Sparse generalized principal component analysis for large-scale applications beyond Gaussianity

Qiaoya Zhang[†] and Yiyuan She[*,†]

Principal Component Analysis (PCA) is a dimension reduction technique. It produces inconsistent estimators when the dimensionality is moderate to high, which is often the problem in modern large-scale applications where algorithm scalability and model interpretability are difficult to achieve, not to mention the prevalence of missing values. While existing sparse PCA methods alleviate inconsistency, they are constrained to the Gaussian assumption of classical PCA and fail to address algorithm scalability issues. We generalize sparse PCA to the broad exponential family distributions under high-dimensional setup, with built-in treatment for missing values. Meanwhile, we propose a family of iterative sparse generalized PCA (SG-PCA) algorithms such that despite the non-convexity and non-smoothness of the optimization task, the loss function decreases in every iteration. In terms of ease and intuitive parameter tuning, our sparsity-inducing regularization is far superior to the popular Lasso. Furthermore, to promote overall scalability, accelerated gradient is integrated for fast convergence, while a progressive screening technique gradually squeezes out nuisance dimensions of a large-scale problem for feasible optimization. High-dimensional simulation and real data experiments demonstrate the efficiency and efficacy of SG-PCA.

AMS 2000 subject classifications: Primary 62H25, 62J12; secondary 62H12.
Keywords and phrases: Sparsity, Low rank estimation, Principal component analysis, Generalized linear models, Non-convex optimization, Missing values, Variable screening.

1. INTRODUCTION

Suppose an $n \times p$ matrix $\boldsymbol{X}$ represents a data set with n observations on p variables centered column-wise. In modern statistical applications, both n and p can be quite large especially with $p \gg n$. Principal Component Analysis (PCA) is a well-known and popular dimension reduction technique. It *sequentially* searches r ($r \ll p$) leading principal directions along which the projected data points have maximal variance. Equivalently, it can also be realized *jointly* by solving

a multivariate low-rank matrix approximation problem

$$(1) \qquad \min_{\boldsymbol{B}:rank(\boldsymbol{B}) \leq r} \|\boldsymbol{X} - \boldsymbol{B}\|_F^2,$$

with the optimal $\boldsymbol{B}$ given by $\boldsymbol{X} \boldsymbol{V}_r \boldsymbol{V}_r^T$, where $\boldsymbol{V}_r = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_r]$ are formed by the top r right singular vectors of $\boldsymbol{X}$ [8]. The principal components (PCs) $\{\boldsymbol{z}_1, \cdots, \boldsymbol{z}_r\}$ are then given by $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{V}$ and $\boldsymbol{v}_i$'s are called the principal loading vectors. Clearly, PCs are also the eigenvectors of the sample covariance matrix, which further explains its power in illustrating the variability within the multivariate data in a lower-dimensional space.

There are, however, many characteristics to modern large-scale data that regular PCA finds inappropriate to handle. Three specific challenges we want to address in this article are: a) the prevalence of non-Gaussian Data, b) the curse of dimensionality, as well as c) the existence of missing entries in data arrays.

1.1 Modern challenges

Non-Gaussianity PCA is often employed without addressing any parametric assumption on the original data set $\boldsymbol{X}$, however, its criterion inherently assumes a Gaussian distribution: the squared-error loss function (1) does not make the best sense with, for example, the misclassification error associated with categorical data. The sample covariance matrix, whose eigenvectors are computed as principal components, does not capture all kinds of association either. When the data does not follow a Gaussian distribution, an alternative loss functions might be more proper in measuring the affinity between $\boldsymbol{X}$ and its low-rank estimate. There are a lot of real-world motivations: Netflix's user rating system is typical of a categorical data type, SNPs data denotes mutation by binary coding, and spam email detection often examines the number of times a flagged word appears.

Curse of dimensionality Besides the jeopardy of soundness PCA faces when prompted with an extended range of data types, the method itself also encounters theoretical, practical, and computational challenges when the dimensionality p is high.

Theoretically, the estimated PCs prove to be *inconsistent* if $\lim_{n\to\infty} p/n$ does not go to zero [11, 21, 18], that is, when the number of variables p is comparable or larger than n, which is often the case in applications related to network,

genetics and so on. This conclusion manifests a surprising and controversial result: the statistical accuracy of PCA is highly restricted to $p \ll n$ despite that its incentive stems from the challenge of high dimension itself.

What's worse, when the squared signal-to-noise ratio is less than the converging constant of p/n, the estimated principal components can be asymptotically orthogonal to the true ones, essentially containing no authentic information at all! Practically, since each PC $\boldsymbol{z}_i = \sum_{j=1}^{p} \boldsymbol{x}_j v_{ji}$ still utilizes *all* p dimensions in $\boldsymbol{X}$ with the loading matrix as weights, there is essentially no hope of interpretability if the entries in $\boldsymbol{V}$ are mostly nonzero. Therefore, the ideal model in terms of both theoretical consistency and practical selective property, calls for a parsimonious representation of the original dimensions, which corresponds to enforcing sparse nonzero entries in the loading matrix.

The big data era exacerbates the curse of dimensionality from the computational perspective. Even when p is moderately high, many existing algorithms find it difficult to handle such computation complexity. Fast and feasible algorithms need to be developed without sacrificing accuracy.

Missing values One might argue that missing value is not as modern a challenge as its previous peers, but it is for sure a problem that evolves with modern data formats. The Netflix's MovieLens, for example, has a missing percentage of up to 99% [13] and a pattern of missingness distinct from traditional problem. If these entries were to be removed or imputed in a conventional fashion, the training of the model is likely to suffer from a great deal of inaccuracy due to missingness. Under such circumstances, it is much desired to develop a novel approach which makes no assumptions on the missing entries and integrate them into the optimization problem itself.

With these challenges in mind, we examine the existing literature for ideas explored. On the topic sparse PCA, there are quite a few iterative estimating algorithms such as [12], [30], [36], [40], among others, where each column of the loading matrix is sequentially retrieved. Despite of its simple formulation and the resulting nested spaces, sequential techniques lack joint optimality and joint orthogonality when multiple features are desired. More importantly, these methodologies are proposed under Gaussian assumption. Generalizations of PCA to fit various non-Gaussian data types are seen mostly in the machine learning community targeted on specific applications, such as [14], [10], [2], [23], [7], [17]. [5] generalizes PCA to the exponential family. However, it discusses the rank-one case only under a large-n background. In the statistical literature, perhaps the most similar work to ours is logisticPCA by [16] and [15]—both utilizing ℓ_1 regularization for sparsity, whose parameter is tuned by a greedy sequential grid search which may result in a suboptimal estimator. The criterion is established for binary data in specific, a subset of our target. Missing values are discussed, but proposed to handle in a conventional imputation manner. More details of logisticPCA will be given in Section 4.1. None of the aforementioned works addresses algorithm scalability when p is high and computationally demanding. While sparsity lies in the heart of these methods, it is often enforced rank by rank such that no overall selection power is guaranteed unless the rank level is very low. In sum, there is a scarcity of existing literature that compares to the scope and scale of our work.

1.2 An outline

We propose a novel low-rank multivariate data approximation method called SG-PCA. SG-PCA establishes a universal framework for any exponential family distribution under the high-dimensional setting. We launch from a low-rank data approximation perspective and propose a joint rank-r algorithm with built-in treatment for missing values. To achieve this goal, we solve a non-smooth optimization criterion with non-convex rank, sparsity and orthogonality regularization. Matrix decomposition eases the low-rank condition in the optimization criterion. Element-wise and group-wise sparsity constraints are studied and differentiated. A Stiefel manifold optimization problem is simplified by an iterative process with some MM (majorization minimization) flavor. As a result, we are able to recover the loading space with rank reduction and dimension selection all in one step, while rectifying the inconsistency issue brought by high dimensionality. Our formulation treats missing values inherently as a part of the criterion. For fast computation, accelerated gradient methods are incorporated in SG-PCA to promote algorithm efficiency. Last but not least, we develop what is essential to the feasibility of large-scale applications—a progressive screening scheme which throws away nuisance dimensions gradually, providing a smaller problem size each round.

The rest of the paper is organized as below. In Section 2, we formulate the criterion for non-Gaussian data with missing values and describe the necessity and approach to enforce sparse loading vectors. Section 3 gives the SG-PCA algorithms to solve the rank-r problem under both sparsity and orthogonality constraints and incorporate accelerated gradient methods, a line search scheme, and a powerful progressive screening strategy to enhance algorithm efficiency and scalability. Simulation studies and real data applications are given in Section 4.

2. SPARSE GENERALIZED PCA

This section is dedicated to formulate and layout the SG-PCA procedure in every detail. We start with deriving a suitable criterion, proceed to discussion of the penalties and constraints used for regularization and the missing data issues.

2.1 The regularized log-likelihood criterion

Suppose our data is stored in an $n \times p$ matrix $\boldsymbol{X}$ that follows some underlying distribution. Low-rank data approxi-

Table 1. A list of functions of interest with respect to various distributions

Distribution	$g(\mu)$	$b'(\theta) = g^{-1}(\theta)$	$b(\theta)$	$b''(\theta)$
Gaussian	μ	θ	$\frac{\theta^2}{2}$	1
Bernoulli Binomial Multinomial	$\log\frac{\mu}{1-\mu}$	$\frac{e^\theta}{1+e^\theta}$	$\log(1+e^\theta)$	$\frac{1}{e^\theta+e^{-\theta}+1}$
Poisson	$\log\mu$	e^θ	e^θ	e^θ
Exponential Gamma	$-\frac{1}{\mu}$	$-\frac{1}{\theta}$	$-\log(|\theta|)$	$\frac{1}{\theta^2}$

mation methods search for the projection $\boldsymbol{B}$ of the data to an r-dimensional ($r < p$) subspace such that the loss of information $L(\boldsymbol{X}, \boldsymbol{B})$ is minimized. Under the Gaussian assumption, sparse PCA finds $\boldsymbol{B}$ by maximizing a sum-of-squares criterion $\|\tilde{\boldsymbol{X}} - \tilde{\boldsymbol{u}}\tilde{\boldsymbol{v}}^T\|_F^2 + P_\lambda(\boldsymbol{v})$ sequentially, the data matrix deflated each time getting a pair of $(\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{v}})$ [30]. The sparsity regularization is enforced on $\tilde{\boldsymbol{v}}$ to obtain an estimated loading vector. Therefore, sparse PCA has a closed-form update under the simple quadratic loss.

However, our data of interest encompass all distributions in the exponential family, for which the sum-of-squares criterion may fail. The negative log-likelihood is an intuitive alternative for such non-Gaussian data types.

This extension of loss function resembles that of the generalized linear models (GLMs). GLMs deal with non-Gaussian response variables for the lack of linear relationship between the predictors and the response. Let $\boldsymbol{\mu} = E(\boldsymbol{X})$ and $g(\cdot)$ denote the canonical link function, then $\boldsymbol{\Theta} = g(\boldsymbol{\mu})$ is the natural parameter. We assume that all the x_{ij}'s are independent given the low-rank structure in $\boldsymbol{\Theta}$ and write the negative log-likelihood function in matrix form:

$$(2) \quad -l(\boldsymbol{\Theta}|\boldsymbol{X}) = -\langle \boldsymbol{X}, \boldsymbol{\Theta}\rangle + \langle \mathbf{1}_n\mathbf{1}_p^T, b(\boldsymbol{\Theta})\rangle - \log(h(\boldsymbol{X})),$$

where the matrix component-wise inner product is defined as $\langle \boldsymbol{A}, \boldsymbol{B}\rangle = tr(\boldsymbol{A}^T\boldsymbol{B})$, the term $\log(h(\boldsymbol{X}))$ is treated as a constant, thus omitted during optimization. Note that the derivative of the log partition function $b'(\cdot)$ is equal to the inverse of the canonical link function $g^{-1}(\cdot)$. The negative log-likelihood of a Gaussian $\boldsymbol{X}$ returns equation (1), the squared-error loss. Table 1 gives a list of functions of interest with respect to some commonly used exponential family distributions, where μ and θ stand for the mean and natural parameter in general.

To represent the natural parameter in a low-rank fashion, we rewrite $\boldsymbol{\Theta} = \mathbf{1}_n\boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T$, where $\mathbf{1}_n$ is a column vector of 1's and $\boldsymbol{\alpha}$ the intercept vector. Here we require $\boldsymbol{V} \in \mathbb{O}^{n\times r} = \{\boldsymbol{A} \in \mathbb{R}^{p\times r}|\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}_{r\times r}\}$. $\boldsymbol{S} \in \mathbb{R}^{p\times r}$ gives the principal loading matrix but does not necessarily have orthogonal columns. Instead, we require $\boldsymbol{S}$ to be sparse. Such a $\boldsymbol{V}\boldsymbol{S}^T$ setup prepares the objective function for regularization ease.

The objective function minimizes the regularized negative log-likelihood:

$$(3)$$
$$-\langle \boldsymbol{X}, \mathbf{1}_n\boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T\rangle + \langle \mathbf{1}_n\mathbf{1}_p^T, b(\mathbf{1}_n\boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T)\rangle + P(\boldsymbol{S}; \lambda)$$
$$\text{subject to } \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{r\times r}$$

where $P(\boldsymbol{S}; \lambda)$ denotes a sparsity-inducing regularization with λ as its parameter.

This criterion is applicable to a variety of large-scale applications for its wide assumptions on distributions. The entire low-rank approximation can also be derived as a whole, preserving joint optimality as compared to the sequential fashion.

2.2 $\boldsymbol{S}$ for sparsity

Since $\boldsymbol{S}$ denotes the principal loading matrix, the PCs are written as $\boldsymbol{z}_i = \boldsymbol{X}\boldsymbol{s}_i$. When the dimension p is high, relative to the sample size n, it is necessary for $\boldsymbol{S}$ to have sparse nonzero entries to ensure consistency [11]. Furthermore, the scope of dimension reduction can and should be improved by the addition of sparsity constraints. Dimension reduction traditionally refers to *rank* reduction in the context of PCA. Geometrically it is a projection of the observed data points to a lower r-dimensional subspace. What promotes model parsimony is the elimination of nuisance dimensions along with rank reduction, leading to sparse representation in the PCs.

On the other hand, enforcing element-wise sparsity does not yield the smallest subset of variables for principal components unless r is extremely low. To remove an entire column of $\boldsymbol{X}$ in the construction of all PCs given by $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{S}$, introducing *row-sparsity* in $\boldsymbol{S}$, i.e., $P(\boldsymbol{S}; \lambda) = \sum_{i=1}^p P(\|\boldsymbol{s}_i\|_2; \lambda)$ (with some abuse of notation), would do the trick. It is easy to see that the row-sparsity in $\boldsymbol{S}$ corresponds to the column sparsity in $\boldsymbol{B} = \boldsymbol{V}\boldsymbol{S}^T$, verified due to the orthoganality of $\boldsymbol{V}$. It is essential for efficiency and selectivity of the algorithm and for the overall interpretability of the results. We will blend the two types of sparsity on the loading matrix for fast computation, with row sparsity employed in a screening step to reduce p to some dimension d ($d < p$) prior to the element-wise sparsity pursuit in each individual loading vector.

There are abundant choices of sparsity-inducing penalty functions $P(\cdot; \cdot)$ in $\sum_{i,j} P(|s_{ij}|; \lambda)$ and $\sum_{i=1}^p P(\|\boldsymbol{s}_i\|_2; \lambda)$. The ℓ_1 penalty [31] is most popular among the sparse PCA literature, but it suffers from inconsistency and biased estimation [39, 37] especially when predictors are correlated. To alleviate those issues, other non-convex approximations of the ideal non-convex ℓ_0 ($\|\boldsymbol{S}\|_0 = \sum_{i,j} 1_{s_{ij}\neq 0}$) penalty such as ℓ_p ($0 < p < 1$) penalty, SCAD [9] and capped ℓ_1 [38] are proposed. However, we propose to use the ℓ_0 itself for its appealing properties in sparsity promotion, because it can directly limit the cardinality of nonzero elements/rows in the

loading matrix hence encourage accurate selection. More-over, the **constraint** form of ℓ_0 is yet preferred over the penalty form due to its tuning ease. While the grid search process of λ is traditionally cumbersome, tuning for the constraint form $\sum_{i,j} 1_{s_{ij} \neq 0}/(pr) \leq q_e$ is intuitive and easy: q_e serves as the upper bound for percent of nonzero entries in the loading matrix, thus the selection accuracy should remain sound as long as parameter q_e is larger than the true value. In fact, it corresponds to a rank-constrained screening problem described in [27].

Note that orthogonality is only imposed on $\boldsymbol{V}$. While this makes $\boldsymbol{V}\boldsymbol{S}^T$ a general representation for any matrix $\boldsymbol{B}$ of rank no higher than r—a convenient translation of the non-convex low-rank constraint, it implies that the obtained sparse PCs are not decorrelated as in ordinary PCA. However, the loss of orthogonality is generally considered a price that sparse PCA has to pay [40, 30, 36].

2.3 Handling missing values

We propose a masking approach for efficient handling of missing entries. Conventionally, it is a common practice to impute or simply remove missing entries before training the model, both holding assumptions on the missing entries thereby introducing additional inaccuracy into the training process. On the contrary, we do not assume any prior knowledge on the missing entries, rather, we mask them as unknown information such that their contribution to the loss function is not taken into consideration. This is made possible by approximating the low-rank and sparse structure of the data, such that masking some amount of missing values does not interfere with matrix recovery. Perhaps it is interesting to note that our masking technique for missing data is deeply connected and naturally applicable to matrix completion type of problems (see examples in [4]), where only an extraordinary small fraction of data is observed and imputation of the missing values is precisely the goal. Instead of writing the problem as a summation in a subset, we will develop a *multivariate* approach that utilizes matrix-wise operations, making it easier in implementation. The masking method introduces minimal cost to the computational algorithm even as the dimension of the problem is rocket high, and it is entirely integrated into the estimation process.

Let Ω denote the index set of all available observations, intuitively the optimization criterion (3) is only evaluated when $\boldsymbol{X} \in \Omega$. However, for computation efficiency and ease in analysis, we prefer the following formulation. Define the masking matrix $\boldsymbol{H} = [h_{ij}]$ such that

$$(4) \qquad h_{ij} = \begin{cases} 1 & \text{if } (i,j) \in \Omega \\ 0 & \text{if } (i,j) \in \Omega^C \end{cases}$$

Throughout the paper we use the Hadamard Product $\circ$ to denote the element-wise matrix multiplication: $\forall \boldsymbol{X}$ where

$dim(\boldsymbol{X}) = dim(\boldsymbol{H})$,

$$\boldsymbol{H} \circ \boldsymbol{X} = [h_{ij} \cdot x_{ij}].$$

From here on, our attention shifts to the loss function f^m below:

$$(5) \qquad -\langle \boldsymbol{H} \circ \boldsymbol{X}, \boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T \rangle + \langle \boldsymbol{H}, b(\boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T) \rangle$$

The gradients of the masked loss function can be calculated (details omitted):

$$(6)$$
$$\begin{cases} \boldsymbol{G_S} = -(\boldsymbol{H} \circ \boldsymbol{X})^T \boldsymbol{V} + (\boldsymbol{H} \circ g^{-1}(\boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T))^T \boldsymbol{V} \\ \boldsymbol{G_V} = -(\boldsymbol{H} \circ \boldsymbol{X})\boldsymbol{S} + (\boldsymbol{H} \circ g^{-1}(\boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T))\boldsymbol{S} \\ \boldsymbol{G_\alpha} = -(\boldsymbol{H} \circ \boldsymbol{X})^T \boldsymbol{1}_n + (\boldsymbol{H} \circ g^{-1}(\boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T))^T \boldsymbol{1}_n \end{cases}$$

For simplicity, we use $\boldsymbol{X} = \boldsymbol{H} \circ \boldsymbol{X}$ hereafter to demonstrate the masked observed data matrix.

3. AN ITERATIVE ALGORITHM

3.1 A surrogate function

A natural idea to solve optimization problem 3 is to utilize a block coordinate descent (BCD) [32] algorithm where $\boldsymbol{\alpha}$, $\boldsymbol{S}$ and $\boldsymbol{V}$ are updated alternatively. While $\boldsymbol{\alpha}^{[k]}$ has a closed-form solution when all entries are observed and the distribution is Gaussian, in the existence of missing values or under GLM setting, $\boldsymbol{\alpha}^{[k]}$ has no explicit form.

The bigger challenge lies in optimizing $\boldsymbol{V}$ while holding $\boldsymbol{\alpha}$ and $\boldsymbol{S}$ fixed. Although the objective function is smooth in $\boldsymbol{V}$, the unitary constraint $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{p \times p}$ is non-convex and non-smooth. One may treat the update of $\boldsymbol{V}$ as a constrained optimization problem with quite a few Lagrangian multipliers, but it is awkward and slow in computation. The problem is better phrased as a Stiefel manifold optimization one, for which packages are already available [35, 28]. That is what we first try. We have implemented and tested the manifold optimization algorithm for SG-PCA only to find it to be a valid yet rather expensive method when the problem size is big. With no need to call any external package we develop a new algorithm with some MM (majorization minimization) flavor in its employment of a surrogate function.

Consider the optimization problem in the element-wise sparse case as an instance. We minimize the objective function

$$(7)$$
$$\min_{\boldsymbol{\alpha}, \boldsymbol{V}, \boldsymbol{S}} f = -\langle \boldsymbol{X}, \boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T \rangle + \langle \boldsymbol{H}, b(\boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T) \rangle$$
$$+ P(\boldsymbol{S}; \lambda), \text{ subject to } \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{p \times p},$$

where $l(\boldsymbol{\alpha}, \boldsymbol{V}, \boldsymbol{S}) = -\langle \boldsymbol{X}, \boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T \rangle + \langle \boldsymbol{H}, b(\boldsymbol{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V}\boldsymbol{S}^T) \rangle$ denotes the loss.

Solving this non-quadratic loss function with a non-convex orthogonality constraint is rather difficult, for the

prerequisites of the convenient candidate approaches such as Procrustes rotation are deprived. We seek to utilize a quadratic loss through deriving a surrogate function.

Concretely, given the $(k-1)^{th}$ step estimates $\boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}$, define $\boldsymbol{\Theta}^{[k-1]}$ as a function of $\boldsymbol{\alpha}^{[k-1]}, \boldsymbol{V}^{[k-1]}, \boldsymbol{S}^{[k-1]}$:

$$\boldsymbol{\Theta}^{[k-1]} = \mathbf{1}_n \boldsymbol{\alpha}^{[k-1]T} + \boldsymbol{V}^{[k-1]} \boldsymbol{S}^{[k-1]T}.$$

Linearization will be applied at $\boldsymbol{\Theta}^{[k-1]}$ instead—a similar idea we used in [26] to deal with singular-value penalized vector GLMs. Define

$$\begin{aligned} &h(\boldsymbol{\Theta}^{[k]}, \boldsymbol{\Theta}^{[k-1]}) \\ &= l(\boldsymbol{\Theta}^{[k-1]}) + \langle \nabla_{\boldsymbol{\Theta}} l(\boldsymbol{\Theta}^{[k-1]}), \boldsymbol{\Theta} - \boldsymbol{\Theta}^{[k-1]} \rangle \\ &\quad + \frac{\rho_k}{2} \|\boldsymbol{\Theta} - \boldsymbol{\Theta}^{[k-1]}\|_F^2 + P(\boldsymbol{S}; \lambda) \\ &= l(\boldsymbol{\Theta}^{[k-1]}) + \langle -\boldsymbol{X} + \boldsymbol{H} \circ g^{-1}(\boldsymbol{\Theta}^{[k-1]}), \\ &\quad \boldsymbol{\Theta} - \boldsymbol{\Theta}^{[k-1]} \rangle + \frac{\rho_k}{2} \|\boldsymbol{\Theta} - \boldsymbol{\Theta}^{[k-1]}\|_F^2 + P(\boldsymbol{S}; \lambda) \end{aligned}$$

as our surrogate function. Then the kth iterate is given by

$$\begin{aligned} (8) \quad &(\boldsymbol{\alpha}^{[k]}, \boldsymbol{S}^{[k]}, \boldsymbol{V}^{[k]}) \\ &= \underset{\boldsymbol{\alpha}, \boldsymbol{S}, \boldsymbol{V} \in \mathbb{O}}{\arg\min} \, h(\boldsymbol{\alpha}, \boldsymbol{S}, \boldsymbol{V}; \boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}). \end{aligned}$$

It is clear that h satisfies $h(\boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}; \boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}) = f(\boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]})$. Suppose ρ_k is chosen such that

$$\begin{aligned} (9) \quad &h(\boldsymbol{\alpha}^{[k]}, \boldsymbol{S}^{[k]}, \boldsymbol{V}^{[k]}; \boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}) \\ &\geq f(\boldsymbol{\alpha}^{[k]}, \boldsymbol{S}^{[k]}, \boldsymbol{V}^{[k]}) \end{aligned}$$

This can be realized by setting a large enough ρ_k based on Taylor expansion, as detailed in Section 3.2. The objective function value is guaranteed nonincreasing throughout the iteration, as demonstrated by the sequence below:

$$\begin{aligned} f(\boldsymbol{\alpha}^{[k]}, \boldsymbol{S}^{[k]}, \boldsymbol{V}^{[k]}) \leq &h(\boldsymbol{\alpha}^{[k]}, \boldsymbol{S}^{[k]}, \boldsymbol{V}^{[k]}; \boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}) \\ \leq &h(\boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}; \\ &\boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}) \\ = &f(\boldsymbol{\alpha}^{[k-1]}, \boldsymbol{S}^{[k-1]}, \boldsymbol{V}^{[k-1]}). \end{aligned}$$

Clearly the second inequality does not depend on the optimality of $(\boldsymbol{\alpha}^{[k]}, \boldsymbol{S}^{[k]}, \boldsymbol{V}^{[k]})$ in equation (8). One can use iterative methods which approximates the solution to solve (8), as detailed below. Problem (8) can be rewritten as:

$$\begin{aligned} (\boldsymbol{\alpha}^{[k]}, \boldsymbol{S}^{[k]}, \boldsymbol{V}^{[k]}) = &\underset{\boldsymbol{\alpha}, \boldsymbol{S}, \boldsymbol{V} \in \mathbb{O}^{n \times r}}{\arg\min} \, \langle \nabla_{\boldsymbol{\Theta}} l(\boldsymbol{\Theta}^{[k-1]}), \mathbf{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V} \boldsymbol{S}^T \\ &- \boldsymbol{\Theta}^{[k-1]} \rangle + \frac{\rho_k}{2} \|\mathbf{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V} \boldsymbol{S}^T \\ &- \boldsymbol{\Theta}^{[k-1]}\|_F^2 + P(\boldsymbol{S}; \lambda) \end{aligned}$$

$$\begin{aligned} = &\underset{\boldsymbol{\alpha}, \boldsymbol{S}, \boldsymbol{V} \in \mathbb{O}^{n \times r}}{\arg\min} \, \frac{1}{2} \|\mathbf{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V} \boldsymbol{S}^T - \boldsymbol{\Xi}^{[k]}\|_F^2 \\ &+ \frac{1}{\rho_k} P(\boldsymbol{S}; \lambda), \end{aligned}$$

where

$$(10) \quad \boldsymbol{\Xi}^{[k]} = \boldsymbol{\Theta}^{[k-1]} + \frac{1}{\rho_k}(\boldsymbol{X} - \boldsymbol{H} \circ g^{-1}(\boldsymbol{\Theta}^{[k-1]})).$$

For notational simplicity, we write the problem as

$$\begin{aligned} (11) \quad &\underset{\boldsymbol{\alpha}, \boldsymbol{V}, \boldsymbol{S}}{\min} \, \tilde{f} = \frac{1}{2} \|\mathbf{1}_n \boldsymbol{\alpha}^T + \boldsymbol{V} \boldsymbol{S}^T - \boldsymbol{\Xi}^{[k]}\|_F^2 + P(\boldsymbol{S}; \lambda_{\rho_k}) \\ &\text{subject to } \boldsymbol{V}^T \boldsymbol{V} = \boldsymbol{I}_{r \times r}, \end{aligned}$$

where λ_{ρ_k} is defined such that $P(t; \lambda)/\rho_k = P(t; \lambda_{\rho_k}), \forall t \in \mathbb{R}$. The quadratic problem is much simpler than the initial criterion. As illustrated below, BCD can be easily applied, leading to an inner loop from line 8 to 13 in Algorithm 1.

$\boldsymbol{\alpha}$-*optimization* For the tth inner iteration step, $\boldsymbol{\alpha}$ has a closed-form solution: let $\nabla_{\boldsymbol{\alpha}} \tilde{f} = 0$, then $(\boldsymbol{\Xi}^{[k]} - \mathbf{1}_n \boldsymbol{\alpha}^T - \boldsymbol{V} \boldsymbol{S}^T)^T \mathbf{1}_n = 0$, that is $\boldsymbol{\alpha} \mathbf{1}_n^T \mathbf{1}_n = \boldsymbol{\Xi}^{[k]T} \mathbf{1}_n - \boldsymbol{S} \boldsymbol{V}^T \mathbf{1}_n$, therefore $\boldsymbol{\alpha}^{opt} = \frac{1}{n}(\boldsymbol{\Xi}^{[k]T} - \boldsymbol{S}^{[t-1]} \boldsymbol{V}^{[t-1]T}) \mathbf{1}_n$.

$\boldsymbol{S}$-*optimization* To solve the $\boldsymbol{S}$-optimization problem for a general penalty function $P(t; \lambda)$, we propose to use the thresholding rule based Θ-estimators, because multiple penalty functions often correspond to one threshold rule. Thresholding-based iterative selection procedure (TISP) [24] can be used to solve a P-penalized problem for any P associated with a thresholding rule (an odd, unbounded monotone shrinkage function) [29]. According to [25], Θ-estimator is linked to a general penalty function $P(t; \lambda)$ by

$$\begin{aligned} (12) \quad &P(t; \lambda) - P(0; \lambda) \\ &= \int_0^{|t|} (\sup\{s : \Theta(s; \lambda) \leq u\} - u) du + q(t; \lambda) \end{aligned}$$

for some nonnegative $q(\cdot; \lambda)$ such that $q(\Theta(s; \lambda)) = 0$ for all s. This conclusion is valid for any thresholding rule, so through Θ-estimators, we can handle all popular penalties including but not limited to the aforementioned ℓ_1, SCAD, and ℓ_p $(0 < p < 1)$, making this technique universally applicable.

Recall that we advocate the use of ℓ_0 *constraint*

$$\|\boldsymbol{S}\|_0/(pr) \leq q_e$$

in place of the ℓ_0 penalty $\lambda \|\boldsymbol{S}\|_0$ for tuning ease. The modified algorithm guarantees the nonincreasing of function value. TISP can be nicely adapted to solve the problem by employing a quantile thresholding rule $\Theta^{\#}$. The quantile thresholding rule is a special case of hard thresholding which correspond to the ℓ_0 penalty. Let $\Theta^{\#}(\boldsymbol{S}; q_e)$ be the

Algorithm 1 The SG-PCA Algorithm

Input: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$; r: the desired rank; M_{out}/M_{in}: the maximum outer/inner iteration number; $\varepsilon_{out}/\varepsilon_{in}$: inner and outer error tolerance; and the initial estimates $\boldsymbol{\alpha}^{[0]} \in \mathbb{R}^{p \times 1}$, $\boldsymbol{V}^{[0]} \in \mathbb{O}^{n \times r}$, $\boldsymbol{S}^{[0]} \in \mathbb{R}^{p \times r}$.

1: $k \leftarrow 0$;
2: $\boldsymbol{\Theta}^{[0]} = \boldsymbol{1}_n \boldsymbol{\alpha}^{[0]} + \boldsymbol{V}^{[0]} \boldsymbol{S}^{[0]^T}$;
3: **repeat**
4:     $k \leftarrow k + 1$;
5:     $\boldsymbol{\Xi}^{[k]} = \boldsymbol{\Theta}^{[k-1]} + \frac{1}{\rho_k}(\boldsymbol{X} - \boldsymbol{H} \circ g^{-1}(\boldsymbol{\Theta}^{[k-1]}))$;
6:     $t \leftarrow 0$;
7:     Initialize $\tilde{\boldsymbol{\alpha}}^{[0]} \leftarrow \boldsymbol{\alpha}^{[k-1]}$, $\tilde{\boldsymbol{S}}^{[0]} \leftarrow \boldsymbol{S}^{[k-1]}$, $\tilde{\boldsymbol{V}}^{[0]} \leftarrow \boldsymbol{V}^{[k-1]}$;
8:     **repeat**
9:         $t \leftarrow t + 1$;
10:         $\tilde{\boldsymbol{\alpha}}^{[t]} \leftarrow \frac{1}{n}(\boldsymbol{\Xi}^{[k]} - \tilde{\boldsymbol{V}}^{[t-1]} \tilde{\boldsymbol{S}}^{[t-1]^T})^T \boldsymbol{1}_n$;
11:         $\tilde{\boldsymbol{S}}^{[t]} \leftarrow \Theta^{\#}((\boldsymbol{\Xi}^{[k]^T} - \tilde{\boldsymbol{\alpha}}^{[t]} \boldsymbol{1}_n^T)\tilde{\boldsymbol{V}}^{[t-1]}; q_e)$;
12:         Compute SVD of $(\boldsymbol{\Xi}^{[k]} - \boldsymbol{1}_n \tilde{\boldsymbol{\alpha}}^{[t]^T})\tilde{\boldsymbol{S}}^{[t]} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{Q}^T$, set $\tilde{\boldsymbol{V}}^{[t]} \leftarrow \boldsymbol{P}\boldsymbol{Q}^T$;
13:     **until** $t \geq M_{in}$ or changes in $\tilde{\boldsymbol{\alpha}}$, $\tilde{\boldsymbol{S}}$, $\tilde{\boldsymbol{V}}$ no bigger than ε_{in}
14:     $\boldsymbol{\alpha}^{[k]} \leftarrow \tilde{\boldsymbol{\alpha}}^{[t]}$, $\boldsymbol{S}^{[k]} \leftarrow \tilde{\boldsymbol{S}}^{[t]}$, $\boldsymbol{V}^{[k]} \leftarrow \tilde{\boldsymbol{V}}^{[t]}$;
15:     $\boldsymbol{\Theta}^{[k]} \leftarrow \boldsymbol{1}_n \boldsymbol{\alpha}^{[k]^T} + \boldsymbol{V}^{[k]} \boldsymbol{S}^{[k]^T}$;
16: **until** $k \geq M_{out}$ or ($\|\boldsymbol{\Theta}^{[k]} - \boldsymbol{\Theta}^{[k-1]}\|_{\max} \leq \varepsilon_{out}$ & $|f^{[k]} - f^{[k-1]}| \leq \varepsilon_{out}$)
17: **return** $\boldsymbol{\alpha}^{[k]}$, $\boldsymbol{V}^{[k]}$, $\boldsymbol{S}^{[k]}$.

element-wise quantile threshold function, then

$$(13) \qquad \Theta^{\#}(\boldsymbol{S}; q_e) = \begin{cases} 0 & |s_{ij}| \leq \lambda_e \\ s_{ij} & |s_{ij}| > \lambda_e, \end{cases}$$

where λ_e is the $(1 - q_e)^{th}$ quantile of the $|s_{ij}|'s$. The tuning parameter q_e has a definite range ($0 < q_e < 1$), which serves as an upper bound for the nonzero percentage hence is much easier to interpret. Thus, the closed-form solution for $\boldsymbol{S}$ associated with (11) given $\boldsymbol{\alpha}$ and $\boldsymbol{V}$ is

$$\boldsymbol{S}^{opt} = \Theta^{\#}((\boldsymbol{\Xi}^{[k]^T} - \boldsymbol{\alpha}\boldsymbol{1}_n^T)\boldsymbol{V}; q_e).$$

See [25] for further details.

$\boldsymbol{V}$-*optimization* The optimization problem given $\boldsymbol{\alpha}$ and $\boldsymbol{S}$ with respect to $\boldsymbol{V}$ becomes

$$(14) \qquad \begin{aligned} &\min_{\boldsymbol{V}} \|\boldsymbol{\Xi}^{[k]^T} - \boldsymbol{\alpha}\boldsymbol{1}_n^T - \boldsymbol{S}\boldsymbol{V}^T\|_F^2 \\ &\text{subject to } \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{r \times r}. \end{aligned}$$

This can be identified as a Procrustes rotation problem, realizable through computing the Singular Value Decomposition of $(\boldsymbol{\Xi}^{[k]} - \boldsymbol{1}_n \boldsymbol{\alpha}^T)\boldsymbol{S} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{Q}^T$. The optimal $\boldsymbol{V}^{opt} = \boldsymbol{P}\boldsymbol{Q}^T$.

The full optimization process is given in details as Algorithm 1.

Although for simplicity of presentation, the main algorithm is illustrated with the element-wise form, group sparsity regularization can be used for screening and selection

purposes and is also interesting to us. In particular, for the group ℓ_0 constraint

$$\|\boldsymbol{S}\|_{2,0}/p \leq q_g,$$

it calls for the multivariate version of quantile thresholding $\overrightarrow{\Theta}^{\#}(\boldsymbol{S}^T; q_g)$:

$$(15) \qquad \overrightarrow{\Theta}^{\#}(\boldsymbol{S}^T; q_g) = \begin{cases} \boldsymbol{0} & \|\tilde{\boldsymbol{s}}_i\| \leq \lambda_g \\ \tilde{\boldsymbol{s}}_i & \|\tilde{\boldsymbol{s}}_i\| > \lambda_g, \end{cases}$$

where λ_g is the $(1 - q_g)^{th}$ quantile of the p row norm $\|\tilde{\boldsymbol{s}}_i\|'s$. The group-wise constraint version can be realized by simply changing $\Theta^{\#}$ to $\overrightarrow{\Theta}^{\#}$ in line 11 of Algorithm 1, which also satisfies the inequality (9) given ρ_k properly chosen.

3.2 Step size

Define $\tau_k = 1/\rho_k$ as the step size along the gradient in Equation (10). To guarantee function value nonincreasing, it suffices to derive a minimal ρ_k, or a maximal step size τ_k such that inequality (9) holds. Note that the regularization terms cancel out on both sides of the inequality, hence we focus on the negative log likelihood loss $l(\cdot)$ of $f(\cdot)$ throughout this section. Since the function value of $l(\cdot)$ depends on $\boldsymbol{\alpha}$, $\boldsymbol{V}$, and $\boldsymbol{S}$ *only* through $\boldsymbol{\Theta}$, we regard $\boldsymbol{\Theta}$ as the target variable accordingly.

Such a ρ_k can be obtained by various line search methods with (9) served as a stopping criterion. However, in some cases a *universal* step size may be derived based on Taylor expansion. Fortunately $l(\boldsymbol{\Theta})$ is usually smooth under the GLM setting, making Taylor expansion appropriate in order to derive a universal step size.

In the univariate case, for arbitrary y and x and f that is at least differentiable to the second degree, we have the approximation $f(y) \approx f(x) + f'(x)(y - x) + \frac{1}{2}f''(x)(y - x)^2$. To show the multivariate Taylor expansion for $l(\boldsymbol{\Theta}^{[k]})$, we consider a perturbation in the gradient function of the loss:

$$\begin{aligned} &\nabla l(\boldsymbol{\Theta}^{[k-1]} + \boldsymbol{\Delta}) \\ &= -\boldsymbol{X} + \boldsymbol{H} \circ b'(\boldsymbol{\Theta}^{[k-1]} + \boldsymbol{\Delta}) \\ &= -\boldsymbol{X} + \boldsymbol{H} \circ (b'(\boldsymbol{\Theta}^{[k-1]}) + (\boldsymbol{H} \circ b''(\boldsymbol{\xi}^{[k-1]})) \circ \boldsymbol{\Delta}) \\ &= \nabla l(\boldsymbol{\Theta}^{[k-1]}) + (\boldsymbol{H} \circ b''(\boldsymbol{\xi}^{[k-1]})) \circ \boldsymbol{\Delta}, \end{aligned}$$

with $\boldsymbol{\Delta} = \boldsymbol{\Theta}^{[k]} - \boldsymbol{\Theta}^{[k-1]}$ for some $\boldsymbol{\xi} = c\boldsymbol{\Theta}^{[k-1]} + (1-c)\boldsymbol{\Theta}^{[k]}$ where $c \in [0, 1]$. Hence

$$\begin{aligned} l(\boldsymbol{\Theta}^{[k]}) = l(\boldsymbol{\Theta}^{[k-1]}) &+ \langle \nabla_{\Theta} l(\boldsymbol{\Theta}^{[k-1]}), \boldsymbol{\Delta} \rangle + \\ &\frac{1}{2}\langle \boldsymbol{H} \circ b''(\boldsymbol{\xi}^{[k-1]}) \circ \boldsymbol{\Delta}, \boldsymbol{\Delta} \rangle. \end{aligned}$$

Denote $\boldsymbol{\Delta} = [\delta_{ij}]$, $\boldsymbol{H} = [h_{ij}]$, $b''(\boldsymbol{\xi}) = [b_{ij}]$, we have

$$\begin{aligned} &h(\boldsymbol{\Theta}^{[k]}, \boldsymbol{\Theta}^{[k-1]}) - f(\boldsymbol{\Theta}^{[k]}) \\ &= \frac{\rho_k}{2}\|\boldsymbol{\Delta}\|_F^2 - \frac{1}{2}\langle \boldsymbol{H} \circ b''(\boldsymbol{\xi}^{[k-1]}) \circ \boldsymbol{\Delta}, \boldsymbol{\Delta} \rangle \end{aligned}$$

$$\geq \frac{\rho_k}{2}\|\boldsymbol{\Delta}\|_F^2 - \frac{1}{2}\sum_{i,j}\delta_{ij}^2 h_{ij}b_{ij}$$
$$\geq \frac{\rho_k}{2}\|\boldsymbol{\Delta}\|_F^2 - \frac{1}{2}\sum_{i,j}\delta_{ij}^2 b_{ij}$$
$$\geq \frac{\rho_k}{2}\|\boldsymbol{\Delta}\|_F^2 - \frac{1}{2}\|b''(\boldsymbol{\xi})\|_{\max}\|\boldsymbol{\Delta}\|_F^2,$$

where $\|\cdot\|_{\max}$ is defined as the maximal absolute value in the matrix, and $\boldsymbol{H}$ is a matrix of at most $1's$.

If $\rho_k \geq \|b''(\boldsymbol{\xi})\|_{\max}$, that is, step size

$$\tau_k = \frac{1}{\rho_k} \leq 1/\|b''(\boldsymbol{\xi})\|_{\max},$$

function value will decrease.

$\|b''(\boldsymbol{\xi})\|_{\max}$ depends on the specific distribution. Table 1 provides the forms of $b''(\cdot)$ according to a list of distributions. Notice that it always equals 1 under Gaussian distribution and it is bounded by $\frac{1}{4}$ for Bernoulli, Binomial and Multinomial data. By taking these suprema, nonincreasing function value can be guaranteed via the universal step size choices $\tau = 1$ or $\tau = 4$ respectively for all k. However, in the Poisson, Gamma, and Exponential cases, $\|b''(\boldsymbol{\xi})\|_{\max}$ has no finite supremum, thus no universal step size may be achieved. Since inequality (9) only requires decrease on the $(k-1)^{th}$ step, we may select τ_k based on an ad-hoc reasonably large value of $\|b''(\boldsymbol{\xi})\|_{\max}$ or from line search methods to satisfy the local descent condition. Since an arbitrary small constant τ_k easily leads to slow convergence and inaccurate results, line search is vastly preferred for efficient and accurate convergence. The line search scheme will be outlined in details as a part of Algorithm 2.

3.3 Fast computation

3.3.1 Accelerated gradient with line search

The lack of theoretical maximal step size under some distributions is expected to result in slow convergence, which motivates us to find an accelerated algorithm. Nesterov's second accelerated first-order method [19] is a popular tool to improve the convergence speed for unconstrained smooth problem. It can achieve the convergence rate $O(1/k^2)$ where k is the iteration number. This rate is shown to be the optimal convergence rate for smooth and convex first-order problems, and later extended to a large class of non-smooth convex ones including Lasso [1]. We borrow the framework of accelerated proximal gradient [33, 20] and define an operator $\mathcal{P}(\cdot): \mathbb{R}^n \to \mathbb{R}^n$ which solves a *non-convex* optimization problem. Theoretical work has been done analyzing the approximation accuracy when such regularization is convex [34], to which our problem does not belong. However, experience shows that under non-convex settings the second accelerated method still works well if the step size τ_k is further reduced and appropriately selected.

For notation simplicity, we take $\boldsymbol{\Theta}$ to represent $\boldsymbol{\alpha}$, $\boldsymbol{V}$ and

---

Algorithm 2 The SG-PCA$^\ell$ Algorithm with acceleration and line search

Input: $\boldsymbol{X} \in \mathbb{R}^{n \times p}$; r: the desired rank; M_{out}/M_{in}: the maximum outer/inner iteration number; $\varepsilon_{out}/\varepsilon_{in}$: inner and outer error tolerance; $\eta(0 < \eta < 1)$; and the initial estimates $\boldsymbol{\alpha}^{[0]} \in \mathbb{R}^{p \times 1}$, $\boldsymbol{V}^{[0]} \in \mathbb{O}^{n \times r}$, $\boldsymbol{S}^{[0]} \in \mathbb{R}^{p \times r}$.

1: $k \leftarrow 0$;
2: $\boldsymbol{\Theta}^{[0]} \leftarrow \mathbf{1}_n \boldsymbol{\alpha}^{[0]} + \boldsymbol{V}^{[0]}\boldsymbol{S}^{[0]^T}$;
3: $\boldsymbol{\nu}^{[0]} \leftarrow \boldsymbol{\Theta}^{[0]}$;
4: **repeat**
5:     $k \leftarrow k+1$;
6:     $\theta_k \leftarrow 1$ when $k = 1,2$, and $\theta_k = \frac{2}{k+2}$ otherwise;
7:     $n_{ls} \leftarrow 0$;
8:     $\boldsymbol{\nu}_{ls}^{cur} \leftarrow \boldsymbol{\nu}^{[k-1]}$, $\boldsymbol{\Theta}_{ls}^{cur} \leftarrow \boldsymbol{\Theta}^{[k-1]}$, $\tau = 1/\|\boldsymbol{X}\|_{\max}$;
9:     **repeat**
10:         $n_{ls} \leftarrow n_{ls} + 1$;
11:         $\tau \leftarrow \eta\tau$;
12:         $\boldsymbol{Y} \leftarrow (1-\theta_k)\boldsymbol{\Theta}_{ls}^{cur} + \theta_k\boldsymbol{\nu}_{ls}^{cur}$;
13:         $\boldsymbol{\nu}_{ls}^{new} \leftarrow \boldsymbol{\nu}_{ls}^{cur} - \frac{\tau}{\theta_k}\nabla l_\Theta(\boldsymbol{Y})$ where $\nabla l_\Theta(\boldsymbol{Y}) = -\boldsymbol{X} + \boldsymbol{H} \circ g^{-1}(\boldsymbol{Y})$;
14:         $t \leftarrow 0$;
15:         Initialize $\tilde{\boldsymbol{\alpha}}^{[0]} \leftarrow \boldsymbol{\alpha}^{[k-1]}$, $\tilde{\boldsymbol{S}}^{[0]} \leftarrow \boldsymbol{S}^{[k-1]}$, $\tilde{\boldsymbol{V}}^{[0]} \leftarrow \boldsymbol{V}^{[k-1]}$;
16:         **repeat**
17:             $t \leftarrow t+1$;
18:             $\tilde{\boldsymbol{\alpha}}^{[t]} \leftarrow \frac{1}{n}(\boldsymbol{\nu}_{ls}^{cur} - \tilde{\boldsymbol{V}}^{[t-1]}\tilde{\boldsymbol{S}}^{[t-1]T})^T\mathbf{1}_n$;
19:             $\tilde{\boldsymbol{S}}^{[t]} \leftarrow \Theta^{\#}((\boldsymbol{\nu}_{ls}^{curT} - \tilde{\boldsymbol{\alpha}}^{[t]}\mathbf{1}_n^T)\tilde{\boldsymbol{V}}^{[t-1]}; q_e)$;
20:             Compute SVD of $(\boldsymbol{\nu}_{ls}^{cur} - \mathbf{1}_n\tilde{\boldsymbol{\alpha}}^{[t]T})\tilde{\boldsymbol{S}}^{[t]} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{Q}^T$, set $\tilde{\boldsymbol{V}}^{[t]} \leftarrow \boldsymbol{P}\boldsymbol{Q}^T$;
21:         **until** $t > M_{in}$ or changes in $\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{S}}, \tilde{\boldsymbol{V}}$ no bigger than ε_{in}
22:         $\boldsymbol{\nu}_{ls}^{new} \leftarrow \mathbf{1}_n\tilde{\boldsymbol{\alpha}}^{[t]T} + \tilde{\boldsymbol{V}}^{[t]}\tilde{\boldsymbol{S}}^{[t]T}$;
23:         $\boldsymbol{\Theta}_{ls}^{new} \leftarrow (1-\theta_k)\boldsymbol{\Theta}_{ls}^{cur} + \theta_k\boldsymbol{\nu}_{ls}^{cur}$;
24:     **until** $f(\boldsymbol{\Theta}_{ls}^{new}) \leq f(\boldsymbol{Y}) + \langle\nabla f(\boldsymbol{Y}), \boldsymbol{\Theta}_{ls}^{new} - \boldsymbol{Y}\rangle + \frac{\theta_k}{2\tau}\|\boldsymbol{\Theta}_{ls}^{new} - \boldsymbol{Y}\|_F^2$ or $n_{ls} > 10$
25:     $\boldsymbol{\alpha}^{[k]} \leftarrow \tilde{\boldsymbol{\alpha}}^{[t]}$, $\boldsymbol{S}^{[k]} \leftarrow \tilde{\boldsymbol{S}}^{[t]}$, $\boldsymbol{V}^{[k]} \leftarrow \tilde{\boldsymbol{V}}^{[t]}$;
26:     $\boldsymbol{\nu}^{[k]} \leftarrow \boldsymbol{\nu}_{ls}^{new}$, $\boldsymbol{\Theta}^{[k]} \leftarrow \boldsymbol{\Theta}_{ls}^{new}$;
27: **until** $k > M_{out}$ or $(\|\boldsymbol{\Theta}^{[k]} - \boldsymbol{\Theta}^{[k-1]}\|_{\max} \leq \varepsilon_{out}$ & $|f^{[k]} - f^{[k-1]}| \leq \varepsilon_{out})$
28: **return** $\boldsymbol{\alpha}^{[k]}, \boldsymbol{V}^{[k]}, \boldsymbol{S}^{[k]}$.

$\boldsymbol{S}$ jointly. The second method introduces two momentum terms $\boldsymbol{Y}$ and $\boldsymbol{\nu}^{[k]}$ in the kth iterate such that

$$(16) \qquad \boldsymbol{Y} = (1-\theta_k)\boldsymbol{\Theta}^{[k-1]} + \theta_k\boldsymbol{\nu}^{[k-1]},$$

$$(17) \qquad \boldsymbol{\nu}^{[k]} = \mathcal{P}(\boldsymbol{\Theta} - \frac{\tau_k}{\theta_k}\nabla l(\boldsymbol{Y})),$$

$$(18) \qquad \boldsymbol{\Theta}^{[k]} = (1-\theta_k)\boldsymbol{\Theta}^{[k-1]} + \theta_k\boldsymbol{\nu}^{[k]}.$$

$\mathcal{P}$ yields the answers to the non-convex optimization problem

$$\underset{\boldsymbol{\Theta}=\mathbf{1}_n\boldsymbol{\alpha}^T+\boldsymbol{V}\boldsymbol{S}^T}{\operatorname{argmin}} l(\boldsymbol{Y}) + \langle\nabla l(\boldsymbol{Y}), \boldsymbol{\Theta} - \boldsymbol{Y}\rangle + \rho_k\theta_k\|\boldsymbol{\Theta} - \boldsymbol{\nu}^{[k]}\|_F^2$$
$$+ P(\boldsymbol{S}; \lambda)$$
$$\text{subject to } \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_{r \times r},$$

essentially problem (11) with $\boldsymbol{\Xi} = \boldsymbol{\Theta} - \rho_k \theta_k \nabla l(\boldsymbol{Y})$. θ_k should be chosen such that

$$\frac{1 - \theta_k}{\theta_k^2} \leq \frac{1}{\theta_k^2},$$

and $\theta_k = \frac{2}{k+2}$ functions as a simple choice that satisfies this inequality.

When the theoretical step size τ is not available, it can be relaxed by an initial guess and further decreased in every iterate until the backtracking criterion is met—in our case, inequality (9): the idea is to take a conservative step size $\tau^{(0)}$ first in iteration k, and repeatedly proceed along the gradient with a subsequently even smaller step size $\tau^{(m)} = \eta \tau^{(m-1)}$ ($0 < \eta < 1$) until sufficient decrease in function value has been achieved, before entering iteration $k + 1$.

Algorithm 2 lines out SG-PCA$^\natural$, geared especially towards the Poisson case. Note that line 13 to line 22 in Algorithm 2 compose the non-convex $\mathcal{P}$ which includes a low-rank and a sparsity regularization.

3.3.2 Progressive screening

In cases where p is extremely high, it is sometimes just infeasible to iterate till convergence, neither is it suitable to remove a big fraction of dimensions altogether in carrying out the algorithm *once*. For instance, p is 1,000 and the desired post-screening dimension is 100. It is considered greedy to deem 90% of the original dimensions nuisance from the very first iteration.

To enhance the scalability as well as to reduce such greediness as problem size blows up, it is natural to *adaptively* kill the dimensions depending on the iteration progress. The elimination of one dimension is equivalent to enforcing the row norm of $\boldsymbol{S}$ to zero. Therefore, by optimizing the problem subject to a row-sparse criterion repeatedly and complying a once-zero-stays-zero strategy, we are able to progressively screen and squeeze the dimensions.

The screening problem is considered under the groupwise sparse setting. Let the desired percentage of nonzero dimensions be q_g, then instead of enforcing q_g as the tuning parameter directly in step 19 of Algorithm 2, we introduce a sequence $Q(t, k)$ which decreases from p to $q_g p$ and discard the zero dimensions, where t and k stands for the inner and outer loop iteration number respectively. $Q(t, k)/d$ serves as the new sparsity parameter with d denoting the cardinality of the nonzero index set $\mathcal{N}$ in the current iteration. In this way, the problem is tackled in a smaller space as the iteration proceeds. Since reducing the same amount of dimension is often easier at the beginning when the candidate d is much higher than towards the end, a sigmoidal decay best suits the need—$Q(t, k) = 2p/(1 + \exp(aT))$ is recommended in particular, where $a \in [0.01, 0.1]$ determines the speed of the decay. $T = k, t, kt$ controls whether the decaying process is dependent on the outer, inner loop progression or both, with the choice of inner loop being the fastest one and outer loop on the conservative side.

Algorithm 3 Progressive Screening

> **Input:** $\tilde{\boldsymbol{S}}^{[t]} \in \mathbb{R}^{p \times r}$, $\tilde{\boldsymbol{\alpha}}^{[t]}$, $\tilde{\boldsymbol{V}}^{[t-1]}$, $\boldsymbol{X}$, $Q(t, k)$, $\mathcal{N}$;

1: $d \leftarrow \text{card}(\mathcal{N})$
2: $\tilde{\boldsymbol{S}}^{[t]} \leftarrow \overrightarrow{\Theta}^{\#}((\boldsymbol{\nu}^{[k]T} - \tilde{\boldsymbol{\alpha}}^{[t]} \mathbf{1}_n^T) \tilde{\boldsymbol{V}}^{[t-1]}; Q(t, k)/d)$;
3: $\mathcal{J} \leftarrow \{\mathcal{J} : \|\tilde{\boldsymbol{S}}^{[t]}(j, 1 : r)\| \neq 0\}$;
4: $\mathcal{N} \leftarrow \mathcal{N}(J)$;
5: $\tilde{\boldsymbol{S}}^{[t]} \leftarrow \tilde{\boldsymbol{S}}^{[t]}(\mathcal{J}, 1 : r)$, $\boldsymbol{X} \leftarrow \boldsymbol{X}(1 : n, \mathcal{J})$, $\tilde{\boldsymbol{\alpha}}^{[t]} \leftarrow \tilde{\boldsymbol{\alpha}}^{[t]}(\mathcal{J})$;
6: **return** $\tilde{\boldsymbol{S}}^{[t]}$, $\tilde{\boldsymbol{\alpha}}^{[t]}$, $\tilde{\boldsymbol{V}}^{[t-1]}$, $\boldsymbol{X}, \mathcal{J}, \mathcal{N}$.

Algorithm 3 lines out the progressive screening scheme to replace step 19 of Algorithm 2. $\mathcal{N}$ stands for the index set of nonzero dimensions with respect to the original index and it is initialized as $\mathcal{N} = (1, 2, \cdots, p)$.

The progressive screening design integrated in SG-PCA greatly enhances the scalability of our algorithm when problem complexity explodes. Section 4.2 includes results with dimension up to $p = 13{,}220$ to demonstrate such scalability when the competing methods take several times longer or even fail to converge within reasonable time.

4. NUMERICAL EXPERIMENT

4.1 Simulation data

We generate Gaussian, Bernoulli and Poisson simulation data under three different settings. Simulation data set is built similarly to the spiked covariance model, which is popular among sparse PCA related research [11, 3]. Simulate an $n \times p$ matrix $\boldsymbol{X}$ such that

$$(19) \qquad g(E(\boldsymbol{X})) = \boldsymbol{P} \boldsymbol{D} \boldsymbol{Q}^T,$$

where $\boldsymbol{P} \in \mathbb{R}^{n \times r}$ has independent and identically distributed standard normal entries, $\boldsymbol{D}$ is a diagonal matrix with $(\lambda_1, \cdots \lambda_r)$ on its diagonal, $\boldsymbol{Q}$ is a deterministic $p \times r$ orthonormal matrix, and q^* is the true nonzero rate in $\boldsymbol{Q}$, with q_e^* denoting element-wise nonzero percentage and q_g^* the percentage of nonzero rows.

The three scenarios are generated with $n = 100$ observations and $p = 200$ dimensions. We further specify the true r^*, q_e^* and q_g^* below: (a) $r^* = 1$ with sparse vector $\boldsymbol{q}$ such that $q_e^* = \|\boldsymbol{q}\|_0/p = 1\%$, with the exception of Bernoulli data at $q_e^* = 5\%$; (b) $r^* = 4$ with element-wise sparse matrix $\boldsymbol{Q}$ such that $q_e^* = \|\boldsymbol{Q}\|_0/(pr) = 8\%$; and (c) $r^* = 4$ with row sparse matrix $\boldsymbol{Q}$ such that $q_g^* = \|\boldsymbol{Q}\|_{2,0}/p = 20\%$.

We implement logisticPCA [15] to compare Bernoulli experiments, and sPCA-rSVD [30] is applied for all distribution types.

To gauge the precision of algorithms, we compare the error on $\boldsymbol{\Theta}$, function value, subspace and selection. Different from a supervised problem, the complexity goes up *whenever n or p increases*. We thus scale the $\boldsymbol{\Theta}$-error to $1000\|\hat{\boldsymbol{\Theta}} - \boldsymbol{\Theta}^\star\|_F^2/np$. Deviance is defined as $2(l(\boldsymbol{X}; \hat{\boldsymbol{\Theta}}) - l(\boldsymbol{X}; \boldsymbol{\Theta}_S))$ where $\hat{\boldsymbol{\Theta}}$ denotes the estimated parameter and

Table 2. $n = 100, p = 200$. *Trimmed mean (10%) results of 100 repetitions. Gaussian experiments utilizes 2 out of 10 initial points; Bernoulli 3/20; Poisson examines 5/30. Gaussian methods under the Poisson model produces Θ as estimators of $E(X)$, and it results in numerically unstable deviance thus omitted in table*

Data	Method	Error			Selection		Time
		Θ-Error	Dev	Angle	MR(%)	FP(%)	(s)
Setting (a): $r^* = 1$, $q_e^* = 0.01$, $q_e = q_e^*$							
Gaussian	SG-PCA	0.39	1.00	0.14	0.00	0.00	0.23
	sPCA-rSVD	0.48	1.17	0.14	0.00	0.00	0.25
Bernoulli $q_e^* = 0.05$	SG-PCA	3.48	1.00	21.17	15.48	0.81	0.58
	logisticPCA	4.00	1.01	22.19	22.00	1.15	0.48
	sPCA-rSVD	4.63	1.02	24.59	20.00	1.05	0.51
Poisson	SG-PCA$^{\sharp}$	8.83	1.00	22.93	49.00	0.49	1.45
	sPCA-rSVD	49.04	–	16.02	15.00	0.15	1.50
	SG-PCA^{Gau}	8.74	–	16.02	15.00	0.15	0.36
Setting (b): $r^* = 4$, $q_e^* = 0.08$, $q_e = 4q_e^*$, $q_g = 4q_e^*$							
Gaussian	SG-PCA	0.03	1.00	0.13	3.53	26.26	0.02
	sPCA-rSVD	0.93	52.28	0.13	1.56	26.13	0.27
	SG-PCA$_g$	0.03	1.00	0.12	0.10	26.10	0.02
Bernoulli	SG-PCA	6.43	1.00	33.22	51.33	30.55	0.79
	logisticPCA	7.64	1.03	46.38	48.70	30.32	1.41
	sPCA-rSVD	9.14	1.10	44.66	42.19	29.76	1.67
	SG-PCA$_g$	5.94	1.01	29.53	12.29	27.16	0.79
Poisson	SG-PCA$^{\sharp}$	12.31	1.00	88.74	66.17	31.84	1.32
	sPCA-rSVD	46.35	–	77.83	27.64	27.64	6.07
	SG-PCA$_g^{\sharp}$	12.25	0.55	88.54	66.26	31.85	1.29
	SG-PCA$_g^{Gau}$	13.09	–	76.98	21.31	27.94	0.56
Setting (c): $r^* = 4$, $q_g^* = 0.20$, $q_e = 2q_g^*$, $q_g = 2q_g^*$							
Gaussian	SG-PCA	0.03	1.00	0.18	0.19	25.03	0.19
	sPCA-rSVD	0.93	40.91	0.18	0.25	25.03	2.70
	SG-PCA$_g$	0.03	1.00	0.16	0.00	25.00	0.21
Bernoulli	SG-PCA	6.05	1.00	31.93	23.91	30.98	0.85
	logisticPCA	7.65	1.03	49.33	26.64	31.66	1.52
	sPCA-rSVD	9.12	1.11	45.18	25.00	31.25	2.51
	SG-PCA$_g$	5.58	1.01	28.03	0.15	25.04	0.79
Poisson	SG-PCA$^{\sharp}$	12.31	1.00	88.40	66.17	31.84	1.29
	sPCA-rSVD	44.29	–	84.31	3.20	25.80	6.49
	SG-PCA$_g^{\sharp}$	12.26	0.16	87.97	57.06	39.27	1.27
	SG-PCA$_g^{Gau}$	14.56	–	84.31	0.50	25.13	0.60
	SG-PCA$_g^{Gau\ 1}$	11.19	–	49.20	0.00	35.71	0.69

Θ_S that of the saturated model. For simplicity of presentation and comparison, we scale the deviance of SG-PCA for each distribution under each setting to 1, and deviance of the other methods as a ratio to the first one. In addition, the largest canonical angle between the estimated loading space and the true space is also interesting. Selection accuracy is evaluated by missing rate (MR) which stands for missed nonzero loading elements, and false positive rate (FP) which represents the false alarm rates for actual zero elements.

100 simulation experiments are run for each distribution under each setting. The 10% trimmed means of evaluation metrics are given in Table 2, since trimmed means are more robust than means for non-Gaussian metrics and more comprehensive than median for using all the values. Notice that sparsity parameter q_e and q_g differ among settings. $r = 1$

[1]$r = 3r^*$

is a much simpler problem compared to the others, thus $q_e = q_e^*$ would suffice. Setting (b) has $q_e = 4q_e^*$ and $q_g = 4q_e^*$ compared to (c) where the sparsity parameter is only twice that of the true nonzero level, because a more conservative sparsity parameter is necessary for keeping the missing rate under control when the loading matrix is simulated element-wise sparse. The highly non-convex optimization problem produces local minima, hindering the recovery of the global optimal solution. Therefore we use a multiple-start scheme [22]. It first utilizes m_1 random initial points, out of which we choose m_2 ($m_2 < m_1$) points with the lowest function values after n_1 ($n_1 = 2$) iterations and proceed until convergence; finally, the initial point producing the lowest function value is selected. Different m_1 and m_2 are used for the three distributions, since the degree to which non-convexity affects convergence differs across the distributions according to our experiments. Notice that for the rank-four scenarios (b) and (c), both the element-wise and group-wise SG-PCA are implemented. The element-wise version is employed as a fair reference comparing with the competitive methods due to their sequential property, where any group constraint would act just like an element one. However, the group ℓ_0 constraint should ideally be utilized for selection and fit under setting (c), and for simplicity and speed under setting (b).

In the Gaussian experiments, SG-PCA almost always demonstrates better results to sPCA-rSVD across all evaluation metrics, establishing a trustworthy ground line for our algorithm. SG-PCA also takes a fraction of the computation time of sPCA-rSVD especially when $r^* = 4$, probably due to its *joint* estimating property, as opposed to the rank-by-rank behavior of sPCA-rSVD. In fact, SG-PCA always demonstrates higher efficiency across all experiments. Under settings (b) and (c), the group-wise algorithm SG-PCA$_g$ further enhances the overall performance compared to the element-wise form, especially in terms of selection accuracy.

This is also true among the Bernoulli experiments. While our method consistently outperforms the comparison ones across all three settings in every aspect—supporting the superior formulation of the negative log-likelihood serving as the loss function, SG-PCA$_g$ is always able to improve the performance even further. Group sparsity constraint is an effective tool unique to our algorithm which helps with fast dimension elimination, makes selection possible and much more accurate. Allowing q_g to take a slightly higher value enables it to function as an upper bound, reducing the missing rate considerably at a small cost of the false positive rate, which in turn improves the space recovery accuracy.

The Poisson simulations are more challenging because of its complexity in step size selection and its numerical tendency to diverge. Algorithm 2 (SG-PCA$^\ell$) is applied to all Poisson settings for its comparatively superior performance. Yet the $r = 4$ scenarios still see difficulty in Poisson estimation due to the space ambiguity issues—the set of loading vectors that define a space is not unique. It is suspected that sPCA-rSVD outperforms SG-PCA under Poisson because of the simple Gaussian algorithm where numerical issues are less likely to occur. Thus, we list our Gaussian algorithm SG-PCA^{Gau} for comparison as well. Better results are achieved through the SG-PCA^{Gau} and SG-PCA$_g^{Gau}$ algorithms. In an attempt to cope with the space ambiguity, SG-PCA$_g^{Gau}$ with $r = 3r^*$ is also included under setting (c), because a higher rank parameter introduces more flexibility in defining the subspace. This results in substantially superior space recovery accuracy.

To sum up, our method demonstrates better efficacy and efficiency of the joint space much more accurately under the Gaussian and Bernoulli settings. It is able to directly eliminate nuisance dimensions for either screening or selection purposes due to the group sparsity constraint, regardless of the underlying true sparse pattern of the loading matrix. Although the Poisson experiments do not show results as excellent as the other two, we are able to achieve encouraging recovery accuracy with the Gaussian alternative of SG-PCA. Relaxation of rank r to a higher number may also help relieve space ambiguity issues. It is important to note that the results and conclusions are restricted to the formulation of the synthetic examples above.

4.2 SNPs data

Single nucleotide polymorphism (SNP) data denotes the variation at the level of a single base pair in DNA sequence that occurs at over 1% within a species. SNPs data attracts a great amount of attention due to the belief of its undiscovered association with various stratification ways, especially diseases. Two tasks are of particular interest: one is to reveal the underlying structure for a certain population, the other is to select particular features meaningful in clustering the observations. Our proposed methodology, while its primary goal is to recover the jointly sparse and low-rank structure, also takes an interest in learning the informative features.

The specific data set we use as an example is the SNPs data made available from the international HapMap project [6]. There are 1,322 shared base pair information from 270 observations, consisting of 90 Africans, 90 Caucasians and 90 Asians. The missing rate is 0.53% and it is handled by the masking scheme. Since the data consist of binary entries denoting the existence of a base pair mutation, traditional dimension reduction with Gaussian assumption is inappropriate. SG-PCA is applied with parameters selected as $r = 3$, $q_g = 0.10$ and $q_e = 0.60$, while the group-wise sparsity is enforced progressively and the element-wise one implemented afterwards. Rank and sparsity parameters in our formulation are not as sensitive as some other algorithms such as ℓ_1. We fix one, alternate the other on a crude scale until the best clustering effect is achieved. Figure 1 shows the data projected to the first three principal directions learned by SG-PCA. Although the nature of our approach is unsupervised, the European, Asian and African subjects are well separated in the subspace. It is noteworthy that only 12.56%
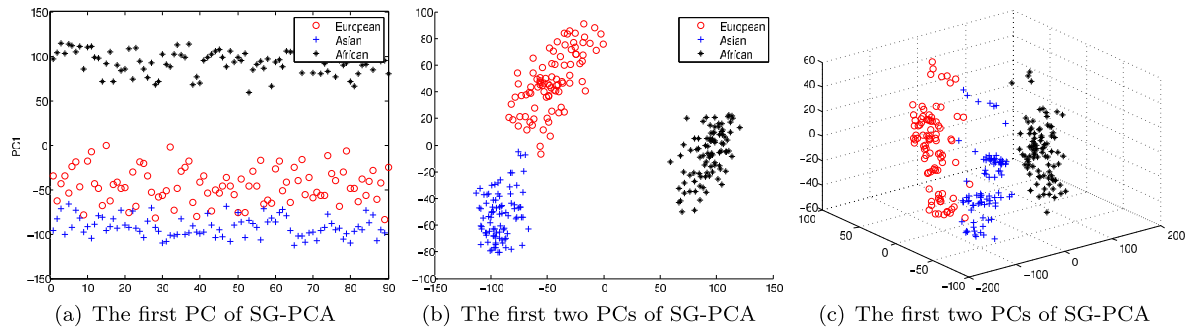
(a) The first PC of SG-PCA   (b) The first two PCs of SG-PCA   (c) The first two PCs of SG-PCA

Figure 1. SG-PCA on HapMap SNPs data. $n = 270$, $p = 1{,}322$, $r = 3$, $q_g = 0.10$ *with progressive screening, and* $q_e = 0.60$.

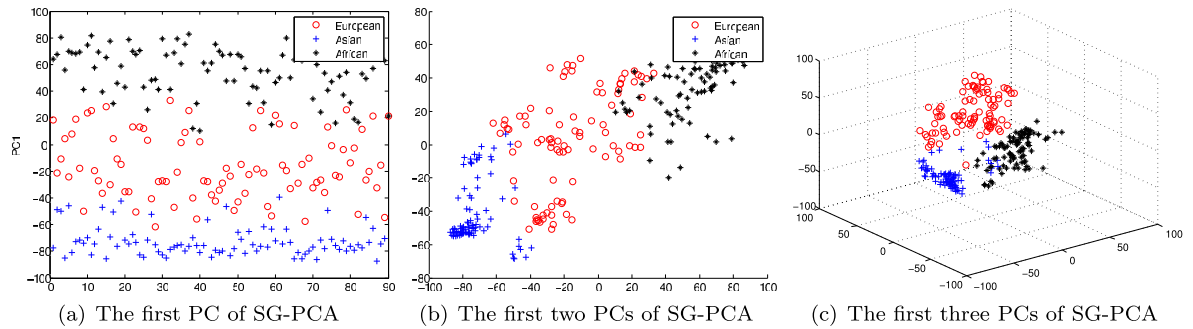(a) The first PC of SG-PCA   (b) The first two PCs of SG-PCA   (c) The first three PCs of SG-PCA

Figure 2. SG-PCA on inflated HapMap SNPs data. $n = 270$, $p = 13{,}220$, $r = 3$, $q_g = 0.01$ *with progressive screening, and* $q_e = 0.60$.

of the original dimensions are used in all three PCs, out of which only 8 dimensions are shared by the three directions in common.

To further demonstrate the selection capability and algorithm scalability of SG-PCA, we inflate the original 270 by 1,322 data matrix with 9 times as many nuisance dimensions. With parameters chosen at $r = 3$, $q_g = 0.01$ and $q_e = 0.60$, our algorithm is able to produce the clustering in Figure 2. Only 76 are selected from the starting 13,220 dimensions and none of the inflated junk dimensions is falsely chosen. It is also worth noting that the entire algorithm with 3 second-stage initial points out of 20 first-stage ones takes 198 seconds, a fraction of the 712 seconds with logisticPCA. sPCA-rSVD appears infeasible under such high-dimensional setting, for each rank takes more than 10 minutes to converge.

4.3 The CNAE-9 text data

The CNAE-9 text data is a set of 9 categories from the National Classification of Economic Activities. It has been preprocessed so that each document is presented as a row, and each column contains the frequencies of a specific word in the documents.

Three out of nine categories are extracted for use in the experiments, where two thirds of the observations are for training purposes, leaving the remaining one third as testing

data set. The training data consists of 240 observations only, compared to a much larger p of 857.

The data set is highly zero-inflated (99.27%), which is a native characteristic of text data, as not all words are present in all categories of documents. It is not to be confused with the missing rate 2.8%, handled by the masking method. The low nonzero entries in data probably further supports the recovery of sparse loadings. A series of progressive group-wise SG-PCA$^{\ell}$ models are fitted under the Poisson distribution, projecting the data to some lower-dimensional subspaces, after which a K-nearest-neighbor (KNN) classification procedure is carried out on the transformed space to calculate the test misclassification error.

Rank r and sparsity parameter q_g are tuned based on the testing KNN misclassification error. The tuning approach is similar to the HapMap application by a series of experiments, where the parameter values are alternatively changed. Since q_g serves as an upper bound of the true nonzero loading vectors, the result is not very sensitive to q_g as long as a conservative choice is given. Hence an exhaustive grid search is not necessary.

Classification on the reduced dimensions produces considerably better results compared to the original space. Naïve KNN on the unprocessed data produces a misclassification rate of 66.67%—the equivalence of random guess. Regular PCA projects the data to a new coordinate system, greatly enhancing the classification accuracy from an error
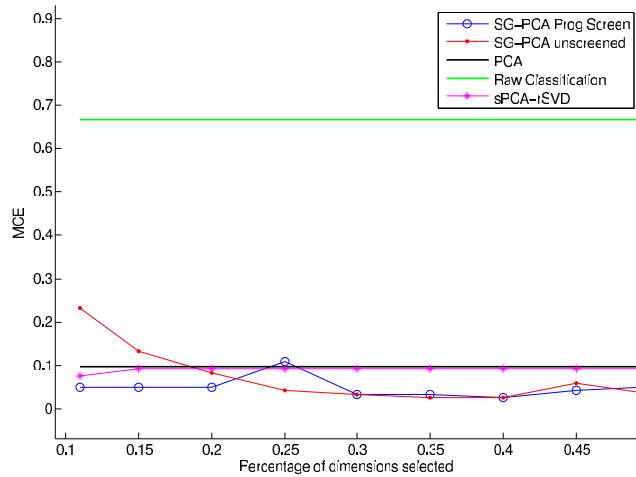
Figure 3. Misclassification rate comparison of SG-PCA$^\sharp$ at
r = 8 on CNAE-3 data.

of 66.67% to 9.58% when $r = 8$, and 8.33% when $r = 20$. However, since no sparse loading vector is pursued, each PC still utilizes all the original words.

The new loss function coupled with the sparsity constraint delivers a parsimonious representation in the resulting PCs while keeping the strong classification power in the following KNN procedure. Figure 3 shows the misclassification rate at $r = 8$ corresponding to a range of q_g values chosen at $q_g = 0.11, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50$. In sum, SG-PCA$^\sharp$ with progressive screening seem to have the best performance. It surpasses SG-PCA$^\sharp$ without progressive screening especially at the lower q_g values because greediness is very much alleviated when the sparsity regularization is stringent. Both achieve a much lower misclassification error than all the other methods, with a selection rate as low as 11% of the original dimension. sPCA-rSVD performs similarly to regular PCA. SG-PCA$^\sharp$ with progressive screening is also the most computationally efficient among all. One multiple-initial-point process consumes on average 12.07 seconds, whereas regular SG-PCA$^\sharp$ takes 30.53 seconds and sPCA-rSVD 34.79 seconds to converge.

5. SUMMARY

SG-PCA algorithm designs a scalable distribution-specific methodology for the purpose of unsupervised low-rank and sparse data representation, such that PCA is both rectified in theory and accurate to exponential family distributions beyond Gaussianity. Missing values are taken into consideration with almost no cost to computation efficiency. Nesterov's second accelerated gradient method as well as line search are incorporated for faster optimization especially under circumstances where theoretical maximal step size is not calculable. A progressive screening strategy is

employed to alleviate the greedy nature of sharp dimension reduction, while enhancing the scalability. Although the canonical link function is favored throughout the paper for its convenience, non-canonical links can also be handled with the same surrogate function technique, as long as the inverse link functions are differentiable.

However, the convenience of masking the missing values is based upon the assumption that the observations are independent given the missing indices. In fact, the whole model depends on the *conditional* independence assumption among the observations—apart from the association made possible by the low-rank projection of the multivariate data matrix. Such an assumption coincides with that of the regular PCA, but it is obviously an over-simplification of reality. An immediate extension is to corporate association structure into formulating the model, capturing correlation among the non-Gaussian covariates.

REFERENCES

[1] BECK, A. and TEBOULLE, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences* **2** 183–202. MR2486527

[2] BLEI, D. M., NG, A. Y. and JORDAN, M. I. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research* **3** 993–1022.

[3] CAI, T. T., MA, Z., WU, Y. et al. (2013). Sparse PCA: Optimal rates and adaptive estimation. *The Annals of Statistics* **41** 3074–3110. MR3161458

[4] CANDÈS, E. J. and RECHT, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational Mathematics* **9** 717–772. MR2565240

[5] COLLINS, M., DASGUPTA, S. and SCHAPIRE, R. E. (2001). A generalization of principal components analysis to the exponential family. In *Advances in Neural Information Processing Systems* 617–624.

[6] CONSORTIUM, T. I. H. (2005). A haplotype map of the human genome. *Nature* **437** 1299–1320.

[7] DE LEEUW, J. (2006). Principal component analysis of binary data by iterated singular value decomposition. *Computational Statistics & Data Analysis* **50** 21–39. MR2196220

[8] ECKART, C. and YOUNG, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika* **1** 211–218.

[9] FAN, J. and LI, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association* **96** 1348–1360. MR1946581

[10] HOFMANN, T. (1999). Probabilistic latent semantic indexing. 50–57. ACM.

[11] JOHNSTONE, I. M. and LU, A. Y. (2009). On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association* **104**. MR2751448

[12] JOLLIFFE, I. T., TRENDAFILOV, N. T. and UDDIN, M. (2003). A modified principal component technique based on the LASSO. *Journal of Computational and Graphical Statistics* **12** 531–547. MR2002634

[13] KOREN, Y. (2008). Factorization meets the neighborhood: A multifaceted collaborative filtering model. *ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining* 426–434.

[14] KRAMER, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal* **37** 233–243.

[15] LEE, S. and HUANG, J. Z. (2013). A coordinate descent MM algorithm for fast computation of sparse logistic PCA. *Computational Statistics & Data Analysis* **62** 26–38. MR3040260

[16] LEE, S., HUANG, J. Z. and HU, J. (2010). Sparse logistic principal components analysis for binary data. *The Annals of Applied Statistics* **4** 1579. MR2758342

[17] LINTING, M., MEULMAN, J. J., GROENEN, P. J. and VAN DER KOOJJ, A. J. (2007). Nonlinear principal components analysis: Introduction and application. *Psychological Methods* **12** 336.

[18] NADLER, B. (2008). Finite sample approximation results for principal component analysis: A matrix perturbation approach. *The Annals of Statistics* 2791–2817. MR2485013

[19] NESTEROV, Y. (1988). On an approach to the construction of optimal methods of minimization of smooth convex functions. *Ekonomika i Mateaticheskie Metody* **24** 509–517. MR0968064

[20] PARIKH, N. and BOYD, S. (2013). Proximal algorithms. *Foundations and Trends in Optimization* **1** 123–231.

[21] PAUL, D. (2007). Asymptotics of sample eigenstructure for a large dimensional spiked covariance model. *Statistica Sinica* **17** 1617. MR2399865

[22] ROUSSEEUW, P. J. and DRIESSEN, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics* **41** 212–223.

[23] SCHEIN, A. I., SAUL, L. K. and UNGAR, L. H. (2003). A generalized linear model for principal component analysis of binary data. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics* **38** 46.

[24] SHE, Y. (2009). Thresholding-based iterative selection procedures for model selection and shrinkage. *Electronic Journal of statistics* **3** 384–415. MR2501318

[25] SHE, Y. (2012). An iterative algorithm for fitting nonconvex penalized generalized linear models with grouped predictors. *Computational Statistics & Data Analysis* **56** 2976–2990. MR2929353

[26] SHE, Y. (2013). Reduced rank vector generalized linear models for feature extraction. *Statistics and Its Interface* **6** 197–209. MR3066685

[27] SHE, Y. (2015). Selective Factor Construction in High Dimensions. arXiv:1403.6212.

[28] SHE, Y., LI, S. and WU, D. (2015). Robust orthogonal complement principal component analysis. *Journal of the American Statistical Association (to appear)*.

[29] SHE, Y. and OWEN, A. B. (2011). Outlier detection using nonconvex penalized regression. *Journal of the American Statistical Association* **106**. MR2847975

[30] SHEN, H. and HUANG, J. (2008). Sparse principal component analysis via low rank matrix approximation. *Journal of Multivariate Analysis* **99** 1015–1034.

[31] TIBSHIRANI, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 267–288. MR1379242

[32] TSENG, P. (2001). Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications* **109** 475–494. MR1835069

[33] TSENG, P. (2008). On accelerated proximal gradient methods for convex-concave optimization. *Submitted to SIAM Journal on Optimization*.

[34] TSENG, P. (2010). Approximation accuracy, gradient methods, and error bound for structured convex optimization. *Mathematical Programming* **125** 263–295. MR2733565

[35] WEN, Z. and YIN, W. (2013). A feasible method for optimization with orthogonality constraints. *Mathematical Programming* 1–38. MR3127080

[36] WITTEN, D., TIBSHIRANI, R. and HASTIE, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Journal of Multivariate Analysis* **10** 515–534.

[37] ZHANG, C.-H. (2010). Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics* 894–942. MR2604701

[38] ZHANG, T. (2010). Analysis of multi-stage convex relaxation for sparse regularization. *The Journal of Machine Learning Research* **11** 1081–1107. MR2629825

[39] ZHAO, P. and YU, B. (2006). On model selection consistency of Lasso. *The Journal of Machine Learning Research* **7** 2541–2563. MR2274449

[40] ZOU, H., HASTIE, T. and TIBSHIRANI, R. (2006). Sparse principal component analysis. *Journal of Computational and Graphical Statistics* **15(2)** 265–286.

Qiaoya Zhang
Department of Statistics
Florida State University
Tallahassee, FL 32306-4330
USA
E-mail address: qzhang@stat.fsu.edu

Yiyuan She
Department of Statistics
Florida State University
Tallahassee, FL 32306-4330
USA
E-mail address: yshe@stat.fsu.edu