for the solution uniqueness of (9) is that the function $g(\mathbf{U}) = \|\mathbf{Y} - \mathbf{UB}\|_F^2$ be strictly convex (the function argument is defined as $\mathbf{U} = \mathbf{XA}$) and the entries of $\mathbf{X} \in \mathbb{R}^{n\times p}$ are drawn from a continuous probability distribution on $\mathbb{R}^{np}$. It is known that a quadratic function is strictly convex if and only if its Hessian is positive definite. The Hessian of $g(\mathbf{U})$ is equal to $\mathbf{BB}^T \otimes \mathbf{I}_n$, where $\otimes$ denotes the Kronecker product. In other words, the Hessian is block-diagonal with the blocks equal to $\mathbf{BB}^T$. Thus, the Hessian is positive definite if and only if $\mathbf{BB}^T$ is positive definite which is true if and only if $\mathbf{B}$ has full row rank. Thus, the solution of (9) is unique if $\mathbf{B}_{i+1}$ has full row rank.

C. Proof of Theorem 2

Some of the proofs in this subsection exploit the biconvexity of the problem we are addressing and are based on the proofs of similar results in [41].

Proof of part (i):

Definition 4. $\mathcal{A}$ is called the *algorithmic map* of Algorithm 1, if for $\mathbf{C}_1 = (\mathbf{A}_1, \mathbf{B}_1)$ and $\mathbf{C}_2 = (\mathbf{A}_2, \mathbf{B}_2)$ we have:

$$\begin{aligned} \mathbf{C}_2 \in \mathcal{A}(\mathbf{C}_1) \quad \text{iff} \quad & f(\mathbf{A}_1, \mathbf{B}_2) \le f(\mathbf{A}_1, \mathbf{B}), \forall \mathbf{B} \in \mathbb{R}^{m\times q} \\ \text{and} \quad & f(\mathbf{A}_2, \mathbf{B}_2) \le f(\mathbf{A}, \mathbf{B}_2), \forall \mathbf{A} \in \mathbb{R}^{n\times m}. \end{aligned}$$

In other words, $\mathbf{C}_2 \in \mathcal{A}(\mathbf{C}_1)$ iff we can go from $\mathbf{C}_1$ to $\mathbf{C}_2$ in one iteration of Algorithm 1.

Lemma 3. *The algorithmic map $\mathcal{A}$ is closed, i.e., we have:*

$$\left.\begin{array}{ll} \mathbf{C}_i = (\mathbf{A}_i, \mathbf{B}_i) & \textit{and } \lim_{i\to\infty} \mathbf{C}_i = \mathbf{C}^* \\ \mathbf{C}'_i \in \mathcal{A}(\mathbf{C}_i) & \textit{and } \lim_{i\to\infty} \mathbf{C}'_i = \mathbf{C}' \end{array}\right\} \Rightarrow \mathbf{C}' \in \mathcal{A}(\mathbf{C}^*) \tag{16}$$

Proof.

$$\begin{aligned} \mathbf{C}'_i \in \mathcal{A}(\mathbf{C}_i) \Rightarrow f(\mathbf{A}_i, \mathbf{B}'_i) \ &\le \ f(\mathbf{A}_i, \mathbf{B}),\ \forall \mathbf{B} \in \mathbb{R}^{m\times q} \\ \text{and } f(\mathbf{A}'_i, \mathbf{B}'_i) \ &\le \ f(\mathbf{A}, \mathbf{B}'_i),\ \forall \mathbf{A} \in \mathbb{R}^{n\times m} \end{aligned}$$

Since f is continuous, we have:

$$\begin{aligned} f(\mathbf{A}^*, \mathbf{B}') = \lim_{i\to\infty} f(\mathbf{A}_i, \mathbf{B}'_i) &\le \lim_{i\to\infty} f(\mathbf{A}_i, \mathbf{B}) \\ &= f(\mathbf{A}^*, \mathbf{B}) \quad \forall \mathbf{B} \in \mathbb{R}^{m\times q} \\ f(\mathbf{A}', \mathbf{B}') = \lim_{i\to\infty} f(\mathbf{A}'_i, \mathbf{B}'_i) &\le \lim_{i\to\infty} f(\mathbf{A}, \mathbf{B}'_i) \\ &= f(\mathbf{A}, \mathbf{B}') \quad \forall \mathbf{A} \in \mathbb{R}^{n\times m} \end{aligned}$$

Thus, $\mathbf{C}' \in \mathcal{A}(\mathbf{C}^*)$, and $\mathcal{A}$ is closed. □

Lemma 4. *For a given starting point,* $(\mathbf{A}_0, \mathbf{B}_0)$*, the solutions* $\{(\mathbf{A}_i, \mathbf{B}_i)\}_{i\in\mathbb{N}}$ *stay in a bounded set.*

Proof. We have

$$\lambda_1 \|\mathbf{A}_i\|_{1,1} + \lambda_2 \|\mathbf{B}_i\|_{1,1} \le f(\mathbf{A}_i, \mathbf{B}_i) \le f(\mathbf{A}_0, \mathbf{B}_0). \tag{17}$$

Thus, $\{(\mathbf{A}_i, \mathbf{B}_i)\}_{i\in\mathbb{N}}$ stay in a bounded set. □

From Lemmas 3 and 4, we conclude that for a given starting point, the sequence of solutions $\{(\mathbf{A}_i, \mathbf{B}_i)\}_{i\in\mathbb{N}}$ stays in a bounded, closed, and hence compact set and thus has at least one accumulation point.

Proof of part (ii):
Since $\mathbf{C}_{i+1} \in \mathcal{A}(\mathbf{C}_i)$, we have:

$$\begin{aligned} f(\mathbf{A}_i, \mathbf{B}_{i+1}) &\le f(\mathbf{A}_i, \mathbf{B}),\ \forall \mathbf{B} \in \mathbb{R}^{m\times q} \\ \text{and} \quad f(\mathbf{A}_{i+1}, \mathbf{B}_{i+1}) &\le f(\mathbf{A}, \mathbf{B}_{i+1}),\ \forall \mathbf{A} \in \mathbb{R}^{n\times m} \end{aligned}$$

Moreover, if we have $f(\mathbf{C}_{i+1}) = f(\mathbf{C}_i)$, Then:

$$f(\mathbf{A}_{i+1}, \mathbf{B}_{i+1}) = f(\mathbf{A}_i, \mathbf{B}_{i+1}) = f(\mathbf{A}_i, \mathbf{B}_i). \tag{18}$$

Therefore, if the solution of (8) is unique (i.e., $\mathbf{B}_{i+1} = \mathbf{B}_i$), then $\mathbf{C}_i$ is a partial optimum, and if the solution of (9) is unique (i.e., $\mathbf{A}_{i+1} = \mathbf{A}_i$), then $\mathbf{C}_{i+1}$ is a partial optimum.

We know that the sequence $\{\mathbf{C}_i\}_{i\in\mathbb{N}}$ has at least one accumulation point, say $\mathbf{C}^*$. Thus we have a convergent subsequence $\{\mathbf{C}_i\}_{i\in\mathbb{K}}$ with $\mathbb{K} \subset \mathbb{N}$ that converges to $\mathbf{C}^*$. Similarly, $\{\mathbf{C}_{i+1}\}_{i\in\mathbb{K}}$ has an accumulation point, say $\mathbf{C}^+$, to which a subsequence $\{\mathbf{C}_{i+1}\}_{i\in\mathbb{L}}$ with $\mathbb{L} \subset \mathbb{K}$ converges. From Lemma 3 we get $\mathbf{C}^+ \in \mathcal{A}(\mathbf{C}^*)$, and using Proposition 1 we conclude $f(\mathbf{C}^+) = f(\mathbf{C}^*)$. Similarly, if $\mathbf{C}^-$ shows the accumulation point of $\{\mathbf{C}_{i-1}\}_{i\in\mathbb{K}}$, we can show $f(\mathbf{C}^-) = f(\mathbf{C}^*)$.

Combining the results of these two paragraphs, if the solution of (8) is unique, $f(\mathbf{C}^+) = f(\mathbf{C}^*)$ implies that $\mathbf{C}^*$ is partial optimum, and if the solution of (9) is unique, $f(\mathbf{C}^-) = f(\mathbf{C}^*)$ implies that $\mathbf{C}^*$ is partial optimum. Therefore, solution uniqueness of either (8) or (9) implies that $\mathbf{C}^*$ is a partial optimum.

Proof of part (iii)

We prove by contradiction; assume that $\|\mathbf{C}_{i+1} - \mathbf{C}_i\|$ does not converge to zero. Then, for infinitely many $i \in \mathbb{N}$, we have $\|\mathbf{C}_{i+1} - \mathbf{C}_i\| > \delta$ for a $\delta > 0$. Thus, denoting the accumulation points of sequences $\{\mathbf{C}_i\}_{i\in\mathbb{N}}$ and $\{\mathbf{C}_{i+1}\}_{i\in\mathbb{N}}$ respectively by $\mathbf{C}^*$ and $\mathbf{C}^+$, we must have $\|\mathbf{C}^* - \mathbf{C}^+\| > \delta$ and hence $\mathbf{C}^+ \neq \mathbf{C}^*$. On the other hand, since $\mathbf{C}^*$ is a partial optimum and $\mathbf{C}^+ \in \mathcal{A}(\mathbf{C}^*)$, we have:

$$f(\mathbf{A}^*, \mathbf{B}^*) = f(\mathbf{A}^*, \mathbf{B}^+) = f(\mathbf{A}^+, \mathbf{B}^+). \tag{19}$$

Both $\mathbf{A}^*$ and $\mathbf{B}^*$ are full rank and thus, by Theorem 1, $\mathbf{B}^+ = \mathbf{B}^*$, $\mathbf{A}^+ = \mathbf{A}^*$, and consequently, $\mathbf{C}^+ = \mathbf{C}^*$. This is in contradiction with the result of the previous paragraph and hence, $\|\mathbf{C}_{i+1} - \mathbf{C}_i\|$ converges to 0.

D. Proof of Theorem 3

Under the conditions described in part (ii) of Theorem 2, we show that our proposed alternative minimization scheme converges to a partial optimum, say $(\mathbf{A}^*, \mathbf{B}^*)$. Any partial optimum is a stationary point (with zero gradient) which might or might not be a local minimum. For $(\mathbf{A}^*, \mathbf{B}^*)$, the two minimization problems described in (8) and

(9) are in the penalized form. For any given λ_1 and λ_2, there exist t_1 and t_2 such that the following constrained problems are equivalent to those formed in (8) and (9):

$$\begin{aligned}\mathbf{B}^* &= \underset{\mathbf{B}}{\operatorname{argmin}} \quad \frac{1}{2}\|\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{B}\|_{1,1} \le t_2\\ \mathbf{A}^* &= \underset{\mathbf{A}}{\operatorname{argmin}} \quad \frac{1}{2}\|\mathbf{Y}-\mathbf{X}\mathbf{A}\mathbf{B}^*\|_F^2 \quad \text{s.t.} \quad \|\mathbf{A}\|_{1,1} \le t_1\end{aligned}$$

Note that t_2 depends on $(\mathbf{X},\mathbf{Y},\mathbf{A}^*,\lambda_2)$ and t_1 depends on $(\mathbf{X},\mathbf{Y},\mathbf{B}^*,\lambda_1)$. Denoting $S_{\mathbf{A}} = \{\mathbf{A} : \|\mathbf{A}\|_{1,1} \le t_1\}$ and $S_{\mathbf{B}} = \{\mathbf{B} : \|\mathbf{B}\|_{1,1} \le t_2\}$, partial optimality of $(\mathbf{A}^*,\mathbf{B}^*)$ implies:

$$\begin{aligned} & g(\mathbf{A}^*,\mathbf{B}^*) \le g(\mathbf{A}^*,\mathbf{B}), \ \forall \mathbf{B} \in S_{\mathbf{B}}\\ \text{and} \quad & g(\mathbf{A}^*,\mathbf{B}^*) \le g(\mathbf{A},\mathbf{B}^*), \ \forall \mathbf{A} \in S_{\mathbf{A}}\end{aligned}$$

where $g(\mathbf{A},\mathbf{B}) = \frac{1}{2}\|\mathbf{Y}-\mathbf{X}\mathbf{A}\mathbf{B}\|_F^2$.

Lemma 5. *Assume that $(\mathbf{A}^*,\mathbf{B}^*)$ is a partial optimum. Then, for any $\mathbf{A} \in S_{\mathbf{A}}$, we have:*

$$0 \le \text{trace}\{\mathbf{B}^{*T}\boldsymbol{\delta}_{\mathbf{A}}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\delta}_{\mathbf{A}}\mathbf{B}^* - 2(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)^T\mathbf{X}\boldsymbol{\delta}_{\mathbf{A}}\mathbf{B}^*\},$$

where $\boldsymbol{\delta}_{\mathbf{A}} = \mathbf{A}-\mathbf{A}^*$. Moreover, the equality holds iff $g(\mathbf{A},\mathbf{B}^*) = g(\mathbf{A}^*,\mathbf{B}^*)$.

Proof. By the definition of partial optimality, we can write:

$$\begin{aligned}&\|\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*\|_F^2 \le \|\mathbf{Y}-\mathbf{X}\mathbf{A}\mathbf{B}^*\|_F^2 \qquad \forall \mathbf{A} \in S_{\mathbf{A}}\\ &\Rightarrow \text{trace}\{(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)^T(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)\}\\ &\le \text{trace}\{(\mathbf{Y}-\mathbf{X}\mathbf{A}\mathbf{B}^*)^T(\mathbf{Y}-\mathbf{X}\mathbf{A}\mathbf{B}^*)\}\\ &= \text{trace}\{(\mathbf{Y}-\mathbf{X}(\mathbf{A}^*+\boldsymbol{\delta}_{\mathbf{A}})\mathbf{B}^*)^T(\mathbf{Y}-\mathbf{X}(\mathbf{A}^*+\boldsymbol{\delta}_{\mathbf{A}})\mathbf{B}^*)\}\\ &\qquad\qquad \forall \boldsymbol{\delta}_{\mathbf{A}} \ \text{ such that } \ (\mathbf{A}^*+\boldsymbol{\delta}_{\mathbf{A}}) \in S_{\mathbf{A}}\end{aligned}$$

Simple algebra and canceling equal terms from both sides of the inequality yields:

$$\begin{aligned}&0 \le \text{trace}\{\mathbf{B}^{*T}\boldsymbol{\delta}_{\mathbf{A}}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\delta}_{\mathbf{A}}\mathbf{B}^* - 2(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)^T\mathbf{X}\boldsymbol{\delta}_{\mathbf{A}}\mathbf{B}^*\}\\ &\qquad\qquad \forall \boldsymbol{\delta}_{\mathbf{A}} \ \text{ such that } \ (\mathbf{A}^*+\boldsymbol{\delta}_{\mathbf{A}}) \in S_{\mathbf{A}}\end{aligned}$$

The equality holds iff $g(\mathbf{A}^*,\mathbf{B}^*) = g(\mathbf{A},\mathbf{B}^*)$. □

Lemma 6. *Assume that $(\mathbf{A}^*,\mathbf{B}^*)$ is a partial optimum. Then, for any $\mathbf{B} \in S_{\mathbf{B}}$, we have:*

$$0 \le \text{trace}\{\boldsymbol{\delta}_{\mathbf{B}}^T\mathbf{A}^{*T}\mathbf{X}^T\mathbf{X}\mathbf{A}^{*T}\boldsymbol{\delta}_{\mathbf{B}} - 2(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)^T\mathbf{X}\mathbf{A}^*\boldsymbol{\delta}_{\mathbf{B}}\},$$

where $\boldsymbol{\delta}_{\mathbf{B}} = \mathbf{B}-\mathbf{B}^*$. The equality holds iff $g(\mathbf{A}^*,\mathbf{B}) = g(\mathbf{A}^*,\mathbf{B}^*)$.

The proof is similar to that of Lemma 5.

Proof of Theorem:
Assume that $\mathbf{A} \in S_{\mathbf{A}}$ and $\mathbf{B} \in S_{\mathbf{B}}$. Also, assume that $\|\boldsymbol{\delta}_{\mathbf{A}}\|_{\infty,\infty} = \delta_1$ and $\|\boldsymbol{\delta}_{\mathbf{B}}\|_{\infty,\infty} = \delta_2$, where $\boldsymbol{\delta}_{\mathbf{A}} = \mathbf{A}-\mathbf{A}^*$, $\boldsymbol{\delta}_{\mathbf{B}} = \mathbf{B}-\mathbf{B}^*$, and for any matrix $\mathbf{M}$, $\|\mathbf{M}\|_{\infty,\infty} = \max_{ij} |\mathbf{M}_{i,j}|$. We prove that $(\mathbf{A}^*,\mathbf{B}^*)$ is a local minimum by showing that

$$g(\mathbf{A}^*,\mathbf{B}^*) \le g(\mathbf{A},\mathbf{B}) \qquad \text{as} \ \ \delta \to 0^+, \tag{20}$$

where $\delta = \max\{\delta_1,\delta_2\}$.

We have:

$$\begin{aligned}2g(\mathbf{A},\mathbf{B}) &= \text{trace}\Big\{(\mathbf{Y}-\mathbf{X}(\mathbf{A}^*+\boldsymbol{\delta}_\mathbf{A})(\mathbf{B}^*+\boldsymbol{\delta}_\mathbf{B}))^T \\ &\qquad (\mathbf{Y}-\mathbf{X}(\mathbf{A}^*+\boldsymbol{\delta}_\mathbf{A})(\mathbf{B}^*+\boldsymbol{\delta}_\mathbf{B}))\Big\} \\ &= \underbrace{\text{trace}\left\{(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)^T(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)\right\}}_{I} \\ &+ \underbrace{\text{trace}\left\{\mathbf{B}^{*T}\boldsymbol{\delta}_\mathbf{A}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\delta}_\mathbf{A}\mathbf{B}^* - 2(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)^T\mathbf{X}\boldsymbol{\delta}_\mathbf{A}\mathbf{B}^*\right\}}_{II} \\ &+ \underbrace{\text{trace}\left\{\boldsymbol{\delta}_\mathbf{B}^T\mathbf{A}^{*T}\mathbf{X}^T\mathbf{X}\mathbf{A}^*\boldsymbol{\delta}_\mathbf{B} - 2(\mathbf{Y}-\mathbf{X}\mathbf{A}^*\mathbf{B}^*)^T\mathbf{X}\mathbf{A}^*\boldsymbol{\delta}_\mathbf{B}\right\}}_{III} \\ &+ \underbrace{O(\delta^2)}_{IV}\end{aligned}$$

By the definition of g, we have $I = 2g(\mathbf{A}^*,\mathbf{B}^*)$. According to Lemmas 5 and 6: $II \geq 0$ and $III \geq 0$. If $II+III = 0$, then $II = 0$ and $III = 0$. From Lemma 5, $II = 0$ gives $g(\mathbf{A}^*,\mathbf{B}^*) = g(\mathbf{A},\mathbf{B}^*)$, and from Lemma 6, $III = 0$ implies $g(\mathbf{A}^*,\mathbf{B}) = g(\mathbf{A}^*,\mathbf{B}^*)$. Since either $\mathbf{A}^*$ or $\mathbf{B}^*$ is full rank, we get $g(\mathbf{A}^*,\mathbf{B}^*) = g(\mathbf{A},\mathbf{B})$, and thus, inequality (20) holds (with the two sides being equal).

So, we assume $II + III > 0$. In this case, $II + III = \Omega(\delta)$ unless either $\mathbf{Y} = \mathbf{X}\mathbf{A}^*\mathbf{B}^*$, which automatically gives (20), or $\mathbf{A}^* = \mathbf{0}, \mathbf{B}^* = \mathbf{0}$, which cannot happen by the assumption that one of these matrices should be full rank. Therefore as $\delta \to 0^+$, $II + III + IV \geq 0$ and we get $g(\mathbf{A},\mathbf{B}) > g(\mathbf{A}^*,\mathbf{B}^*)$.

References

[1] C.-F. Lee and J. Lee, *Handbook of quantitative finance and risk management*. Springer Science & Business Media, 2010.

[2] L. R. Monteiro, "Multivariate regression models and geometric morphometrics: the search for causal factors in the analysis of shape," *Systematic Biology*, pp. 192–199, 1999.

[3] S. Ranhao, Z. Baiping, and T. Jing, "A multivariate regression model for predicting precipitation in the daqing mountains," *Mountain Research and Development*, vol. 28, no. 3, pp. 318–325, 2008.

[4] S. Wold, M. Sjöström, and L. Eriksson, "PLS-regression: a basic tool of chemometrics," *Chemometrics and Intelligent Laboratory Systems*, vol. 58, no. 2, pp. 109–130, 2001.

[5] L. Harrison, W. D. Penny, and K. Friston, "Multivariate autoregressive modeling of fmri time series," *NeuroImage*, vol. 19, no. 4, pp. 1477–1491, 2003.

[6] R. J. Tibshirani, "Regression shrinkage and selection via the lasso: a retrospective," *J. Royal Statistical Society: Series B*, vol. 73, no. 3, pp. 273–282, 2011.

[7] M. J. Wainwright, "Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (LASSO)," *IEEE Trans. Info. Theory*, vol. 55, no. 5, pp. 2183–2202, 2009.

[8] B. A. Turlach, W. N. Venables, and S. J. Wright, "Simultaneous variable selection," *Technometrics*, vol. 47, no. 3, pp. 349–363, 2005.

[9] C.-H. Zhang and J. Huang, "The sparsity and bias of the lasso selection in high-dimensional linear regression," *The Annals of Statistics*, vol. 36, no. 4, pp. 1567–1594, 2008.

[10] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. Royal Statistical Society, Series B*, vol. 68, no. 1, pp. 49–67, 2006.

[11] G. Obozinski, M. J. Wainwright, M. I. Jordan *et al.*, "Support union recovery in high-dimensional multivariate regression," *The Annals of Statistics*, vol. 39, no. 1, pp. 1–47, 2011.

[12] A. Jalali, S. Sanghavi, C. Ruan, and P. K. Ravikumar, "A dirty model for multi-task learning," in *Proc. Advances in Neural Information Processing Systems*, 2010, pp. 964–972.

[13] J. Peng, J. Zhu, A. Bergamaschi, W. Han, D.-Y. Noh, J. R. Pollack, and P. Wang, "Regularized multivariate regression for identifying master predictors with application to integrative genomics study of breast cancer," *The Annals of Applied Statistics*, vol. 4, no. 1, pp. 53–77, 2010.

[14] J. Chen, J. Zhou, and J. Ye, "Integrating low-rank and group-sparse structures for robust multi-task learning," in *Proc. ACM Int. Conf. Knowledge Discovery and Data Mining*, 2011, pp. 42–50.

[15] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky, "Rank-sparsity incoherence for matrix decomposition," *SIAM J. Optimization*, vol. 21, no. 2, pp. 572–596, 2011.

[16] M. Pourahmadi, *High-Dimensional Covariance Estimation: With High-Dimensional Data*, ser. Wiley Series in Probability and Statistics. Wiley, 2013.

[17] L. Chen and J. Z. Huang, "Sparse reduced-rank regression for simultaneous dimension reduction and variable selection," *J. American Statistical Association*, vol. 107, no. 500, pp. 1533–1545, 2012.

[18] H. Chun and S. Keleş, "Sparse partial least squares regression for simultaneous dimension reduction and variable selection," *J. Royal Statistical Society: Series B*, vol. 72, no. 1, pp. 3–25, 2010.

[19] M. Yuan, A. Ekici, Z. Lu, and R. Monteiro, "Dimension reduction and coefficient estimation in multivariate linear regression," *J. Royal Statistical Society, Series B*, vol. 69, no. 3, pp. 329–346, 2007.

[20] G. C. Reinsel and R. P. Velu, *Multivariate reduced-rank regression*. Springer, 1998.

[21] R. Velu and G. C. Reinsel, *Multivariate reduced-rank regression: theory and applications*. Springer, 1998.

[22] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *Machine Learning*, vol. 73, no. 3, pp. 243–272, 2008.

[23] S. Ji and J. Ye, “An accelerated gradient method for trace norm minimization,” in *Proc. Int. Conf. Machine Learning*. ACM, 2009, pp. 457–464.

[24] X. Zhang, D. Schuurmans, and Y.-l. Yu, “Accelerated training for matrix-norm regularization: A boosting approach,” in *Proc. Advances in Neural Information Processing Systems*, 2012, pp. 2906–2914.

[25] A. Mukherjee and J. Zhu, “Reduced rank ridge regression and its kernel extensions,” *Statistical Analysis and Data Mining*, vol. 4, no. 6, pp. 612–622, 2011.

[26] A. Kumar and H. Daume, “Learning task grouping and overlap in multi-task learning,” *arXiv:1206.6417*, 2012.

[27] P.-H. Chou, P.-H. Ho, and K.-C. Ko, “Do industries matter in explaining stock returns and asset-pricing anomalies?” *J. Banking & Finance*, vol. 36, no. 2, pp. 355–370, 2012.

[28] H. Zou, T. Hastie, and R. Tibshirani, “Sparse principal component analysis,” *J. Computational and Graphical Statistics*, vol. 15, no. 2, pp. 265–286, 2006.

[29] R. J. Tibshirani, “The lasso problem and uniqueness,” *Electronic J. Statistics*, vol. 7, pp. 1456–1490, 2013.

[30] H. Zou, “The Adaptive Lasso and Its Oracle Properties,” *J. American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006.

[31] A. J. Rothman, E. Levina, and J. Zhu, “Sparse multivariate regression with covariance estimation,” *J. Computational and Graphical Statistics*, vol. 19, no. 4, pp. 947–962, 2010.

[32] J. Mairal. (2014) SPAMS: a SPArse Modeling Software, v2.5. [Online]. Available: http://spams-devel.gforge.inria.fr/

[33] M. Schmidt, G. Fung, and R. Rosales. (2009) Optimization methods for L1-regularization. University of British Columbia. [Online]. Available: http://www.cs.ubc.ca/ schmidtm/Software/L1General.html

[34] *SPLS: Sparse Partial Least Squares Regression and Classification (R package)*. [Online]. Available: https://cran.r-project.org/web/packages/spls/index.html

[35] *RemMap: Regularized Multivariate Regression for Identifying Master Predictors (R package)*. [Online]. Available: https://cran.r-project.org/web/packages/remMap/index.html

[36] J. Zhou, J. Chen, and J. Ye. (2011) Malsar: Multi-task learning via structural regularization. Arizona State University. [Online]. Available: http://www.public.asu.edu/ jye02/Software/MALSAR

[37] M. Kharratzadeh and M. Coates, “Sparse multivariate factor regression,” 2015, technical Report, McGill University, available at http://networks.ece.mcgill.ca/pubs.

[38] Standard and Poor’s. (2013, January) S&P 500 factsheet. [Online]. Available: http://bit.ly/1sQAV8M

[39] C. Francq and J.-M. Zakoian, *GARCH models: structure, statistical inference and financial applications*. John Wiley & Sons, 2011.

[40] GICS. (2013, February) MSCI-Barra GICS Tables. [Online]. Available: http://www.msci.com/products/indexes/sector/gics/

[41] J. Gorski, F. Pfeuffer, and K. Klamroth, “Biconvex sets and optimization with biconvex functions: a survey and extensions,” *Mathematical Methods of Operations Research*, vol. 66, no. 3, pp. 373–407, 2007.