Robust Streaming PCA via Percentile Thresholding

David Paul Fleischer,

Department of Mathematics and Statistics,

McGill University,

Montreal, Quebec, Canada

August, 2020

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Master of Science.

Contents

Abstract

Principal component analysis (PCA) is a popular statistical tool typically presented as a dimensionality reduction algorithm. PCA's strengths apply when high-dimensional data is said to roughly lie on a lower-dimensional subspace, allowing for the disposal of non-essential dimensions in favour of representing the inputs more parsimoniously. The procedure is based upon the search for an orthogonal linear subspace spanning a lower dimension than the input dimension. In the classical setting this can be accomplished by performing a singular value decomposition (SVD) on the sample covariance matrix. However, it is this reliance on the SVD that bear out two major weaknesses of PCA. The first such weakness are its computational restrictions. Computing the sample covariance and the following SVD may be infeasible in situations where the volume of data is too large, or simply inappropriate in situations where we wish to compute live updates associated with some serialized data stream. The next weakness is PCA's fragility to outliers deviating off the true subspace.

This thesis first aims to review several existing solutions to these problems by exploring the development of streaming/stochastic algorithms addressing its computational limitations, and the development of robust algorithms addressing its sensitivity, as well as implementations of algorithms that are both robust and streaming. Following this review a novel robust streaming estimator is presented based on a percentile-based hard-thresholding function, presented as a streaming implementation of the robust PCA algorithm developed by Zhang and Yang (2018).

Résumé

L'analyse en composantes principales (ACP) est un outil statistique populaire généralement présenté comme un algorithme de réduction de la dimensionnalité. Les points forts de l'ACP s'appliquent lorsque des données à haute dimension se trouvent en gros dans un sous-espace à plus faible dimension, ce qui permet d'éliminer des dimensions non essentielles au profit d'une représentation plus parcimonieuse des entrées. La procédure est basée sur la recherche d'un sous-espace linéaire orthogonal couvrant une dimension inférieure à la dimension d'entrée. Dans le cas classique, cela peut être accompli en effectuant une décomposition en valeur singulière (DVS) sur la matrice de covariance de l'échantillon. Cependant, c'est cette dépendance à l'égard de la DVS qui confirme deux faiblesses majeures de l'ACP. La première de ces faiblesses réside dans ses restrictions de calcul. Le calcul de la covariance de l'échantillon et de la DVS suivante peut s'avérer impossible dans les cas où le volume de données est trop important, ou simplement inapproprié dans les situations où nous souhaitons calculer des mises à jour en direct associées à un flux de données sérialisées. L'autre faiblesse est la fragilité de l'ACP aux aberrations s'écartant du véritable sous-espace.

Cette thèse vise d'abord à examiner plusieurs solutions existantes à ces problèmes en explorant le développement d'algorithmes de streaming/stochastique répondant à ses limites de calcul, et le développement d'algorithmes robustes répondant à sa sensibilité, ainsi que les implémentations d'algorithmes qui sont à la fois robustes et de streaming. Suite à cet examen, un nouvel estimateur robuste de streaming est présenté, basé sur une fonction de seuillage dur au centile, et est présenté comme une implémentation de streaming de l'algorithme PCA robuste développé par Zhang et Yang (2018).

Acknowledgements

I begin by extending my unyielding gratitude to my supervisors Dr. David Stephens and Dr. Yi Yang for their extrodinary patience and guidance. There is no doubt in my mind that without their direction this thesis would not have been possible. I would also like to thank Dr. Qiang Sun for taking the time to act as external examiner of this thesis.

Throughout my graduate studies I have had the pleasure of meeting many brilliant colleagues, all of whom have inspired me and are a continued source of motivation for me to draw upon. I would like to extend personal thanks to Nicholas Beck, James McVittie, Aram Pooladian, and Tyrel Stokes for their clarifying advice and feedback.

List of Figures

Chapter 1 Introduction

1.1 Motivation

Trends toward large data regimes have seen data collection and storage volumes grow more rapidly than the computing power needed for processing and analysis (Yelick, 2017). In addressing this problem it is a natural question to ask whether there exists some pre-processing transformation which permits the user to "compress" the data without a "material loss of information", both defined herein. Principal component analysis (PCA) is a popular tool for such a transformation following its independent derivation by Pearson (1901) and Hotelling (1933). In the context of PCA we may interpret "compression" to refer to carrying out a linear transformation on the data from its original higher-dimensional space to a lower-dimensional subspace, and "without material loss of information" to refer to minimizing the loss between the original data and its low-dimensional projection. Other interpretations exist and will be examined in later sections.

In situations where the number of observations is high or the dimension of the original space is large, it may become impractical or infeasible to compute the PCA estimator using the entire sample of observations. In such situations *streaming* algorithms are offered as a solution. We refer to an algorithm as *streaming* (also known as *stochastic* or *online*) if it computes iterative updates towards the true estimator using only a single observation at a time. We contrast such algorithms with *offline* algorithms (also known as *batch* algorithms) which have full access to the sample of observations when providing an update rule to an estimator.

In Section 2 we will review the classical description of PCA in both offline and online settings. In Section 3 we motivate the need for robustness and review existing offline and online robust PCA procedures. We then present the robust algorithm from (Zhang and Yang, 2018) and build upon it to derive a novel robust streaming PCA algorithm. In Section 4 we demonstrate the performance of this algorithm in a number of different settings and discuss its efficacy as a robust streaming solution to the problem of low-rank subspace estimation.

Chapter 2 Principal Component Analysis

2.1 Classical PCA

Consider a set of D potentially correlated random variables

$$\mathbf{x} = (x_1, ..., x_D)^T,$$

with covariance matrix $\mathbf{\Sigma_{xx}}$. Without loss of generality assume $\mathbb{E}[\mathbf{x}] = 0$. Principal component analysis seeks to generate a set of d ordred and uncorrelated linear projections

$$\mathbf{z} = (z_1, ..., z_d)^T, \quad 1 \leq d \leq D,$$

from the D the input variables $\mathbf{x}$. The imposition of linearity on the transformation of $\mathbf{x}$ informs us that each composite variable z_j can be written as

$$z_j = v_{j1}x_1 + \cdots + v_{jD}x_D = \mathbf{v}_j^T\mathbf{x}, \quad j = 1, ..., d,$$

where $\mathbf{v}_j = (v_{j1}, ..., v_{jD})^T$ is a $(D \times 1)$-dimensional vector of projection coefficients, known as the j^{th} *principal component*. The PCA algorithm determines the d principal components $(\mathbf{v}_1, ..., \mathbf{v}_d)$ such that

1. the d linear projections $(z_1, ..., z_d)$ of $\mathbf{x}$ are ranked according to their variance

$$\text{Var}(z_j) = \text{Var}(\mathbf{v}_j^T\mathbf{x}) = \mathbf{v}_j^T\text{Var}(\mathbf{x})\mathbf{v}_j = \mathbf{v}_j^T\mathbf{\Sigma_{xx}}\mathbf{v}_j,$$

   in decreasing order, $\text{Var}(z_1) \geq \cdots \geq \text{Var}(z_d)$, and

2. each z_j is orthogonal to all other z_k, $k \neq j$, i.e.

$$0 = \text{Cov}(z_j, z_k) = \mathbb{E}\left[(z_j - \mathbb{E}[z_j])(z_j - \mathbb{E}[z_k])\right] = \mathbb{E}\left[z_j z_k\right] = \mathbb{E}\left[\mathbf{v}_j^T\mathbf{x}\mathbf{x}^T\mathbf{v}_k\right] = \mathbf{v}_j^T\mathbf{\Sigma_{xx}}\mathbf{v}_k.$$

To determine a unique $(D \times d)$-dimensional subspace parameterized by this linear transformation $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_d]$ we must impose some optimality condition on the principal components. Two popular derivations of PCA view such a condition as either a variance-maximization problem or a least-squares optimization problem. The ladder is analogous to the "material loss of information" interpretation of PCA discussed above while the former can be thought as seeking the linear combinations $\mathbf{z}$ retaining the greatest amount of information of its inputs $\mathbf{x}$ under a low-rank constraint. The details of both derivations, as well as their relationship to each other, are discussed next.

2.1.1 PCA as a Variance Maximization Problem

2.1.1.1 Iterative Algorithm

As stated above, PCA seeks to generate an orthogonal linear transformation of its input variables and rank these projections in order of decreasing variance. Therefore, if we wish to determine the first principal component z_1 we must solve

$$\underset{\mathbf{v}_1 \in \mathbb{R}^D}{\text{maximize}} \ \text{Var}(z_1) \quad \text{s.t.} \quad \mathbf{v}_1^T \mathbf{v}_1 = 1.$$

Noting that

$$\text{Var}(z_1) = \mathbf{v}_1^T \mathbf{x}\mathbf{x}^T \mathbf{v}_1 = \mathbf{v}_1^T \mathbf{\Sigma}_{\mathbf{xx}} \mathbf{v}_1,$$

we more typically express this maximization problem as

$$\underset{\mathbf{v}_1 \in \mathbb{R}^D}{\text{maximize}} \ \mathbf{v}_1^T \mathbf{\Sigma}_{\mathbf{xx}} \mathbf{v}_1 \quad \text{s.t.} \quad \mathbf{v}_1^T \mathbf{v}_1 = 1.$$

Note that the normalization constraint must be introduced in order to avoid unbounded solutions $\|\mathbf{v}_1\| \to \infty$ and that orthogonality cannot yet be enforced since no other principal components are in consideration. Define the Lagrangian function f corresponding to the above optimization problem,

$$f(\mathbf{v}_1) = \mathbf{v}_1^T \mathbf{\Sigma}_{\mathbf{xx}} \mathbf{v}_1 + \lambda_1 (1 - \mathbf{v}_1^T \mathbf{v}_1),$$

where λ_1 is the Lagrange multiplier. Differentiating f and setting the result to zero,

$$\frac{\partial f(\mathbf{v}_1)}{\partial \mathbf{v}_1} = 2 \left(\mathbf{\Sigma}_{\mathbf{xx}} - \lambda_1 \mathbb{I}_D \right) \mathbf{v}_1 = \mathbf{0}.$$

From elementary linear algebra, the above equation has a nonzero solution $\mathbf{v}_1$ if and only if the determi-

nant of $\mathbf{\Sigma_{xx}} - \lambda_1 \mathbb{I}_D$ is zero. Therefore, if $\mathbf{v}_1$ is nondegenerate, and $(\lambda_1, \mathbf{v}_1)$ satisfy

$$\mathbf{\Sigma_{xx}} \mathbf{v}_1 = \lambda_1 \mathbf{v}_1,$$

then λ_1 must be largest eigenvalue of $\mathbf{\Sigma_{xx}}$ (since we are maximizing f) and $\mathbf{v}_1$ its correspoding eigenvector. Substituting this solution into the original objective yields

$$\text{Var}(z_1) = \mathbf{v}_1^T \mathbf{\Sigma_{xx}} \mathbf{v}_1 = \mathbf{v}_1^T \lambda_1 \mathbf{v}_1 = \lambda_1,$$

informing us that the variance of z_1 is precisely equal to the largest eigenvalue of $\mathbf{\Sigma_{xx}}$. To find the second principal component $\mathbf{v}_2$ we now solve a similar constrained problem to that of $\mathbf{v}_1$, with the addition of an orthogonality constraint $\mathbf{v}_1^T \mathbf{v}_2 = 0$, i.e.

$$\underset{\mathbf{v}_2 \in \mathbb{R}^D}{\text{maximize}} \ \text{Var}(z_2) \quad \text{s.t.} \quad \mathbf{v}_2^T \mathbf{v}_2 = 1, \ \mathbf{v}_1^T \mathbf{v}_2 = 0.$$

Form the associated Lagrangian function

$$f(\mathbf{v}_2) = \mathbf{v}_2^T \mathbf{\Sigma_{xx}} \mathbf{v}_2 + \lambda_2 (1 - \mathbf{v}_2^T \mathbf{v}_2) + \mu \mathbf{v}_1^T \mathbf{v}_2,$$

now with the additional multiplier of μ associated with the orthogonality constraint. Differentiating with respect to $\mathbf{v}_2$ and setting the result to zero,

$$\frac{\partial f(\mathbf{v}_2)}{\partial \mathbf{v}_2} = 2 \left(\mathbf{\Sigma_{xx}} - \lambda_2 \mathbb{I}_D \right) \mathbf{v}_2 + \mu \mathbf{v}_1 = \mathbf{0}. \tag{2.1}$$

Left-multiplying both sides of (2.1) by $\mathbf{v}_1^T$ yields

$$\begin{aligned} 0 &= 2 \mathbf{v}_1^T \left(\mathbf{\Sigma_{xx}} - \lambda_2 \mathbb{I}_D \right) \mathbf{v}_2 + \mu \mathbf{v}_1^T \mathbf{v}_1 \\ &= 2 \mathbf{v}_1^T \mathbf{\Sigma_{xx}} \mathbf{v}_2 + \mu, \end{aligned}$$

while left-multiplying both sides of (2.1) by $\mathbf{v}_2^T$ yields

$$\begin{aligned} 0 &= 2 \mathbf{v}_2^T \left(\mathbf{\Sigma_{xx}} - \lambda_2 \mathbb{I}_D \right) \mathbf{v}_2 + \mu \mathbf{v}_2^T \mathbf{v}_1 \\ &= \mathbf{v}_2^T \mathbf{\Sigma_{xx}} \mathbf{v}_2. \end{aligned}$$

Summing these two results,

$$\begin{aligned} 0 &= 2\mathbf{v}_1^T \boldsymbol{\Sigma}_{\mathbf{xx}} \mathbf{v}_2 + \mu + \mathbf{v}_2^T \boldsymbol{\Sigma}_{\mathbf{xx}} \mathbf{v}_2 \\ &= 0 + \mu + 0 \\ &= \mu. \end{aligned}$$

Plugging this value for μ into the original derivative gives

$$\frac{\partial f(\mathbf{v}_2)}{\partial \mathbf{v}_2} = 2\left(\boldsymbol{\Sigma}_{\mathbf{xx}} - \lambda_2 \mathbb{I}_D\right)\mathbf{v}_2 = \mathbf{0}.$$

Similar to the first principal component, this equation informs us that λ_2 must be the next largest eigenvalue of $\boldsymbol{\Sigma}_{\mathbf{xx}}$ with corresponding eigenvector $\mathbf{v}_2$, and that $\text{Var}(z_2) = \lambda_2$.

Continuing in this way we can iteratively derive the remaining principal components $z_3, ..., z_d$:

1. Choose $\mathbf{v}_j$ so that $z_j = \mathbf{v}_j^T \mathbf{x}$ has its variance maximized and is uncorrelated with all previous projections $z_1, ..., z_{j-1}$.

2. The coefficients $\mathbf{v}_j$ are given by the j^{th} eigenvector of $\boldsymbol{\Sigma}_{\mathbf{xx}}$, associated with the j^{th} largest eigenvalue λ_j.

3. The variance of the j^{th} component is given by the j^{th} eigenvalue λ_j so that by construction $\text{Var}(z_1) \geq \cdots \geq \text{Var}(z_d)$.

2.1.1.2 Matrix Algorithm

It is typically more convient to express the variance maximization problem as a simultaneous procedure over the matrix of coefficients $\mathbf{V}$ rather than a sequence of procedures over vectors $\mathbf{v}_1, ..., \mathbf{v}_d$. We are able to express the d transformed variables $z_1, ..., z_d$ more succinctly using the matrix product

$$\mathbf{z} = (z_1, ..., z_d)^T = \mathbf{V}^T \mathbf{x},$$

where $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_d] \in \mathbb{R}^{D \times d}$ is a linear transformation aspiring to contain the principal components of $\mathbf{x}$. We can combine the d objective functions from the iterative procedure by noting that

$$\max_{\mathbf{v}_1, ..., \mathbf{v}_d} \sum_{j=1}^{d} \text{Var}(z_j) = \max_{\mathbf{v}_1, ..., \mathbf{v}_d} \text{trace}\left(\boldsymbol{\Sigma}_{\mathbf{zz}}\right),$$

but

$$\mathbf{\Sigma_{zz}} = \text{Var}(\mathbf{z}) = \text{Var}(\mathbf{V}^T\mathbf{x}) = \mathbf{V}^T\text{Var}(\mathbf{x})\mathbf{V} = \mathbf{V}^T\mathbf{\Sigma_{xx}}\mathbf{V}$$

so

$$\begin{aligned}\max_{\mathbf{v}_1,...,\mathbf{v}_d} \sum_{j=1}^{d} \text{Var}(z_j) &= \max_{\mathbf{v}_1,...,\mathbf{v}_d} \text{trace}\left(\mathbf{\Sigma_{zz}}\right), \\ &= \max_{\mathbf{V}} \text{trace}\left(\mathbf{V}^T\mathbf{\Sigma_{xx}}\mathbf{V}\right).\end{aligned}$$

This final quantity is sometimes written even more succinctly using the Frobenius inner product $\langle \mathbf{A}, \mathbf{B}\rangle_F = \text{trace}(\mathbf{A}^T\mathbf{B})$,

$$\begin{aligned}\max_{\mathbf{V}} \text{trace}\left(\mathbf{V}^T\mathbf{\Sigma_{xx}}\mathbf{V}\right) &= \max_{\mathbf{V}} \text{trace}\left(\mathbf{V}\mathbf{V}^T\mathbf{\Sigma_{xx}}\right) \quad \text{(cyclic property of the trace)} \\ &= \max_{\mathbf{V}} \langle \mathbf{V}\mathbf{V}^T, \mathbf{\Sigma_{xx}}\rangle_F.\end{aligned}$$

Next, we can express the sequence of normalization and orthogonality constraints

$$\begin{aligned}\mathbf{v}_j^T\mathbf{v}_j &= 1, \quad j = 1, ..., d \\ \mathbf{v}_j^T\mathbf{v}_k &= 0, \quad j \neq k,\end{aligned}$$

as the product

$$\mathbf{V}^T\mathbf{V} = \mathbb{I}_d.$$

That is, the sequence of constraints in the iterative algorithm is equivalent to restricting $\mathbf{V}$ to orthonormal $(D \times d)$ linear transformations. Putting the objective and constraint together we are able to express the complete variance maximization problem as the following constrained problem

$$\underset{\mathbf{V}}{\text{maximize}} \text{ trace}\left(\mathbf{V}^T\mathbf{\Sigma_{xx}}\mathbf{V}\right) \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V} = \mathbb{I}_d. \tag{2.2}$$

Recall that the iterative algorithm's solutions $\mathbf{v}_1, ..., \mathbf{v}_d$ were precisely the solutions to the (ordered) eigenvalue-vector equations

$$\mathbf{\Sigma_{xx}}\mathbf{v}_j = \lambda_j\mathbf{v}_j, \quad j = 1, ..., d,$$

corresponding to the largest d eigenvalues. Therefore, the solutions can be expressed as the simultaneous

system

$$\mathbf{\Sigma_{xx}V} = \mathbf{\Lambda V},$$

where $\mathbf{\Lambda} = \text{diag}\,(\lambda_1, ..., \lambda_d)$ is a matrix containing the ordered eigenvalues of $\mathbf{\Sigma_{xx}}$ along the diagonal and zeros elsewhere. Letting $\mathbf{V}^* \in \mathbb{R}^{D\times d}$ denote the optimal subspace defined by the first d eigenvectors of $\mathbf{\Sigma_{xx}}$, the maximized variance contained by the first d projections is

$$\begin{aligned}
\max_{\mathbf{v}_1,...,\mathbf{v}_d} \sum_{j=1}^{d} \text{Var}(z_j) &= \max_{\mathbf{V}\in\mathbb{R}^{D\times d}} \text{trace}\left(\mathbf{V}^T\mathbf{\Sigma_{xx}V}\right) \\
&= \text{trace}\left(\mathbf{V}^{*T}\left(\mathbf{V}^*\mathbf{\Lambda V}^{*T}\right)\mathbf{V}^*\right) \\
&= \text{trace}(\mathbf{\Lambda}) \\
&= \sum_{j=1}^{d} \lambda_j.
\end{aligned}$$

That is, the maximized variance of the first d principal components is precisely the sum of the largest d eigenvalues of the input variables' covariance matrix. This result is sometimes more generally referred to as *Ky Fan's maximum principle* (Fan, 1949, 1950; Vu et al., 2013a).

2.1.2 PCA as a Least-Squares Problem

In the least-squares setting we PCA as being tasked with finding some "encoding" function f that generates a lower-dimensional transformation of its inputs $\mathbf{x}$ (Goodfellow et al., 2016). Let $\mathbf{z}$ denote this d-dimensional encoding of $\mathbf{x}$ given by f, i.e.

$$\mathbf{z} = (z_1, ..., z_d)^T = f(\mathbf{x}) = f((x_1, ..., x_D)^T).$$

In addition to the encoding function f, PCA must also find a corresponding "decoding" function g that allows us to approximately reconstruct inputs $\mathbf{x}$ given the encoded variables $\mathbf{z}$,

$$\mathbf{x} \approx g(\mathbf{z}) = g(f(\mathbf{x})).$$

Principal component analysis restricts the decoder g to the set of linear transformations

$$g(\mathbf{z}) = \mathbf{Vz},$$

where $\mathbf{V} \in \mathbb{R}^{D \times d}$ and whose columns are orthonormal,

$$\mathbf{V}^T\mathbf{V} = \mathbb{I}_d.$$

As is no surprise in the *least-squares* setting, we operationalize "approximate reconstruction" of inputs $\mathbf{x}$ to be the squared ℓ_2-norm of the difference between $\mathbf{x}$ and its reconstruction $g(\mathbf{z}) = \mathbf{V}\mathbf{z}$. This gives rise to the following least-squares objective

$$\min_{\mathbf{z}} \mathbb{E}\left[\|\mathbf{x} - g(\mathbf{z})\|_2^2\right] = \min_{\mathbf{z}} \mathbb{E}\left[\|\mathbf{x} - \mathbf{V}\mathbf{z}\|_2^2\right],$$

where we have assumed $\mathbf{V}$ to be fixed for the time being. We can expand this objective as

$$\begin{aligned}
\|\mathbf{x} - \mathbf{V}\mathbf{z}\|_2^2 &= (\mathbf{x} - \mathbf{V}\mathbf{z})^T (\mathbf{x} - \mathbf{V}\mathbf{z}) \\
&= \mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{V}\mathbf{z} + (\mathbf{V}\mathbf{z})^T \mathbf{V}\mathbf{z} \\
&= \mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{V}\mathbf{z} + \mathbf{z}^T\mathbf{V}^T\mathbf{V}\mathbf{z} \\
&= \mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{V}\mathbf{z} + \mathbf{z}^T\mathbf{z}.
\end{aligned}$$

Computing the gradient $\nabla_{\mathbf{z}}$

$$\begin{aligned}
\nabla_{\mathbf{z}}\|\mathbf{x} - \mathbf{V}\mathbf{z}\|_2^2 &= \nabla_{\mathbf{z}}\left(\mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{V}\mathbf{z} + \mathbf{z}^T\mathbf{z}\right) \\
&= -2\mathbf{V}^T\mathbf{x} + 2\mathbf{z},
\end{aligned}$$

and setting this to $\mathbf{0}$ yields the first order condition,

$$\begin{aligned}
&\mathbf{0} = \nabla_{\mathbf{z}}\|\mathbf{x} - \mathbf{V}\mathbf{z}\|_2^2 \\
\iff &\mathbf{0} = -2\mathbf{V}^T\mathbf{x} + 2\mathbf{z} \\
\iff &\mathbf{z} = \mathbf{V}^T\mathbf{x}.
\end{aligned}$$

Therefore, the assumption that decoder g is linear leads to the least-squares optimal encoder f given by

$$\mathbf{z} = f(\mathbf{x}) = \mathbf{V}^T\mathbf{x}.$$

Hence, the d-dimensional reconstruction of $\mathbf{x}$ defined by PCA through the least-squares procedure is

$$g(f(\mathbf{x})) = g(\mathbf{V}^T\mathbf{x}) = \mathbf{V}\mathbf{V}^T\mathbf{x}.$$

In the least-squares context it is not uncommon to see the orthogonality constraint on $\mathbf{V}$ to be presented instead as low-rank constraint on orthogonal projection matrices $\mathbf{W}$. A square matrix $\mathbf{W} \in \mathbb{R}^{D\times D}$ is said to be an *orthogonal projection matrix* if it is both idempotent and symmetric,

$$\mathbf{W}^2 = \mathbf{W}^T = \mathbf{W}.$$

With such a matrix $\mathbf{W}$ the restriction $\mathbf{V}^T\mathbf{V} = \mathbb{I}_d$ is equivalent to restricting $\mathbf{W}$ so that $\text{rank}(\mathbf{W}) = d$ (Chang et al., 2014; Nie et al., 2016; Warmuth and Kuzmin, 2008). To see this equivalence we interpret $\mathbf{W}$ as the linear transformation mapping inputs $\mathbf{x}$ onto a low-rank subspace,

$$g(f(\mathbf{x})) = \mathbf{W}\mathbf{x} = \mathbf{V}\mathbf{V}^T\mathbf{x} \approx \mathbf{x},$$

and if $\mathbf{V} \in \mathbb{R}^{D\times d}$, then $\text{rank}(\mathbf{V}\mathbf{V}^T) = \text{rank}(\mathbf{V}) = d$. Secondly, $\left(\mathbf{V}\mathbf{V}^T\right)^T = \mathbf{V}\mathbf{V}^T$ so

$$\left(\mathbf{V}\mathbf{V}^T\right)^2 = \left(\mathbf{V}\mathbf{V}^T\right)^T\left(\mathbf{V}\mathbf{V}^T\right) = \mathbf{V}\mathbf{V}^T\mathbf{V}\mathbf{V}^T = \mathbf{V}\mathbf{V}^T,$$

i.e. $\mathbf{V}\mathbf{V}^T$ is indeed an idempotent, symmetric, rank-d matrix, admitting it into the set of orthogonal projections. Therefore, any solution $\mathbf{V}^*$ to the constrained least-squares problem

$$\mathbf{V}^* = \arg\min_{\mathbf{V}} \mathbb{E}\left[\|\mathbf{x} - \mathbf{V}\mathbf{V}^T\mathbf{x}\|_2^2\right] \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V} = \mathbb{I}_d$$

must also solve the problem

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \mathbb{E}\left[\|\mathbf{x} - \mathbf{W}^T\mathbf{x}\|_2^2\right] \quad \text{s.t.} \quad \mathbf{W}^2 = \mathbf{W},\ \text{rank}(\mathbf{W}) = d$$

according to $\mathbf{W}^* = \mathbf{V}^*\mathbf{V}^{*T}$.

2.1.3 Least-Squares and Variance Maximization Equivalence

We now wish to show that the two interpretations of PCA are indeed equivalent. Given inputs $\mathbf{x} = (x_1, ..., x_D)^T$ and a $(D \times d)$-dimensional subspace parameterized by $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_d]$, we express a lower-

dimensional set of approximations $\mathbf{z} = (z_1, ..., z_d)^T$ as the linear combinations $\mathbf{z} = \mathbf{V}^T\mathbf{x}$. We begin from the least-squares procedure

$$\underset{\mathbf{V}}{\text{minimize}}\ \mathbb{E}\left[\|\mathbf{x} - \mathbf{V}\mathbf{V}^T\mathbf{x}\|_2^2\right] \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V} = \mathbb{I}_d.$$

Expanding the objective

$$\begin{aligned}
\min_{\mathbf{V}} \mathbb{E}\left[\|\mathbf{x} - \mathbf{V}\mathbf{V}^T\mathbf{x}\|_2^2\right] &= \min_{\mathbf{V}} \mathbb{E}\left[\left(\mathbf{x} - \mathbf{V}\mathbf{V}^T\mathbf{x}\right)^T\left(\mathbf{x} - \mathbf{V}\mathbf{V}^T\mathbf{x}\right)\right] \\
&= \min_{\mathbf{V}} \mathbb{E}\left[\mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x} - \mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x} + \mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{V}\mathbf{V}^T\mathbf{x}\right] \\
&= \min_{\mathbf{V}} \mathbb{E}\left[-2\mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x} + \mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x}\right] \\
&= \min_{\mathbf{V}} \mathbb{E}\left[-\mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x}\right] \\
&= \max_{\mathbf{V}} \mathbb{E}\left[\mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x}\right].
\end{aligned}$$

Since $\mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x}$ is a one-dimensional scalar we can express it as the trivial trace

$$\max_{\mathbf{V}} \mathbb{E}\left[\mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x}\right] = \max_{\mathbf{V}} \mathbb{E}\left[\text{trace}\left(\mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x}\right)\right].$$

Using the cyclic property of the trace operator,

$$\begin{aligned}
\max_{\mathbf{V}} \mathbb{E}\left[\text{trace}\left(\mathbf{x}^T\mathbf{V}\mathbf{V}^T\mathbf{x}\right)\right] &= \max_{\mathbf{V}} \mathbb{E}\left[\text{trace}\left(\mathbf{V}^T\mathbf{x}\mathbf{x}^T\mathbf{V}\right)\right] \\
&= \max_{\mathbf{V}} \mathbb{E}\left[\text{trace}\left(\mathbf{z}\mathbf{z}^T\right)\right] \\
&= \max_{\mathbf{V}} \text{trace}\left(\mathbb{E}\left[\mathbf{z}\mathbf{z}^T\right]\right) \quad \text{(linearity of the trace)} \\
&= \max_{\mathbf{V}} \text{trace}\left(\text{Var}\left(\mathbf{z}\right)\right),
\end{aligned}$$

where the final line was achieved by recalling that our inputs $\mathbf{x}$ are assumed to have mean zero. This informs us that minimizing the reconstruction error $\|\mathbf{x} - \mathbf{V}\mathbf{V}^T\mathbf{x}\|_2^2$ given by the least-squares optimal $\mathbf{V}$ is indeed equivalent to maximizing the variance explained by the principal directions $\mathbf{z}$. That is, the d-dimensional subspace minimizing the reconstruction error is identical to the subspace maximizing the variance contained in the linear projections of the input variables.

2.1.4 Sample Estimators

Classical PCA is commonly presented as an offline linear dimensionality reduction technique using n samples drawn from a D-dimensional distribution, $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^T \in \mathbb{R}^{n \times D}$, $\mathbf{x}_i = (x_{i1}, ..., x_{iD}) \in \mathbb{R}^D$. Since we

rarely have access to the underlying covariance $\mathbf{\Sigma_{xx}}$ we begin by estimating it with the sample covariance $\widehat{\mathbf{\Sigma}}_{\mathbf{xx}} = \frac{1}{n}\mathbf{X}^T\mathbf{X}$, column-centering $\mathbf{X}$ if its columns are not known to be drawn from a mean-zero distribution. Then, letting $\mathbf{Z} \in \mathbb{R}^{n\times D}$ denote the projected samples,

$$\mathbf{Z} = \mathbf{XV},$$

we estimate the variance contained by such a transformation according to

$$\widehat{\mathrm{Var}}\left(\mathbf{Z}\right) = \frac{1}{n}\mathbf{Z}^T\mathbf{Z} = \frac{1}{n}\mathbf{V}^T\mathbf{X}^T\mathbf{XV}.$$

Therefore, the offline optimization problem is to find the subspace, parameterized by $\mathbf{V} \in \mathbb{R}^{D\times d}$, which maximizes the empirical variance of the projections $\mathbf{Z}$,

$$\underset{\mathbf{V}}{\text{maximize}}\ \text{trace}(\mathbf{V}^T\mathbf{X}^T\mathbf{XV}) \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V} = \mathbb{I}_d, \tag{2.3}$$

or, equivalently, which minimizes the distance between the sample matrix $\mathbf{X}$ and its reconstruction $\mathbf{VV}^T\mathbf{X}$,

$$\underset{\mathbf{V}}{\text{minimize}}\ \|\mathbf{X} - \mathbf{VV}^T\mathbf{X}\|_F^2 \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V} = \mathbb{I}_d, \tag{2.4}$$

where we are now considering the least-squares criterion using the Frobenius norm because we are minimizing a difference of matrices rather than a difference of vectors.

In much the same manner as the population case, it can be shown that optimization problems (2.3) and (2.4) are solved when the optimal $\mathbf{V}^*$ has columns set to the first d eigenvectors of the covariance matrix $\frac{1}{n}\mathbf{X}^T\mathbf{X} = \mathbf{V\Lambda V}^T$, with corresponding first d eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$ (Julian, 2008; Goodfellow et al., 2016). Analogously to the population case, such eigenvalues are equal to the variance of the data along the direction of the associated eigenvector. Therefore, the d-dimensional subspace maximizing the empirical variance is given by the first d eigenvectors associated with the largest d eigenvalues. Algorithm (1) outlines the process of using the eigendecomposition on the sample covariance matrix to find the PCA estimator.

Alternatively, we could use the singular value decomposition (SVD) on the (column-centered) samples $\mathbf{X}$ directly instead of the eigendecomposition on the sample covariance matrix $n^{-1}\mathbf{X}^T\mathbf{X}$. This application of the SVD is a direct result of the Eckart-Young theorem (Eckart and Young, 1936; Mirsky, 1960) which states that if $\mathbf{X}$ has the SVD

$$\mathbf{X} = \mathbf{U\Sigma V}^T,$$

Algorithm 1: Offline/Sample PCA Algorithm

Inputs: A sample of n observations $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^T \in \mathbb{R}^{n \times D}$ drawn from a D-dimensional distribution, integer d corresponding to the desired (lower) dimension.

1 Estimate the columnwise means $\mu_{\mathbf{X}}$ by its estimator $\hat{\mu}_{\mathbf{X}} = \bar{\mathbf{x}} = n^{-1} \sum_{i=1}^{n} \mathbf{x}_i$.

2 Columnwise-center $\mathbf{X}$ according to $\mathbf{x}_i^{(c)} = \mathbf{x}_i - \bar{\mathbf{x}}$, $i = 1, ..., n$ so that $\mathbf{X}_c = \left[\mathbf{x}_1^{(c)}, ..., \mathbf{x}_n^{(c)}\right]^T$.

3 Estimate $\mathbf{\Sigma}_{\mathbf{xx}}$ by the sample covariance matrix $\widehat{\mathbf{\Sigma}}_{\mathbf{xx}} = n^{-1} \mathbf{X}_c^T \mathbf{X}_c$.

4 Compute the first d ordered eigenvalues of $\widehat{\mathbf{\Sigma}}_{\mathbf{xx}}$, $\hat{\lambda}_1, ..., \hat{\lambda}_d \geq 0$, and associated eigenvectors $\hat{\mathbf{v}}_1, ..., \hat{\mathbf{v}}_d$.

5 Compute the optimal rank-d reconstruction of some observation $\mathbf{x}' \in \mathbb{R}^D$,

$$\hat{\mathbf{x}}^{(d)} = \bar{\mathbf{x}} + \left(\sum_{j=1}^{d} \hat{\mathbf{v}}_j \hat{\mathbf{v}}_j^T \right) (\mathbf{x}' - \bar{\mathbf{x}}).$$

6 Compute the j^{th} projection of some observation $\mathbf{x}'$,

$$\hat{z}_j = \hat{\mathbf{v}}_j^T (\mathbf{x}' - \bar{\mathbf{x}})$$

then the best rank-d approximation of a matrix $\mathbf{X} \in \mathbb{R}^{n \times D}$ with respect to the Frobenius loss

$$\underset{\mathbf{X}}{\text{minimize}} \, \|\mathbf{X} - \widehat{\mathbf{X}}\|_F^2 \quad \text{s.t.} \quad \text{rank}(\widehat{\mathbf{X}}) = d$$

is given by

$$\mathbf{X}^* = \mathbf{U}_{1:d} \mathbf{\Sigma}_{1:d} \mathbf{V}_{1:d}^T,$$

where $\mathbf{U}_{1:d} \in \mathbb{R}^{n \times d}$ and $\mathbf{V}_{1:d} \in \mathbb{R}^{D \times d}$ correspond to the first d columns of $\mathbf{U}$ and $\mathbf{V}$, and $\mathbf{\Sigma}_{1:d} \in \mathbb{R}^{d \times d}$ a diagonal matrix of the first d singular values. From this result is it easy to compute the first d linear combinations $\mathbf{Z}$,

$$\mathbf{Z} = \mathbf{X} \mathbf{V}_{1:d}^T = \mathbf{U}_{1:d} \mathbf{\Sigma}_{1:d}.$$

2.1.5 Convex Relaxation of the PCA Optimization Problem

Note that neither the variance-maximization problems

$$\underset{\mathbf{V} \in \mathbb{R}^{D \times d}}{\text{maximize}} \, \mathbb{E}\left[\text{trace}\left(\mathbf{V}^T \mathbf{x}\mathbf{x}^T \mathbf{V}\right)\right] \quad \text{s.t.} \quad \mathbf{V}^T \mathbf{V} = \mathbb{I}_d,$$

$$\underset{\mathbf{W} \in \mathbb{R}^{D \times D}}{\text{maximize}} \, \text{trace}\left(\mathbf{\Sigma}_{\mathbf{xx}} \mathbf{W}\right) \quad \text{s.t.} \quad \mathbf{W}^2 = \mathbf{W} = \mathbf{W}^T, \, \text{rank}(\mathbf{W}) = d$$

nor the equivalent least-squares problems

$$\begin{aligned}
&\underset{\mathbf{V}\in\mathbb{R}^{D\times d}}{\text{minimize}}\ \mathbb{E}\left[\|\mathbf{x}-\mathbf{V}\mathbf{V}^T\mathbf{x}\|_2^2\right] \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V}=\mathbb{I}_d,\\
&\underset{\mathbf{W}\in\mathbb{R}^{D\times D}}{\text{minimize}}\ \mathbb{E}\left[\|\mathbf{x}-\mathbf{W}\mathbf{x}\|_2^2\right] \quad \text{s.t.} \quad \mathbf{W}^2=\mathbf{W}=\mathbf{W}^T,\ \text{rank}(\mathbf{W})=d
\end{aligned}$$

are convex. In particular, the constraints imposed on $\mathbf{V}$ or $\mathbf{W}$ do not describe convex domains. To see why consider the pair of orthonormal matrices

$$\mathbb{I}_2=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}, \quad -\mathbb{I}_2=\begin{bmatrix}-1 & 0\\ 0 & -1\end{bmatrix}.$$

The convex combination $\lambda\mathbb{I}_2+(1-\lambda)\mathbb{I}_2$ is equal to the zero matrix for $\lambda=0.5$ and cannot be admitted to the set of orthonormal matrices. Furthermore, the set of low-rank orthogonal projections $\mathbf{W}$ provides no relief since the set of rank-d matrices is not convex (viz. $\lambda\mathbb{I}_1+(1-\lambda)\mathbb{I}_2$ has rank zero for $\lambda=0.5$).

Such nonconvexity is troublesome in situations where we wish to use gradient-based procedures to solve the PCA optimization problem. Consequently, convex relaxations of the PCA constraints have been used as a way of making PCA more amenable to the rich body of work in convex optimization (Boyd and Vandenberghe, 2004; Dattorro, 2010). In the context of the low-rank projection matrix constraint Chang et al. (2014); Hu et al. (2004); Wright et al. (2009) note that the *nuclear norm* (also known as the *trace norm* or *Schatten 1-norm*)

$$\|\mathbf{W}\|_*=\text{trace}\left(\sqrt{\mathbf{W}^T\mathbf{W}}\right)=\sum_i\sigma_i(\mathbf{W})$$

corresponds to the convex hull of rank-d matrices and can be used as a penalization term in the least-squares setting, mediating $\mathbf{W}$ (and thereby controlling $\mathbf{V}$) in a convex regime. This relaxation is more commonly specified by the constraint on $\mathbf{V}$ such that $\mathbf{V}\mathbf{V}^T$ has bound eigenvalues $\lambda_i(\mathbf{V}\mathbf{V}^T)\in[0,1]$ and $\text{trace}(\mathbf{V}\mathbf{V}^T)=d$ (Arora et al., 2012, 2013; Arora and Marinov, 2019; Candès and Tao, 2010; Goes et al., 2014; Journée et al., 2010; Kotłowski and Warmuth, 2015; Mianjy and Arora, 2018; Olfat and Aswani, 2019; Tibshirani, 2016; Vu et al., 2013a,b). To see that this is indeed a convex relaxation, first note that $\mathbf{V}^T\mathbf{V}=\mathbb{I}_d$ must clearly have d eigenvalues identically equal to one. Therefore $\mathbf{V}\mathbf{V}^T$ must also have d eigenvalues identically equal to one, with all others equal to zero.[1] Hence, any convex combination of rank-d projection matrices $\mathbf{V}\mathbf{V}^T$ must also be an element of the set of trace-d projection matrices with eigenvalues bound within 0 and 1. Indeed, this relaxation is in fact the convex hull of rank-d projection matrices, also

[1] If $A\in\mathbb{R}^{n\times p}$ has singular value decomposition $U\Sigma V^T$ then $A^TA=V\Sigma^2V^T$ while $AA^T=U\Sigma^2U^T$. Indicating that both A^TA and AA^T have common nonzero eigenvalues $\text{diag}(\Sigma^2)=(\sigma_1^2,...,\sigma^2_{\min(n,p)})$.

known as the *Fantope* of order d, (Dattorro, 2010; Hastie et al., 2015)

$$\mathcal{F}_d = \text{conv}\left\{\mathbf{V}\mathbf{V}^T \,|\, \mathbf{V} \in \mathbb{R}^{D\times d},\, \mathbf{V}^T\mathbf{V} = \mathbb{I}_d\right\} = \left\{\mathbf{W} \in \mathbb{S}^D \,|\, 0 \preceq \mathbf{W} \preceq \mathbb{I}_D,\, \text{trace}(\mathbf{W}) = d\right\},$$

with $\{\mathbf{V}\mathbf{V}^T \mid \mathbf{V} \in \mathbb{R}^{D\times d},\, \mathbf{V}^T\mathbf{V} = \mathbb{I}_d\}$ the set of extreme points of $\mathcal{F}_d$ (Overton and Womersley, 1992; Vu et al., 2013a). For a proof of this relationship between the set of rank-d orthogonal projections and the Fantope see Fillmore and Williams (1971). A vital property of the Fantope is that performing the convexified PCA problem with respect to elements $\mathbf{W} \in \mathcal{F}_d$ is in fact equivalent to the original nonconvex problem. An outline of the proof found in Overton and Womersley (1992) is as follows. We have

$$\begin{aligned}\text{trace}(\mathbf{\Sigma_{xx}}\mathbf{W}) &= \text{trace}(\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T\mathbf{W}) \\ &= \text{trace}(\mathbf{\Lambda}\mathbf{Q}^T\mathbf{W}\mathbf{Q}) \\ &= \text{trace}(\mathbf{\Lambda}\mathbf{W}'),\end{aligned}$$

with $\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$ the eigendecomposition of $\mathbf{\Sigma_{xx}}$ and $\mathbf{W}' := \mathbf{Q}^T\mathbf{W}\mathbf{Q}$. Note that $\mathbf{W} \in \mathcal{F}_d \iff \mathbf{W}' \in \mathcal{F}_d$ since both clearly have trace d and both share common eigenvalues

$$\begin{aligned}\det\left(\mathbf{Q}^T\mathbf{W}\mathbf{Q} - \lambda\mathbb{I}\right) &= \det\left(\mathbf{Q}^T\left(\mathbf{W} - \lambda\mathbb{I}\right)\mathbf{Q}\right) \\ &= \det\left(\mathbf{Q}^T\right)\det\left(\mathbf{W} - \lambda\mathbb{I}\right)\det\left(\mathbf{Q}\right) \\ &= \det\left(\mathbf{Q}^T\mathbf{Q}\right)\det\left(\mathbf{W} - \lambda\mathbb{I}\right) \\ &= \det\left(\mathbf{W} - \lambda\mathbb{I}\right).\end{aligned}$$

Then,

$$\text{trace}(\mathbf{\Lambda}\mathbf{W}') = \sum_{i=1}^{D}\lambda_i w'_{ii}$$

achieves maximum value $\sum_{i=1}^{d}\lambda_i$ from the conditions on $\mathbf{W}'$. However, this was precisely the same maximal value for the original problem over orthonormal $(D \times d)$ matrices, i.e.

$$\sum_{i=1}^{d}\lambda_i = \max_{\mathbf{V}\,:\,\mathbf{V}^T\mathbf{V}=\mathbb{I}_d} \text{trace}\left(\mathbf{\Sigma_{xx}}\mathbf{V}\mathbf{V}^T\right) = \max_{\mathbf{W}\in\mathcal{F}_d} \text{trace}\left(\mathbf{\Sigma_{xx}}\mathbf{W}\right).$$

A geometric intuition for this equivalence can be presented in terms of the distance between points $\mathbf{x}$ in the original space and their subspace projections. The expansion the constraint set from orthogonal projections to the Fantope can be thought of as including all 'near-orthogonal' projections. Under this

interpretation of $\mathcal{F}_d$ it is clear that the minimizing projection from $\mathcal{F}_d$ must itself be orthogonal since any other 'near-orthogonal' projection necessarily maps $\mathbf{x}$ further away.

With convexity established we now have a relible framework to implement iterative solutions to the PCA objective.

2.2 Streaming PCA

In situations where it is infeasible or otherwise undesirable to either compute a sample covariance matrix or perform an eigendecomposition/SVD, it becomes productive to consider streaming alternatives to the classical PCA estimator. Indeed, given n samples drawn from a D-dimensional distribution, computing the sample covariance matrix has time complexity $\mathcal{O}(nD \cdot \min(n, D))$ and memory complexity $\mathcal{O}(D^2)$, while the eigendecomposition has worst-case time complexity $\mathcal{O}(D^3)$ (Elgamal and Hefeeda, 2015; Garber and Hazan, 2015; Goes et al., 2014; Kreßner, 2004). Even in the case of modest dimensionality it is not impossible to imagine environments in which the user wishes to compute estimates of the PCA estimator using a stream of sequentially received observations, rather than to be forced to wait for a sufficient number of observations to arrive.

Despite its conceptual simplicity, stochastic optimization has been shown to achieve the same error rates as empirical risk minimization (Bottou and Bousquet, 2008; Bottou, 2010; Nemirovski et al., 2009; Shalev-Shwartz and Srebro, 2008; Shalev-Shwartz et al., 2011; Shalev-Shwartz and Tewari, 2011; Shalev-Shwartz and Ben-David, 2014) and the success of these algorithms in a number of machine learning environments suggests that PCA may be afforded the same advantages if formulated properly. This framework for PCA, advocated for in Arora et al. (2012, 2013), models the problem by using sequential samples drawn from an unknown distribution $\mathcal{D}$ to iteratively update estimates of the optimal components $\mathbf{V}$.

2.2.1 Power Iterations

To formulate PCA as a streaming algorithm consider a serialized stream of D-dimensional observations drawn from some unknown distribution $\mathbf{x} \overset{\text{i.i.d}}{\sim} \mathcal{D}$. In this setting we express the problem of finding the first d principal components as the stochastic optimization problem of finding the orthogonal subspace $\mathbf{V} \in \mathbb{R}^{D \times d}$ maximizing the total variation of the projections $\mathbf{z} = \mathbf{V}^T\mathbf{x}$ over the distribution of $\mathbf{x}$, i.e.

$$\underset{\mathbf{V} \,:\, \mathbf{V}^T\mathbf{V}=\mathbb{I}_d}{\text{maximize}} \; \mathbb{E}_{\mathbf{x}}\left[\text{trace}(\text{Var}(\mathbf{z}))\right] = \underset{\mathbf{V} \,:\, \mathbf{V}^T\mathbf{V}=\mathbb{I}_d}{\text{maximize}} \; \mathbb{E}_{\mathbf{x}}\left[\text{trace}(\mathbf{V}^T\mathbf{x}\mathbf{x}^T\mathbf{V})\right], \tag{2.5}$$

or, the equivalent stochastic least-squares problem

$$\underset{\mathbf{V}\,:\,\mathbf{V}^T\mathbf{V}=\mathbb{I}_d}{\text{minimize}}\ \mathbb{E}_{\mathbf{x}}\left[\|\mathbf{x}-\mathbf{V}\mathbf{V}^T\mathbf{x}\|_2^2\right].$$

The gradient of the objective function with respect to $\mathbf{V}$ is

$$\nabla_{\mathbf{V}}\mathbb{E}_{\mathbf{x}}\left[\|\mathbf{x}-\mathbf{V}\mathbf{V}^T\mathbf{x}\|_2^2\right]=-\nabla_{\mathbf{V}}\mathbb{E}_{\mathbf{x}}\left[\text{trace}\left(\mathbf{x}\mathbf{x}^T\mathbf{V}\mathbf{V}^T\right)\right]=-2\mathbb{E}_{\mathbf{x}}\left[\mathbf{x}\mathbf{x}^T\mathbf{V}\right]$$

To compute an estimate of the parameterized subspace $\mathbf{V}$ we wish to apply an iterative projected gradient descent procedure using this gradient. That is, if f is a differentiable function then $-\nabla f(\mathbf{a})$ points in the direction f decreases most quickly at point $\mathbf{a}$. Using this fact we can construct a sequence of estimates $\mathbf{a}_0, \mathbf{a}_1, ...\mathbf{a}_n, ...$ according to

$$\mathbf{a}_{n+1}=\mathbf{a}_n-\alpha\nabla f(\mathbf{a}_n),\quad n=0,1,...$$

where $\alpha\in\mathbb{R}$ corresponds to a (sufficiently small) step size towards the optima in the direction of the gradient. With this in mind we can construct the gradient descent sequence for PCA given the stream of observations $\{\mathbf{x}_t\}_{t=1}^T$, $\mathbf{x}_t\sim\mathcal{D}$, known as the stochastic power method. We compute serial updates $\left\{\mathbf{V}^{(t)}\right\}_{t=0}^T$ according to

$$\begin{aligned}\tilde{\mathbf{V}}^{(t+1)}&=\mathbf{V}^{(t)}-\alpha\nabla_{\mathbf{V}}\left(\|\mathbf{x}_t-\mathbf{x}_t\mathbf{x}_t^T\mathbf{V}^{(t)}\|_2^2\right)\\&=\mathbf{V}^{(t)}+\alpha\mathbf{x}_t\mathbf{x}_t^T\mathbf{V}^{(t)}\\\mathbf{V}^{(t+1)}&=\mathcal{P}_{\text{orth}}\left(\tilde{\mathbf{V}}^{(t+1)}\right),\end{aligned}$$

where $\mathcal{P}_{\text{orth}}\left(\tilde{\mathbf{V}}^{(t+1)}\right)$ projects $\tilde{\mathbf{V}}^{(t)}\tilde{\mathbf{V}^{(t)}}^T$ onto the set of $D\times D$ with d eigenvalues equal to one and $D-d$ eigenvalues equal to zero. This update scheme is known as the *power iteration method* or sometimes simply stochastic gradient descent for PCA. As noted in Arora et al. (2012), the use of $\mathcal{P}_{\text{orth}}(\tilde{\mathbf{V}}^{(t)})$ is an abuse of notation since we are projecting the product $\tilde{\mathbf{V}}^{(t)}\tilde{\mathbf{V}^{(t)}}^T$ and not $\tilde{\mathbf{V}}^{(t)}$. We examine this projection operator $\mathcal{P}_{\text{orth}}$ in more detail in Section 2.3.

Noting that $\mathbf{x}_t\mathbf{x}_t^T$ has rank-1 at each step, calculating the gradient $\mathbf{x}_t\mathbf{x}_t^T\mathbf{V}^{(t-1)}$ requires $\mathcal{O}(Dd)$ operations. This is a significant savings over the complexity $\mathcal{O}(D^2d)$ of $\mathbf{\Sigma}_{\mathbf{xx}}\mathbf{V}^{(t)}$ if D is large. The projection step can be carried out in $\mathcal{O}(Dd^2)$ operations, which in principle would inflate the complexity of the whole algorithm, but (Arora et al., 2012) demonstrate that this projection step need only be performed very infrequently. This preserves the comparatively efficient time complexity of power iterations to $\mathcal{O}(Dd)$ per update. Recalling that simply calculating the sample covariance matrix $\widehat{\mathbf{\Sigma}}_{\mathbf{xx}}=\frac{1}{n}\mathbf{X}^T\mathbf{X}$ over n samples takes $\mathcal{O}(nD\cdot\min(n,D))$

operations with $\mathcal{O}(D^2)$ memory units. Contrast this with the respective time/memory cost of power iterations after n iterations, $\mathcal{O}(nDd)$ and $\mathcal{O}(Dd)$. The advantage of iterative procedures becomes clear for large ambient dimension D and small subspace dimension d.

When retrieving the first principal component ($d = 1$) this update scheme is also commonly referred to as Oja's algorithm (Oja, 1982; Oja E., 1985), or the Hebbian algorithm (Sanger, 1989) extended to PCA by Kim et al. (2005). In the case of $d = 1$, if the covariance matrix of the observations has bounded spectral norm and eigengap λ between its first and second eigenvalues, then the above projected gradient descent scheme can be shown to be ε-optimal after $\mathcal{O}\left(\frac{\log(1/\varepsilon)}{\lambda}\right)$ iterations (Shamir, 2015). Recently, Xu and Gao (2018) have given a gap-free convergence rate for the first-principal-component-retrieval problem to be as low as $\mathcal{O}(\frac{1}{\varepsilon})$. A survey of convergence rates for stochastic PCA algorithms in the both the case for $d = 1$ and $d \geq 1$ can be found in Shamir (2016) and Allen-Zhu and Li (2017).

2.2.2 Matrix Stochastic Gradient

A second iterative algorithm presented in Arora et al. (2013) seeks to optimize over the convexified problem

$$\underset{\mathbf{W} \in \mathcal{F}_d}{\text{maximize}}\ \mathbb{E}\left[\mathbf{x}^T \mathbf{W} \mathbf{x}\right] \iff \underset{\mathbf{W} \in \mathcal{F}_d}{\text{maximize}}\ \text{trace}\left(\mathbf{\Sigma_{xx}} \mathbf{W}\right). \tag{2.6}$$

The gradient of this objective is

$$\nabla_{\mathbf{W}}\ \text{trace}\left(\mathbf{\Sigma_{xx}} \mathbf{W}\right) = \mathbf{\Sigma_{xx}} \equiv \mathbf{x}\mathbf{x}^T,$$

yielding the update rule on iterates $\mathbf{W}^{(0)}, \mathbf{W}^{(1)}, ...$

$$\mathbf{W}^{(t)} = \mathcal{P}_{\mathcal{F}_d}\left(\mathbf{W}^{(t-1)} + \alpha \mathbf{x}_t \mathbf{x}_t^T\right), \tag{2.7}$$

where $\mathcal{P}_{\mathcal{F}_d}$ is now projecting the candidate updates onto the (now convex) constraint set with respect to the Frobenius inner product. Let $\bar{\mathbf{W}}$ denote the average over T iterates

$$\bar{\mathbf{W}} = \frac{1}{T} \sum_{i=1}^{T} \mathbf{W}^{(t)}.$$

Arora et al. (2013) note that since this sum of rank-d matrices is not in general rank-d some auxiliary transformation must be performed on $\bar{\mathbf{W}}$. Arora et al. (2013) address this by drawing upon related work. In particular, if a (suboptimal) feasible solution to convex problem (2.6) is not rank-d matrix then Warmuth and Kuzmin (2008) describe a randomized cap-and-scale algorithm which samples from the suboptimal $\bar{\mathbf{W}}$

to generate a rank-d matrix $\mathbf{W}^*$ whose objective value in (2.6) remains unchanged.

Algorithm 2: Matrix Stochastic Gradient Algorithm

Inputs: Step size α, iteration limit T, initialization $\mathbf{W}^{(0)}$

1 Compute (2.7) T times, each with an independent sample $\mathbf{x}_t$.

2 Average iterates $\{\mathbf{W}^{(t)}\}_{t=1}^{T}$,

$$\bar{\mathbf{W}} = \frac{1}{T}\sum_{i=1}^{T}\mathbf{W}^{(t)}.$$

3 Sample from $\bar{\mathbf{W}}$ to produce rank-d matrix $\mathbf{W}^*$ according to Warmuth and Kuzmin (2008).

Letting $\mathbf{W}_*$ denote the optimal solution to the original nonconvex problem, the MSG algorithm has convergence guarantee (Arora et al., 2013; Mianjy and Arora, 2018; Shamir and Zhang, 2013)

$$\mathbb{E}[\mathbf{x}^T\mathbf{W}_*\mathbf{x}] - \mathbb{E}[\mathbf{x}^T\mathbf{W}^*\mathbf{x}] = \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right).$$

A major limitation of the MSG algorithm is that the rank of $\mathbf{W}^{(t)}$ may increase at any iteration, causing added complexity, while the power iteration algorithm need only track a $(D \times d)$ orthogonal matrix.

In general, the cost of using convex relaxation methods is added complexity, making these implementations more difficult to scale (Mianjy and Arora, 2018). Of course, in cases where the nonconvex algorithm is essentially intractable, having a tractable inefficient problem is much appreciated.

2.2.3 Incremental PCA

A second algorithm presented by Arora et al. (2012) is based on an incremental SVD algorithm (Brand, 2002) and aims to approximate the underlying covariance matrix while accessing only a single sample at each step. The proposed update rule is

$$\mathbf{C}^{(t+1)} = \mathcal{P}_{\text{rank}-d}\left(\mathbf{C}^{(t)} + \mathbf{x}_t\mathbf{x}_t^T\right), \tag{2.8}$$

where $\mathcal{P}_{\text{rank}-d}$ projects it argument to the nearest rank-d matrix $(D \times D)$ matrix with respect to the spectral norm. This projection step is quite simple is achieved by setting the smallest $D - d$ eigenvalues to zero. In principle this would require the incremental PCA algorithm to compute an eigendecomposition at each step, but the limitation of using only single observation per step allows for an efficient algorithm since each $\mathbf{x}_t\mathbf{x}_t^T$ will be symmetric and have rank 1. After n updates the incremental PCA algorithm is shown to have time complexity $\mathcal{O}(nDd^2)$ with memory cost $\mathcal{O}(Dd)$ per. The computational cost of incremental PCA is a factor of d greater than power iterations, representing a material disadvange of the method. However,

the (deliberate) omission of a step size in (2.8) allows incremental PCA to usable out-of-the-box, given a desired subspace dimension d, while power iterations may require step-size tuning to produce satisfactory results.

2.2.4 Online PCA

Another iterative procedure to approach the PCA estimator is referred to as *online PCA* (Arora et al., 2012; Goes et al., 2014; Warmuth and Kuzmin, 2008). The derivation begins by noting that instead of solving the variance maximization problem over orthogonal projections $\mathbf{W}$ (or equivalently, over orthogonal subspaces $\mathbf{V}$) we may solve the variance *minimization* problem for the orthogonal complement of $\mathbf{M} \in \mathbb{R}^{D\times(D-d)}$,

$$\mathbf{M} = \mathbb{I}_D - \mathbf{V}\mathbf{V}^T.$$

Similar to the case of $\mathbf{V}\mathbf{V}^T$, the matrix $\mathbf{M}$ must be rank $D-d$ and have precisely $D-d$ eigenvalues equal to one with the remaining d equal to zero. Once again, this constraint (specifically the 'projection matrix' and 'rank-$(D-d)$' elements) are not convex but can be relaxed to form the convex problem

$$\min_{\mathbf{M}} \mathbb{E}\left[\text{trace}\left(\mathbf{M}\mathbf{x}\mathbf{x}^T\right)\right] \quad \text{s.t.} \quad 0 \preceq \mathbf{M},\ \|\mathbf{M}\|_2 \leq (D-d)^{-1},\ \text{trace}(\mathbf{M}) = 1. \tag{2.9}$$

Note that Warmuth and Kuzmin (2008) scale $\mathbf{M}$ so that its trace is equal to one, with the expected effect on the spectral norm. Estimates $\left\{\mathbf{M}^{(t)}\right\}_{t=1,2,\ldots}$ were shown (Arora et al., 2013) to be the iterates the mirror descent algorithm (Beck and Teboulle, 2003) according to

$$\mathbf{M}^{(t)} = \mathcal{P}_{\text{RE}}\left(\exp\left\{\log \mathbf{M}^{(t-1)} - \alpha\mathbf{x}\mathbf{x}^T\right\}\right),$$

where log and exp refer to matrix logarithms and exponentials

$$\log \mathbf{A} = \sum_{k=1}^{\infty}(-1)^{k+1}\frac{(\mathbf{A}-\mathbb{I})^k}{k},$$
$$\exp \mathbf{A} = \sum_{k=1}^{\infty}\frac{\mathbf{A}^k}{k!},$$

and $\mathcal{P}_{\text{RE}}$ projects its argument onto the nearest member of the constraint set in (2.9) with respect to the quantum relative entropy (see (Arora et al., 2013; Goes et al., 2014) for details).

Fortunately, a key result from in the formulation of online PCA is that this projection step is not too elaborate. To perform $\mathcal{P}_{\text{RE}}(\cdot)$ we cap the largest $D-d'$ eigenvalues to $1/(D-d)$ and shrink the smallest

d' eigenvalues of $\mathbf{M}$ so that trace($\mathbf{M}$) = 1 (ensuring each eigenvalue is within $[0, (D-d')^{-1}]$). Here, d' is chosen to be the largest integer that permits us to perform this cap-and-shrink while satisfying the eigenvalue constraints.

Warmuth and Kuzmin (2008) provide convergence guarantees for their method. For step size α, optimum $\mathbf{M}^*$, and assuming $\|\mathbf{x}_t\|_2 \leq 1$, they find that after T updates, such that T is bound below by

$$T \geq \mathcal{O}\left(\frac{[(D-d)\text{trace}\left(\mathbf{M}^*\mathbf{\Sigma_{xx}}\right)+\varepsilon]\, d \log \frac{d}{D}}{\varepsilon^2}\right),$$

then $\mathbf{M}^{(t)}$ will satify

$$\mathbb{E}\left[\frac{1}{t}\sum_{t=1}^{T}\text{trace}\left((D-d)\,\mathbf{M}^{(t)}\mathbf{\Sigma_{xx}}\right)\right] - \text{trace}\left((D-d)\,\mathbf{M}^*\mathbf{\Sigma_{xx}}\right) \leq \varepsilon.$$

Although such bounds exist it should be noted that the rupdates require an eigendecomposition for the the matrix logarithm and another for the projection. Optimizations to these procedures have been identified (Tsuda et al., 2005) but these do not address the separate fact that $\mathbf{M}^{(t)} \in \mathbb{R}^{D\times D}$ will be rank D at each step while updates $\mathbf{V}^{(t)}$ in the power method were only rank d, which by the nature of PCA is expected to be substantially smaller in any situation which the size of D is an obstacle.

2.3 The Orthogonal Procrustes Problem

Many formulations of a number of PCA algorithms require the estimates of $\mathbf{V}$ to be orthogonal. Hence, it is in our interest to find a projection operator $\mathcal{P}_{\text{orth}} : \mathbb{R}^{D\times d} \to \mathbb{R}^{D\times d}$ mapping an input matrix to the nearest orthogonal matrix with respect to the Frobenius inner product. Having such an operator would permit us to complete the projected gradient descent algorithm by computing iterative orthonormal updates to $\mathbf{V}$. The task of finding such an operator part of a problem more broadly known as the *orthogonal Procrustes problem* (Schönemann, 1966).

The orthogonal Procrustes problem seeks the orthogonal matrix $\mathbf{\Omega}^*$ which most closely projects a matrix $\mathbf{A}$ to a second matrix $\mathbf{B}$ according to

$$\mathbf{\Omega} = \underset{\mathbf{\Omega}}{\arg\min}\ \|\mathbf{\Omega A} - \mathbf{B}\|_F \quad \text{s.t.} \quad \mathbf{\Omega}^T\mathbf{\Omega} = \mathbb{I}.$$

Of present interest is the special case when $\mathbf{A} = \mathbb{I}$. In this setting we find that the optimizer $\mathbf{\Omega}^*$ becomes

the desired projection operator

$$\mathcal{P}_{\text{orth}}(\mathbf{B}) \equiv \mathbf{\Omega}^* = \underset{\mathbf{\Omega}}{\arg\min} \, \|\mathbf{\Omega} - \mathbf{B}\|_F \quad \text{s.t.} \quad \mathbf{\Omega}^T\mathbf{\Omega} = \mathbb{I}.$$

In the general setting with arbitrary matrix $\mathbf{A}$ the problem of finding this optimizer $\mathbf{\Omega}^*$ is equivalent to finding the nearest orthogonal matrix to a given matrix $\mathbf{M} = \mathbf{B}\mathbf{A}^T$. To find this projected matrix $\mathbf{\Omega}^*$ we use the singular value decomposition of $\mathbf{M}$

$$\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T,$$

then the optimizer $\mathcal{P}_{\text{orth}}(\mathbf{B}) \equiv \mathbf{\Omega}^*$ is given by the product

$$\mathcal{P}_{\text{orth}}(\mathbf{B}) \equiv \mathbf{\Omega}^* = \mathbf{U}\mathbf{V}^T.$$

To prove this we apply the definition Frobenius inner product $\|\mathbf{X}\|_F^2 = \text{trace}\left(\mathbf{X}^T\mathbf{X}\right)$ and some elementary properties of the trace operator,

$$\begin{aligned}
\mathbf{\Omega}^* &= \underset{\mathbf{\Omega}}{\arg\min} \, \|\mathbf{\Omega}\mathbf{A} - \mathbf{B}\|_F^2 \\
&= \underset{\mathbf{\Omega}}{\arg\min} \, \text{trace}\left(\mathbf{A}^T\mathbf{\Omega}^T\mathbf{\Omega}\mathbf{A} - 2\mathbf{A}^T\mathbf{\Omega}^T\mathbf{B} + \mathbf{B}\mathbf{B}\right) \\
&= \underset{\mathbf{\Omega}}{\arg\min} \, \left\{\text{trace}\left(\mathbf{A}^T\mathbf{A}\right) - 2\text{trace}\left(\mathbf{A}^T\mathbf{\Omega}^T\mathbf{B}\right) + \text{trace}\left(\mathbf{B}^T\mathbf{B}\right)\right\} \quad \text{(linearity of trace)} \\
&= \underset{\mathbf{\Omega}}{\arg\min} \, \left\{-\text{trace}\left(\mathbf{A}^T\mathbf{\Omega}^T\mathbf{B}\right)\right\} \\
&= \underset{\mathbf{\Omega}}{\arg\max} \, \text{trace}\left(\mathbf{A}^T\mathbf{\Omega}^T\mathbf{B}\right) \\
&= \underset{\mathbf{\Omega}}{\arg\max} \, \text{trace}\left(\mathbf{\Omega}^T\mathbf{B}\mathbf{A}^T\right). \quad \text{(cyclic invariance of trace)}
\end{aligned}$$

Applying the singular value decomposition of $\mathbf{B}\mathbf{A}^T = \mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$,

$$\begin{aligned}
\mathbf{\Omega}^* &= \arg\max_{\mathbf{\Omega}} \text{ trace } \left(\mathbf{\Omega}^T\mathbf{B}\mathbf{A}^T\right) \\
&= \arg\max_{\mathbf{\Omega}} \text{ trace } \left(\mathbf{\Omega}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\right) \\
&= \arg\max_{\mathbf{\Omega}} \text{ trace } \left(\mathbf{V}^T\mathbf{\Omega}^T\mathbf{U}\mathbf{\Sigma}\right) \quad \text{(cyclic invariance)} \\
&= \arg\max_{\mathbf{\Omega}} \text{ trace } \left(\left(\mathbf{U}^T\mathbf{\Omega}\mathbf{V}\right)^T\mathbf{\Sigma}\right) \\
&= \arg\max_{\mathbf{\Omega}} \langle \mathbf{U}^T\mathbf{\Omega}\mathbf{V}, \mathbf{\Sigma}\rangle_F.
\end{aligned}$$

Since all three matrices $\mathbf{U}, \mathbf{\Omega}, \mathbf{V}$ are orthonormal it follows that the product $\mathbf{U}^T\mathbf{\Omega}\mathbf{V}$ is itself orthonormal

$$\left(\mathbf{U}^T\mathbf{\Omega}\mathbf{V}\right)^T\mathbf{U}^T\mathbf{\Omega}\mathbf{V} = \mathbf{V}^T\mathbf{\Omega}^T\mathbf{U}\mathbf{U}^T\mathbf{\Omega}\mathbf{V} = \mathbf{V}^T\mathbf{\Omega}^T\mathbf{\Omega}\mathbf{V} = \mathbf{V}^T\mathbf{V} = \mathbb{I}.$$

Let $\mathbf{Z} = \mathbf{U}^T\mathbf{\Omega}\mathbf{V}$ and note that because $\mathbf{\Sigma}$ is a diagonal matrix we can write the product $\mathbf{Z}^T\mathbf{\Sigma}$ as

$$\mathbf{Z}^T\mathbf{\Sigma} = [\sigma_1\mathbf{z}_1 \;\; \sigma_2\mathbf{z}_2 \;\; \cdots],$$

where $\sigma_1, \sigma_2, ...$ are the singular values of $\mathbf{B}\mathbf{A}^T$ and $\mathbf{z}_1, \mathbf{z}_2, ...$ are the column vectors composing $\mathbf{Z}$. A consequence of the orthonormality of $\mathbf{Z} = [\mathbf{z}_1 \; \mathbf{z}_2 \; \cdots]$ is that the trace

$$\text{trace } \left(\mathbf{Z}^T\mathbf{\Sigma}\right) = \sum_i [\mathbf{Z}^T\mathbf{\Sigma}]_{ii} = \sum_i \sigma_i z_{ii}$$

must be maximized when each unit vector $\mathbf{z}_i = (z_{i1}, z_{i2}, ...)$ has precisely one nonzero element at the i^{th} entry

$$z_{ij} = \begin{cases} 1 & i = j \\ & \\ 0 & i \neq j \end{cases},$$

i.e. when $\mathbf{Z} = \mathbb{I}$. Any other construction of an orthonormal matrix $\mathbf{Z}$ would necessarily lead to a smaller trace. To see why, first note that the entries of $\mathbf{Z}$ must be bound within -1 and 1 as a consequence of orthonormality,

$$1 = {\mathbf{z}_i}^T\mathbf{z}_i = \sum_k {z_{ik}}^2 \geq {z_{ij}}^2 \geq 0 \implies |z_{ij}| \leq 1.$$

Using this property, along with the fact that $\sigma_i \geq 0$, we find that the products along the diagonal $\sigma_i z_{ii}$ are each maximized when $z_{ii} = 1$, and so the sum along the diagonal must itself be maximized for these

values of z_{ii}. The only way to achieve this and retain orthogonality is if all other entries off the diagonal are zero, i.e. $\mathbf{Z} = [z_{ij}] = \mathbb{I}$.

Finally, with this solution for the optimal $\mathbf{Z}^* = \mathbb{I}$, we find the expression for the optimal $\mathbf{\Omega}^*$ to be

$$\mathbb{I} = \mathbf{U}^T \mathbf{\Omega}^* \mathbf{V} \iff \mathbf{\Omega}^* = \mathbf{U}\mathbf{V}^T,$$

giving us a general expression for the orthogonal projection operator

$$\mathcal{P}_{\text{orth}}(\mathbf{B}) \equiv \mathbf{\Omega}^* = \mathbf{U}\mathbf{V}^T,$$

as desired.

Chapter 3 Robust Principal Component Analysis

A well documented property of PCA is its high degree of sensitivity to outliers (Candès et al., 2011; Chandrasekaran et al., 2011; Clarkson and Woodruff, 2017; Frieze et al., 2004; Bhojanapalli et al., 2015; Yi et al., 2016; Chen and Wainwright, 2015; Gu et al., 2016; Netrapalli et al., 2014), with even a single adversarial outlier potentially producing large deviations in the estimated subspace/principal components (Lerman and Maunu, 2018b; Maunu and Lerman, 2019). Such behaviour motivates a need for *robust* PCA estimators. In this section we will examine a number of existing robust solutions in both the online and offline settings, and then present a novel online extension of the robust algorithm derived by Zhang and Yang (2018).

It should be noted that what the literature refers to as "robust PCA" concerns two distinct (but related) problems (Lerman and Maunu, 2018b; Vaswani et al., 2018),

1. retrieving a low-rank subspace in the presence of corrupted high-dimensional data, and
2. finding a low-rank approximation to corrupted data.

The first problem has been associated with robustness to inlier-outlier/Haystack data regimes (Lerman et al., 2015; Lerman and Maunu, 2018b; Maunu and Lerman, 2019). Such models assume that the data $\mathbf{X} \in \mathbb{R}^{n\times D}$ can be partitioned into inlier and outlier components $\mathbf{X}_{\text{in}} \in \mathbb{R}^{n_{\text{in}}\times D}$ and $\mathbf{X}_{\text{out}} \in \mathbb{R}^{n_{\text{out}}\times D}$, $n_{\text{out}} = n - n_{\text{in}}$. The inliers span on or near a low-dimensional subspace while the outliers are dispersed across the ambient high-dimensional space. In the online setting we imagine that an iterate $\mathbf{x}$ is sampled from the inlier distribution with probability p_{in} or the outlier distribution with probability $p_{\text{out}} = 1 - p_{\text{in}}$. On the other hand, second problem has been more associated with sparse-corruption matrix retrieval models. Such models typically assume the data $\mathbf{X}$ has an additive decomposition of an underlying low-rank matrix $\mathbf{L}$ and a sparse corruption matrix $\mathbf{S}$ with elementwise noise,

$$\mathbf{X} = \mathbf{L} + \mathbf{S}.$$

In order to reliably distinguish between the two classes of methods we will follow the naming convention in Lerman and Maunu (2018b) wherein the first problem is referred to as *robust subspace recovery* (RSR)

while the second is referred to as *robust PCA* (RPCA).

Note that these two problems are equivalent in the classical (non-robust) PCA setting. Finding the low-rank subspace $\mathbf{V}$ immediately provides the projection generating low-rank approximations $\mathbf{V}\mathbf{V}^T\mathbf{x}$, and it is only in introducing different corruption regimes that we may distinguish between the two.

3.1 Robust Offline Methods

3.1.1 Fast Median Subspace

Many robust perspectives make use of the relationship between a projection $\mathbf{W}$ onto subspace $\mathbf{V}$ and the orthogonal complement $\mathbf{W}_\perp = \mathbb{I} - \mathbf{W}$ onto $\mathbf{V}_\perp$. Namely,

$$\begin{aligned}
\|\mathbf{W}\mathbf{x}\|_2^2 + \|\mathbf{W}_\perp\mathbf{x}\|_2^2 &= \mathbf{x}^T\mathbf{W}^T\mathbf{W}\mathbf{x} + \mathbf{x}^T\mathbf{W}_\perp^T\mathbf{W}_\perp\mathbf{x} \\
&= \mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{x}^T(\mathbb{I} - \mathbf{W})\mathbf{x} \\
&= \mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{W}\mathbf{x} \\
&= \mathbf{x}^T\mathbf{x} \\
&= \|\mathbf{x}\|_2^2,
\end{aligned}$$

so

$$\|\mathbf{x}\|_2^2 - \|\mathbf{W}_\perp\mathbf{x}\|_2^2 = \|\mathbf{W}\mathbf{x}\|_2^2. \tag{3.1}$$

This allows us to express the PCA objective in terms of an optimization problem over orthogonal complements according to

$$\begin{aligned}
\underset{\mathbf{W}}{\text{minimize}} \sum_{i=1}^n \|\mathbf{x}_i - \mathbf{W}\mathbf{x}_i\|_2^2 &\iff \underset{\mathbf{W}}{\text{maximize}} \sum_{i=1}^n \mathbf{x}_i^T\mathbf{W}\mathbf{x}_i \\
&\iff \underset{\mathbf{W}}{\text{maximize}} \sum_{i=1}^n \mathbf{x}_i^T\mathbf{W}^T\mathbf{W}\mathbf{x}_i \quad \text{(idempotence of } \mathbf{W}\text{)} \\
&\iff \underset{\mathbf{W}}{\text{maximize}} \sum_{i=1}^n \|\mathbf{W}\mathbf{x}_i\|_2^2 \\
&\iff \underset{\mathbf{W}_\perp}{\text{minimize}} \sum_{i=1}^n \|\mathbf{W}_\perp\mathbf{x}_i\|_2^2
\end{aligned}$$

A popular way of inducing robustness in the subspace recovery setting is to manipulate the PCA objective in this form by instead using the least absolute deviations over the set of orthogonal complements $\mathbf{W}_\perp$ instead

of the squared ℓ_2 norm (Lerman and Maunu, 2018b)

$$\underset{\mathbf{W}_\perp}{\text{minimize}} \sum_{i=1}^{n} \|\mathbf{W}_\perp \mathbf{x}_i\|_2. \tag{3.2}$$

It should be noted that the identity (3.1) does not apply to $\|\mathbf{x}\|_2$, and so solutions to (3.2) will not in general coincide with solutions to

$$\underset{\mathbf{W}}{\text{maximize}} \sum_{i=1}^{n} \|\mathbf{W}\mathbf{x}_i\|_2. \tag{3.3}$$

Nevertheless, algorithms for (3.2) and (3.3) can be adapted to suit each other, making both worthwhile to consider.

Lerman and Maunu (2018a) present an iterative algorithm, referred to as the *fast median subspace* (FMS) algorithm, which approximates the solution $\mathbf{V}^*$ of (3.2) by iterates

$$\mathbf{V}^{(t+1)} = \arg\min \sum_{i=1}^{n} w_i^{(t)} \|\mathbf{W}_\perp \mathbf{x}_i\|_2,$$

where weights $\{w_i^{(t)}\}_{i=1}^n$ are given by

$$w_i^{(t)} = \|\mathbf{W}_\perp^{(t)}\|_2^{-1},$$

with $\mathbf{W}_\perp^{(t)}$ the orthogonal complement to the orthogonal projection onto $\mathbf{V}^{(t)}$. The solution to the FMS problem is shown to be given by the solution to the weighted PCA problem by performing an eigendecomposition on the weighted covariance matrix $\mathbf{X}^T \text{diag}(w_1^{(t)}, ..., w_n^{(t)})\mathbf{X}$ on the centered data.

3.1.2 Geometric Median Subspace

The *geometric median subspace* (GMS) algorithm (Zhang and Lerman, 2012) is another robust estimator using objective (3.2) but manipulates the constraint set so that the problem is convexified. The GMS relaxes the constraints on candidates to include all symmetric matrices with trace-1. Since aspirant matrices in the GMS algorithm are no longer the orthogonal complements to orthogonal projections $\mathbf{W}$ onto a candidate subspaces, we instead denote these approximations of $\mathbf{W}_\perp$ by $\mathbf{Q}$. The GMS-optimal $\mathbf{Q}^*$ is the solution to

$$\mathbf{Q}^* = \arg\min_{\mathbf{Q}} \sum_{i=1}^{n} \|\mathbf{Q}\mathbf{x}_i\|_2 \quad \text{s.t.} \quad \mathbf{Q}^T = \mathbf{Q},\ \text{trace}(\mathbf{Q}) = 1. \tag{3.4}$$

The target subspace is then estimated using the last d eigenvectors (those with the smallest d eigenvalues) of the optimal $\mathbf{Q}^*$. Furthermore, the matrix $\mathbf{Q}^*$ is shown to be a robust approximation to the inverse covariance matrix of $\mathbf{X}$ in the context of $\mathbf{X} = \mathbf{L} + \mathbf{S}$ noise structure for magnitudes of noise not too large.

3.1.3 REAPER

The REAPER algorithm (so called because it *harvests* low-rank structure from data) was formulated as an improvement to GMS (Lerman et al., 2015) by defining a tighter relaxation of the original nonconvex problem. To do so they consider the Fantope of order $D - d$, shown earlier to be the tighest relaxation of rank-$(D - d)$ projections. The REAPER-optimal $\mathbf{Q}^*$ is the solution

$$\mathbf{Q}^* = \underset{\mathbf{Q} \in \mathcal{F}_{D-d}}{\arg\min} \sum_{i=1}^{n} \|\mathbf{Q}\mathbf{x}_i\|_2 \tag{3.5}$$

As is the case with the GMS algorithm, the robust estimate of $\mathbf{V}$ is provided by the eigenvectors of $\mathbf{Q}^*$ corresponding to the smallest d eigenvalues.

In both GMS and REAPER procedures the authors recommend an iteratively reweighted least squares (IRLS) strategy for obtaining the minimizer $\mathbf{Q}^*$. The GMS-optimal $\mathbf{Q}^*$ is solved by iteration according to

$$\mathbf{Q}^{(t+1)} = \left(\sum_{i=1}^{n} \frac{\mathbf{x}_i \mathbf{x}_i^T}{\max\left(\delta, \|\mathbf{Q}^{(t)}\mathbf{x}_i\|_2\right)} \right)^{-1} / \text{trace} \left(\left(\sum_{i=1}^{n} \frac{\mathbf{x}_i \mathbf{x}_i^T}{\max\left(\delta, \|\mathbf{Q}^{(t)}\mathbf{x}_i\|_2\right)} \right)^{-1} \right),$$

where δ is some (typically very small) regularizer to avoid dividing by zero. The IRLS strategy for the REAPER algorithm is more subtle and requires establishing an approximate equivalence of its objective with a weighted quadratic program.

3.2 Robust Streaming Methods

3.2.1 Robust Stochastic Gradient Descent

A weakness of the GMS and REAPER algorithms is that they are both derived in the context of full access to a sample $\mathbf{X}$ and so are unsuited to large data regimes defined as is. Goes et al. (2014) offers two stochastic alternatives that modify the GMS objective known as *robust stochastic gradient descent* (robust SGD).

The first of the robust SGD methods considers the problem

$$\underset{\mathbf{U} \in \mathbb{R}^{D \times d}}{\text{maximize}} \; \mathbb{E}\left[\|\mathbf{U}^T\mathbf{x}\|_2\right] \quad \text{s.t.} \quad \|\mathbf{U}\|_2 \leq 1. \tag{3.6}$$

When $d = 1$ (3.6) simplifies to projection-pursuit PCA which has been shown to be robust by earlier

work (Li and Chen, 1985). To define a descent algorithm from (3.6) first compute its gradient

$$\begin{aligned}\nabla_{\mathbf{U}} \|\mathbf{U}^T\mathbf{x}\|_2 &\equiv \nabla_{\mathbf{U}} \sqrt{\|\mathbf{U}^T\mathbf{x}\|_2^2} \\ &= \frac{\nabla_{\mathbf{U}} \|\mathbf{U}^T\mathbf{x}\|_2^2}{\sqrt{\|\mathbf{U}^T\mathbf{x}\|_2^2}} \\ &= \frac{\mathbf{x}\mathbf{x}^T\mathbf{U}}{\|\mathbf{U}^T\mathbf{x}\|_2}.\end{aligned}$$

Goes et al. (2014) then solve (3.6) by considering iterates of the form

$$\mathbf{U}^{(t+1)} = \mathcal{P}_{\text{orth}} \left(\mathbf{U}^{(t)} + \eta_t \frac{\mathbf{x}_t \mathbf{x}_t^T \mathbf{U}^{(t)}}{\|{\mathbf{U}^{(t)}}^T \mathbf{x}_t\|_2} \right). \tag{3.7}$$

By the same argument as was seen in the power iteration algorithm via Arora et al. (2012), Goes et al. (2014) conclude that the projection step may be omitted from the majority of updates. After n samples the first robust SGD algorithm is reported to have equal costs $\mathcal{O}(nDd)$ in time and $\mathcal{O}(Dd)$ in space as in power iterations.

In principle any power $\|\mathbf{U}^T\mathbf{x}\|_2^p$ could be taken for $0 < p < 2$, with increasing robustness anticipated as $p \to 0$. Of course, $p = 1$ corresponds to the least p such that (3.6) remains convex. Should one be brave enough to attempt the nonconvex problem of $p \in (0, 1)$, we modify the update rule to be

$$\mathbf{U}^{(t+1)} = \mathcal{P}_{\text{orth}} \left(\mathbf{U}^{(t)} + \eta_t \mathbf{x}_t \mathbf{x}_t^T \frac{\mathbf{x}_t \mathbf{x}_t^T \mathbf{U}^{(t)}}{\|{\mathbf{U}^{(t)}}^T \mathbf{x}_t\|_2^{p-2}} \right).$$

The second robust SGD method considers a similar problem as (3.6) but instead seeks to minimize over orthogonal complements,

$$\underset{\mathbf{U} \in \mathbb{R}^{D \times (D-d)}}{\text{minimize}} \; \mathbb{E}\left[\|\mathbf{U}^T\mathbf{x}\|_2\right] \quad \text{s.t.} \quad \mathbf{U}^T\mathbf{U} = \mathbb{I}_{D-d}. \tag{3.8}$$

The update rule for (3.8) is found in the same way as for (3.6), defining iterates

$$\mathbf{U}^{(t+1)} = \mathcal{P}_{\text{orth}} \left(\mathbf{U}^{(t)} - \eta_t \frac{\mathbf{x}_t \mathbf{x}_t^T \mathbf{U}^{(t)}}{\|{\mathbf{U}^{(t)}}^T \mathbf{x}_t\|_2} \right). \tag{3.9}$$

3.2.2 Robust Incremenetal SGD

Goes et al. (2014) derive a third algorithm based upon the GMS problem, distinct from the 'robust SGD' methods above. In terms of a stochastic optimization problem, the GMS procedure is expressible as

$$\mathbf{Q}^* = \underset{\mathbf{Q} \in \mathbb{R}^{D \times D}}{\arg\min} \; \mathbb{E}\left[\|\mathbf{Q}\mathbf{x}\|_2\right] \quad \text{s.t.} \quad \mathbf{Q}^T = \mathbf{Q}, \; \text{trace}(\mathbf{Q}) = 1, \tag{3.10}$$

where $\mathbf{Q}$ can be thought of as 'near'-orthogonal complements to orthogonal projections $\mathbf{W}$ onto a low-dimensional subspace, and $\mathbf{Q}^{*-1} = \widehat{\mathbf{\Sigma}}$ the robust covariance estimator. Differentiating the objective in the same manner as for the robust SGD problems yields

$$\nabla_{\mathbf{Q}} \mathbb{E}\left[\|\mathbf{Q}\mathbf{x}\|_2\right] = \mathbb{E}\left[\frac{\mathbf{x}\mathbf{x}^T\mathbf{Q}}{\|\mathbf{Q}\mathbf{x}\|_2}\right] = \mathbf{Q}\ \mathbb{E}\left[\frac{\mathbf{x}\mathbf{x}^T}{\|\mathbf{Q}\mathbf{x}\|_2}\right].$$

So, at $\mathbf{Q} = \mathbf{Q}^*$ Goes et al. (2014) note that

$$\begin{aligned} \mathbf{Q}^* &= c \cdot \mathbb{E}\left[\frac{\mathbf{x}\mathbf{x}^T}{\|\mathbf{Q}^*\mathbf{x}\|_2}\right]^{-1} \\ \iff \widehat{\mathbf{\Sigma}} &= \frac{1}{c} \cdot \mathbb{E}\left[\frac{\mathbf{x}\mathbf{x}^T}{\|\widehat{\mathbf{\Sigma}}^{-1}\mathbf{x}\|_2}\right]. \end{aligned}$$

A naive proposal for an update procedure for robust covariance iterates $\mathbf{\Sigma}^{(t)}$ is given by

$$\mathbf{\Sigma}^{(t+1)} = \mathcal{P}_{\text{rank}-d}\left(\mathbf{\Sigma}^{(t)} + \frac{\mathbf{x}_t\mathbf{x}_t^T}{\|(\mathbf{\Sigma}^{(t)})^{-1}\|_2}\right). \tag{3.11}$$

However, this it is immediately obvious that this is ill-defined since the projection step will necessarily make the iterates singular. The address this deficiency the authors suggest first considering the update on the inverse matrix

$$(\mathbf{\Sigma}^{(t+1)})^{-1} = (\mathbf{\Sigma}^{(t)})^{-1} - \frac{(\mathbf{\Sigma}^{(t)})^{-1}\mathbf{x}\mathbf{x}^T(\mathbf{\Sigma}^{(t)})^{-1}}{1 + \mathbf{x}^T(\mathbf{\Sigma}^{(t)})^{-1}\mathbf{x}}, \tag{3.12}$$

then following up this step with the rank-d projection. This adjustment is a direct application of the Sherman-Morrison formula for an invertible matrix $\mathbf{S}$ and outer product $\mathbf{u}\mathbf{v}^T$

$$(\mathbf{S} + \mathbf{u}\mathbf{v})^{-1} = \mathbf{S}^{-1} - \frac{\mathbf{S}^{-1}\mathbf{u}\mathbf{v}^T\mathbf{S}^{-1}}{1 + \mathbf{v}^T\mathbf{S}^{-1}\mathbf{u}}.$$

The inverse adjustment procedure is reported to be computed in $4D + 5Dd$ operations. Therefore, since the rest of the rank-1 update is identical to the incremental PCA algorithm in Arora et al. (2012), we find that the robust incremental procedure is able benefit from robustness while preserving time cost $\mathcal{O}(nDd^2)$ and space cost $\mathcal{O}(Dd)$.

3.2.3 Robust Online PCA

The final algorithm presented by Goes et al. (2014) extends the REAPER procedure to a stochastic setting,

$$\underset{\mathbf{Q}\in\mathcal{F}_{D-d}}{\text{minimize}}\ \mathbb{E}\left[\|\mathbf{Q}\mathbf{x}\|_2\right] \tag{3.13}$$

Scaling $\mathbf{Q}$ so that it has trace-1 yields formulation

$$\underset{\mathbf{Q}}{\text{minimize}}\ \mathbb{E}\left[\|\mathbf{Q}\mathbf{x}\|_2\right] \quad \text{s.t.} \quad 0 \preceq \mathbf{Q},\ \|\mathbf{Q}\|_2 \leq \frac{1}{D-d},\ \text{trace}(\mathbf{Q}) = 1. \tag{3.14}$$

In this form we are able to draw an immediate comparison with the online PCA of Warmuth and Kuzmin (2008), now with 'robustified' objective $\mathbb{E}\left[\|\mathbf{Q}\mathbf{x}\|_2\right]$ in lieu of the classical (online) PCA objective $\mathbb{E}\left[\|\mathbf{Q}\mathbf{x}\|_2^2\right]$. Using the mirror descent solution from the original online PCA problem to the robust version (3.14) defines the sequence of updates

$$\mathbf{Q}^{(t+1)} = \mathcal{P}_{\text{RE}}\left(\exp\left\{\log\mathbf{Q}^{(t)} - \eta_t\frac{\mathbf{x}_t\mathbf{x}_t^T\mathbf{Q}^{(t)} + \mathbf{Q}^{(t)}\mathbf{x}_t\mathbf{x}_t^T}{2\|\mathbf{Q}^{(t)}\mathbf{x}_t\|_2}\right\}\right). \tag{3.15}$$

A principal result from Goes et al. (2014) is that after T iterations with step size $\eta_t = \eta = \frac{2}{\sqrt{T}}\sqrt{\frac{d}{D-d}}$ the updates (3.15) achieve convergence guarantee

$$(D-d)\mathbb{E}\left[\|\widehat{\mathbf{Q}}\mathbf{x}\|_2\right] \leq (D-d)\inf_{\mathbf{Q}}\mathbb{E}\left[\|\mathbf{Q}\mathbf{x}\|_2\right] + \sqrt{\frac{d(D-d)}{T}},$$

where $\widehat{\mathbf{Q}}$ is sampled uniformly from the iterates $\mathbf{Q}^{(1)}, ..., \mathbf{Q}^{(T)}$.

3.3 Robustness via Percentile Hard-Thresholding

3.3.1 Offline Setting

One method to achieve robustness that has seen success in low-rank recovery is to apply hard-thresholding to the PCA objective (Zhang and Yang, 2018). Let $F_\gamma : \mathbb{R}^{n\times D} \to \mathbb{R}^{n\times D}$ be a hard-thresholding function mapping entries of input $\mathbf{A} = [\mathbf{A}_{ij}]_{i,j}$ to zero if their magnitudes simultaneously exceed the $(1-\gamma)^{\text{th}}$ percentile of both its row and column values (in absolute value), i.e.,

$$\left[F_\gamma\left(\mathbf{A}\right)\right]_{ij} = \begin{cases} 0 & \text{if } |\mathbf{A}_{ij}| > |\mathbf{A}_{i,\cdot}|^{[\gamma]} \text{ and } |\mathbf{A}_{ij}| > |\mathbf{A}_{\cdot,j}|^{[\gamma]} \\ \mathbf{A}_{ij} & \text{otherwise,} \end{cases} \tag{3.16}$$

where $\mathbf{A}_{i,\cdot}$ represents the i^{th} row of $\mathbf{A}$ and $\mathbf{A}_{\cdot,j}$ represents the j^{th} column of $\mathbf{A}$, while $|\mathbf{A}_{i,\cdot}|^{[\gamma]}$ and $|\mathbf{A}_{\cdot,j}|^{[\gamma]}$ represent the $(1-\gamma)^{\text{th}}$ percentile of the absolute values of the i^{th} row and j^{th} column, respectively. With this hard-thresholding function the robust PCA problem as described in Zhang and Yang (2018) is formulated as follows: Given data $\mathbf{X} = \mathbf{L}^* + \mathbf{S}^*$, for low-rank $\mathbf{L}^*$ and sparse corruption matrix $\mathbf{S}^*$, the sample robust PCA problem seeks to recover $\mathbf{L}^*$ (and thereby recovering $\mathbf{S}^*$ as well) from $\mathbf{X}$ by solving the optimization problem

$$\widehat{\mathbf{L}} = \underset{\mathbf{L} \,:\, \text{rank}(\mathbf{L})=d}{\arg\min} \|F_\gamma(\mathbf{L} - \mathbf{X})\|_F^2. \tag{3.17}$$

An iterative projected gradient descent solution using manifold optimization is provided by Zhang and Yang (2018), with memory usage $\mathcal{O}(nD)$ and a per-iteration complexity of $\mathcal{O}(nDd)$. The high relatively complexity emerges from the requirement of computing a full singular value decomposition in each step of the descent procedure. The cost of this algorithm motivates a need for streaming alternatives that have less costly update rules. Such streaming algorithms are also expected to be less memory-intensive since they will be unable to access more than a single observation per iteration.

3.3.2 Extension to the Streaming Setting

We now turn towards the novel contribution of this thesis by extending the robust PCA algorithm described in Zhang and Yang (2018) to a streaming setting. We seek to design an objective analogous to that of objective (3.17) but expressed as a stochastic optimization problem, as was the case for the the streaming formulation of classical PCA. To do so, as was the case for classical PCA, we treat the data as a stream of D-dimensional observations drawn from some unknown distribution $\mathbf{x} \overset{\text{i.i.d}}{\sim} \mathcal{D}$.

Consider the vector-valued analogue of thresholding function (3.16) $F_\gamma : \mathbb{R}^D \to \mathbb{R}^D$ mapping the entries of a vector to zero if the magnitudes exceeds the $(1-\gamma)^{\text{th}}$ percentile of the input entries (in absolute value), i.e.

$$\left(F_\gamma(\mathbf{a})\right)_j = \begin{cases} 0 & \text{if } |a_j| > |\mathbf{a}|^{[\gamma]} \\ a_j & \text{otherwise,} \end{cases} \tag{3.18}$$

where $|\mathbf{a}|^{[\gamma]}$ denotes the $(1-\gamma)^{\text{th}}$ percentile of the absolute values of $\mathbf{a} = (a_1, ..., a_D) \in \mathbb{R}^D$. Our optimization procedure is as follows: We simultaneously seek the d-dimensional orthonormal subspace parameterized by $\mathbf{V} \in \mathbb{R}^{D\times d}$ and d-dimensional transformation $\mathbf{z}$ solving the least-squares problem

$$\underset{\mathbf{z}\in\mathbb{R}^d,\, \mathbf{V}\in\mathbb{R}^{D\times d}}{\text{minimize}} \mathbb{E}_{\mathbf{x}} \left[\|F_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z})\|_2^2\right] \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V} = \mathbb{I}_d. \tag{3.19}$$

To explore this objective function define the operator $S_\gamma : \mathbb{R}^D \to \mathbb{R}^D$ serving as an indicator of whether

an entry of its input exceeds the thresholding percentile of its determined by its entries

$$\left(S_\gamma(\mathbf{a})\right)_j = \begin{cases} 0 & \text{if } |a_j| > |\mathbf{a}|^{[\gamma]} \\ 1 & \text{otherwise.} \end{cases}$$

Using S_γ rewrite the hard-thresholding operator as

$$F_\gamma(\mathbf{a}) = S_\gamma(\mathbf{a}) \circ \mathbf{a},$$

where $\circ$ represents the elementwise product of the arguments. Two key properties of S_γ are that it is idempotent with respect to $\circ$

$$S_\gamma(\mathbf{a}) \circ S_\gamma(\mathbf{a}) = S_\gamma(\mathbf{a}),$$

and that if the magnitudes of the entries of $\mathbf{a}$ are not all identical different then, for some small perturbation $\Delta\mathbf{a}$,

$$S_\gamma(\mathbf{a}) = S(\mathbf{a} + \Delta\mathbf{a}).$$

That is, the sparsity pattern of $\mathbf{a}$ remains unchanged under small perturbations. This decomposition of F_γ in terms of S_γ, and the following properties of S_γ, will prove to be convenient when deriving the gradient of our robust objective.

3.3.2.1 Alternating Minimization

The objective (3.19) requires joint optimization of both the parameterized subspace $\mathbf{V}$ and the composite variables $\mathbf{z}$. To solve this we perform an alternating minimization procedure by separating the joint problem into two marginal optimization problems. First, keep $\mathbf{V}$ fixed at some estimate $\mathbf{V}_0$ and optimize (3.19) with respect to $\mathbf{z}$,

$$\mathbf{z}^* = \underset{\mathbf{z}\in\mathbb{R}^d}{\arg\min} \; \|F_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z})\|_2^2. \tag{3.20}$$

Next, keep $\mathbf{z}$ fixed at the updated value $\mathbf{z} = \mathbf{z}^*$ and now minimize (3.19) with respect to $\mathbf{V}$

$$\mathbf{V}^* = \underset{\mathbf{V}\in\mathbb{R}^{D\times d}}{\arg\min} \; \|F_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}^*)\|_2^2 \quad \text{s.t.} \quad \mathbf{V}^T\mathbf{V} = \mathbb{I}_d. \tag{3.21}$$

Procedures (3.20) and (3.21) are then iterated through until some tolerance has been reached. To solve

(3.20) consider a finite difference using a small perturbation of our variables $\mathbf{z} + \Delta\mathbf{z}$. We have

$$\begin{aligned}
&\|F_\gamma(\mathbf{x} - \mathbf{V}_0(\mathbf{z} + \Delta\mathbf{z}))\|_2^2 - \|F_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z})\|_2^2 \\
&\qquad = \|S_\gamma(\mathbf{x} - \mathbf{V}_0(\mathbf{z} + \Delta\mathbf{z})) \circ (\mathbf{x} - \mathbf{V}_0(\mathbf{z} + \Delta\mathbf{z}))\|_2^2 - \|S_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}_0\mathbf{z})\|_2^2 \\
&\qquad = \|S_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}_0\mathbf{z} - \mathbf{V}_0\Delta\mathbf{z})\|_2^2 - \|S_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}_0\mathbf{z})\|_2^2 \\
&\qquad = 2\langle S_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}_0\mathbf{z}), -\mathbf{V}_0\Delta\mathbf{z}\rangle \\
&\qquad = 2\langle F_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z}), -\mathbf{V}_0\Delta\mathbf{z}\rangle.
\end{aligned}$$

Hence, the gradient of $\mathbb{E}_{\mathbf{x}}\left[\|F_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z})\|_2^2\right]$ with respect to $\mathbf{z}$ is given by the inner product $\langle F_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z}), -\mathbf{V}_0\Delta\mathbf{z}\rangle$ (up to a constant), yielding a gradient descent step towards the minimizer $\mathbf{z}^*$,

$$\mathbf{z}^+ \leftarrow \mathbf{z} + \alpha F_\gamma(\mathbf{x} - \mathbf{V}_0\mathbf{z})\mathbf{V}_0.$$

Using similar arguments in search of incremental updates $\mathbf{V} + \Delta\mathbf{V}$ towards the minimzer $\mathbf{V}^*$,

$$\begin{aligned}
&\|F_\gamma(\mathbf{x} - (\mathbf{V} + \Delta\mathbf{V})(\mathbf{z} + \Delta\mathbf{z}))\|_2^2 \\
&\qquad = \|S_\gamma(\mathbf{x} - (\mathbf{V} + \Delta\mathbf{V})(\mathbf{z} + \Delta\mathbf{z})) \circ (\mathbf{x} - (\mathbf{V} + \Delta\mathbf{V})(\mathbf{z} + \Delta\mathbf{z}))\|_2^2 \\
&\qquad = \|S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ (\mathbf{x} - (\mathbf{V} + \Delta\mathbf{V})(\mathbf{z} + \Delta\mathbf{z}))\|_2^2 \\
&\qquad = \|S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ ((\mathbf{x} - \mathbf{V}\mathbf{z}) - (\mathbf{V}\Delta\mathbf{z} + \Delta\mathbf{V}\mathbf{z}))\|_2^2 \\
&\qquad = \langle S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}\mathbf{z}), \mathbf{x} - \mathbf{V}\mathbf{z}\rangle - \langle S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}\mathbf{z}), \mathbf{V}\Delta\mathbf{z} + \Delta\mathbf{V}\mathbf{z}\rangle.
\end{aligned}$$

Meanwhile,

$$\begin{aligned}
\|F_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z})\|_2^2 &= \|S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}\mathbf{z})\|_2^2 \\
&= \langle S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}\mathbf{z}), \mathbf{x} - \mathbf{V}\mathbf{z}\rangle.
\end{aligned}$$

Therefore, subtracting these two results, to form our finite difference

$$\begin{aligned}
&\|F_\gamma(\mathbf{x} - (\mathbf{V} + \Delta\mathbf{V})(\mathbf{z} + \Delta\mathbf{z}))\|_2^2 - \|F(\mathbf{x} - \mathbf{V}\mathbf{z})\|_2^2 \\
&\qquad = 2\langle S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}\mathbf{z}), -\mathbf{V}\Delta\mathbf{z} - \Delta\mathbf{V}\mathbf{z}\rangle \\
&\qquad = 2\langle S_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}) \circ (\mathbf{x} - \mathbf{V}\mathbf{z}), -\Delta\mathbf{V}\mathbf{z}\rangle \\
&\qquad = 2\langle F_\gamma(\mathbf{x} - \mathbf{V}\mathbf{z}), -\Delta\mathbf{V}\mathbf{z}\rangle.
\end{aligned}$$

Hence, the gradient of $\mathbb{E}_{\mathbf{x}}\left[\|F_\gamma(\mathbf{x}-\mathbf{V}_0\mathbf{z})\|_2^2\right]$ with respect to $\mathbf{V}$ is given by $-F_\gamma(\mathbf{x}-\mathbf{V}\mathbf{z})\mathbf{z}^T$, yielding a gradient descent step towards the optimal $\mathbf{V}^*$

$$\mathbf{V}^+ \leftarrow \mathcal{P}_{\text{orth}}\left(\mathbf{V}+\alpha F_\gamma(\mathbf{x}-\mathbf{V}\mathbf{z})\mathbf{z}^T\right),$$

where the $\mathcal{P}_{\text{orth}}$ is performed to ensure $\mathbf{V}^+$ is a feasible solution. Given an observation $\mathbf{x}_t$, step size α, and iterates $\mathbf{V}^{(t)}$, $\mathbf{z}^{(t)}$, we define robust streaming PCA algorithm as the iterative alternating update rules

$$\begin{aligned}\mathbf{z}^{(t+1)} &= \mathbf{z}^{(t)} + \alpha F_\gamma(\mathbf{x}_t - \mathbf{V}^{(t)}\mathbf{z}^{(t)})\mathbf{V}^{(t)}\\ \mathbf{V}^{(t+1)} &= \mathcal{P}_{\text{orth}}\left(\mathbf{V}^{(t)} + \alpha F_\gamma\left(\mathbf{x}_t - \mathbf{V}^{(t)}\mathbf{z}^{(t+1)}\right)\mathbf{z}^{(t+1)^T}\right).\end{aligned}$$

3.3.2.2 Additional Projections of V

We now explore some properties of $\mathbf{V}$ which will permit us to derive an additional projection operation. This second projection will allow us to form a second update rule for $\mathbf{V}$, and therefore a second alternating minimization procedure for our robust streaming algorithm. Applying the orthonormality of $\mathbf{V}$, and since an update $\mathbf{V}+\Delta\mathbf{V}$ must also satisfy, we have

$$(\mathbf{V}+\Delta\mathbf{V})^T(\mathbf{V}+\Delta\mathbf{V}) = \mathbb{I}_d.$$

Then,

$$\begin{aligned}\mathbb{I}_d &= (\mathbf{V}+\Delta\mathbf{V})^T(\mathbf{V}+\Delta\mathbf{V})\\ &= \mathbf{V}^T\mathbf{V} + \mathbf{V}^T\Delta\mathbf{V} + \Delta\mathbf{V}^T\mathbf{V} + \Delta\mathbf{V}^T\Delta\mathbf{V}\\ \iff \mathbf{0} &= \mathbf{V}^T\Delta\mathbf{V} + \Delta\mathbf{V}^T\mathbf{V} + \Delta\mathbf{V}^T\Delta\mathbf{V},\end{aligned}$$

with $\Delta\mathbf{V}^T\Delta\mathbf{V} \approx \mathbf{0}$ we may simplify this as

$$\mathbf{V}^T\Delta\mathbf{V} + \Delta\mathbf{V}^T\mathbf{V} = \mathbf{0}.$$

That is, $\mathbf{V}^T\Delta\mathbf{V}$ is a *skew-symmetric matrix*.[1] We are able to exploit this constraint on $\mathbf{V}^T\Delta\mathbf{V}$ to derive a projection operator on increment $\Delta\mathbf{V}$. Denote the set of increments $\Delta\mathbf{V}$ of $\mathbf{V}$ satisfying the skew-symmetry

[1] A matrix $\mathbf{A}$ is said to be *skew-symmetric* if $\mathbf{A}^T = -\mathbf{A}$.

of $\mathbf{V}^T\Delta\mathbf{V}$ by

$$\mathcal{T} = \left\{\Delta\mathbf{V} \in \mathbb{R}^{D\times d} \ : \ \mathbf{V}^T\Delta\mathbf{V} + \Delta\mathbf{V}^T\mathbf{V} = \mathbf{0}\right\}.$$

We can see that any step $\Delta\mathbf{V} \in \mathcal{T}$ should have the decomposition

$$\Delta\mathbf{V} = \mathbf{Y}_1 + \mathbf{V}\mathbf{Y}_2,$$

where $\mathbf{Y}_1 \in \mathbb{R}^{D\times d}$ satisfies $\mathbf{V}^T\mathbf{Y}_1 = \mathbf{0}$ and $\mathbf{Y}_2 \in \mathbb{R}^{d\times d}$ is some skew-symmetric matrix. With this we denote $\mathcal{P}_{\text{skewsym}}$ to be the projection operator projecting matrices onto the set $\mathcal{T}$.[2] Then, given a matrix $\mathbf{A} \in \mathbb{R}^{D\times d}$, its projection onto the set $\mathcal{T}$ (with respect to $\mathbf{V}$) is given by

$$\mathcal{P}_{\text{skewsym}}(\mathbf{A}) = \left(\mathbb{I}_D - \mathbf{V}\mathbf{V}^T\right)\mathbf{A} + \mathbf{V}\left(\frac{\mathbf{V}^T\mathbf{A} - \mathbf{A}^T\mathbf{V}}{2}\right)$$

This gives rise to two possibilities for updating estimates of $\mathbf{V}$

$$\text{(Algorithm 1)} \quad \mathbf{V}^{(t+1)} = \mathcal{P}_{\text{orth}}\left(\mathbf{V}^{(t)} + \alpha F_\gamma\left(\mathbf{x}_t - \mathbf{V}^{(t)}\mathbf{z}^{(t+1)}\right)\mathbf{z}^{(t+1)^T}\right)$$

$$\text{(Algorithm 2)} \quad \mathbf{V}^{(t+1)} = \mathcal{P}_{\text{orth}}\left(\mathbf{V}^{(t)} + \alpha\mathcal{P}_{\text{skewsym}}\left(F_\gamma\left(\mathbf{x}_t - \mathbf{V}^{(t)}\mathbf{z}^{(t+1)}\right)\mathbf{z}^{(t+1)^T}\right)\right).$$

[2]Note that this is an abuse of notation since $\mathcal{P}_{\text{skewsym}}$ takes as its argument increment $\Delta\mathbf{V}$ and transforms it so that $\mathbf{V}^T\Delta\mathbf{V}$ is skew-symmetric, not $\Delta\mathbf{V}$ itself.

Chapter 4 Experiments

4.1 Artificial Data

In this section we seek to evaluate the performance of both proposals for robust streaming extensions of PCA. Of present interest is a data regime in which observations $\mathbf{x} \in \mathbb{R}^D$ are subject to a sparse additive noise component $\mathbf{s} \in \mathbb{R}^D$. Let $q \in [0, 1]$ denote the proportion of nonzero entries of $\mathbf{s}$ to zero entries, so that $\mathbf{x}$ is exposed to corruption in a total of $q \cdot D$ entries. This outlier setting is designed as a vectorized analogy to the sparse-corruption noise regime found in the low-rank matrix decomposition literature $\mathbf{X} = \mathbf{L} + \mathbf{S}$, including Zhang and Yang (2018), which used this setting to derive and test the offline robust PCA via percentile thresholding algorithm. To this end, we generate i.i.d. samples of the form

$$\mathbf{x} = \mathbf{l} + \mathbf{s}$$

where $\mathbf{l} \in \mathbb{R}^D$ represents the underlying signal from a lower-dimensional space embedded in in $\mathbb{R}^D$ and $\mathbf{s} \in \mathbb{R}^D$ provides the additive noise components with $q \cdot D$ nonzero elements. For brevity, we will abuse terminology and refer to the signal $\mathbf{l}$ as a "low-" or dimensional" or "lower-dimensional" signal rather than "the signal drawn from a low-rank subspace of $\mathbb{R}^D$". To generate $\mathbf{l}$ we draw i.i.d. samples $\mathbf{l} \sim \mathcal{N}(0, \mathbf{\Sigma})$, where $\mathbf{\Sigma} = \mathbf{V}^* \mathbf{V}^{*T} \in \mathbb{R}^{D \times D}$ is a rank-d covariance matrix, and $\mathbf{V}^* \in \mathbb{R}^{D \times d}$ is an orthonormal parametrization of the underlying subspace. We generate the gross corruption vector $\mathbf{s}$ by drawing $q \cdot D$ entries from $\mathcal{N}(0, 100)$, with the remaining $D - q \cdot D$ entries set to zero.

We follow the testing metric used in Goes et al. (2014) for assessing the performance of robust streaming analogues to PCA. Specifically, for each update $\mathbf{V}^{(t)}$ of $\mathbf{V}^*$ we compute the normalized subspace angle between the estimated subspace and the true subspace, given by the largest inner product of the columns of $\mathbf{V}^{(t)}$ and $\mathbf{V}^*$ (Knyazev and Argentati, 2002). If the two subspaces are orthogonal to one another then the angle between them will be $\frac{\pi}{2}$, and so for convenience we choose to scale the subspace angle to range over $[0, 1]$. Additionally, the two novel robust streaming algorithms are contrasted with the classical PCA

estimates which provides a lower bound for the performance of a streaming PCA implementation.

4.1.1 Step Size Tuning

We first investigate both algorithms' performance as a function of the step size α, as well as the three-way relationship between chosen step size α, thresholding parameter γ, and noise proportion q. In the following simulations we set the dimension of the ambient space $D = 100$, the target subspace dimension $d = 4$. In order to empirically address what an appropriate step size should be, we consider both constant step sizes. Constant step sizes are chosen from $\alpha \in \{0.01, 0.025, 0.05, 0.1, 0.25, 0.5\}$ while step sizes proportional to $\frac{1}{\sqrt{t}}$ and $\frac{1}{t}$ are chosen as candidates for adaptive steps.

The influence of step size on performance can be seen in Figure 4.1, with the subspace angles calculated from Algorithm 1 drawn as solid lines, and the angles calculated from Algorithm 2's solutions drawn as dashed lines. Note that in all four simulations the noise proportion q and thresholding parameter γ are equal. Figure 4.1 makes it immediately clear that a constant step size is unsuited to both algorithms since constant steps appear to provide (comparatively) unstable estimates when approaching the minimum while being no faster than either adaptive option. Using adaptive step sizes permits the two algorithms to quickly recover reasonable estimates of the underlying subspace, with no clear performance distinction between sizes $\alpha \propto \frac{1}{t}$ or $\alpha \propto \frac{1}{\sqrt{t}}$.

A second observation from Figure 4.1 is that both Algorithm 1 and 2 seem to have some difficulty in attaining the minimum and appear to effectively plateau to suboptimal levels if the proportion q of noisy entries is too large, even after 3×10^4 iterations.

It is also worthwhile to investigate the case when $\gamma \neq q$ as it is unrealistic to assume *apriori* knowledge of the data stream corruption proportion. To demonstrate the performance of both algorithms we consider both scenarios $\gamma < q$ and $\gamma > q$.

The results of such settings are visualized in Figure 4.2. Once again we find that constant step sizes prove to be comparatively mediocre at generating reasonable increments of the estimated subspace towards $\mathbf{V}^*$, with little distinction between the two algorithms. More interestingly, we find that both streaming methods trend towards the optima in the case of $\gamma < q$ (albeit slowly), but quickly stall in the case of of $\gamma > q$. One interpretation of this is that if $\gamma > q$ then both algorithms are guaranteed to threshold entries belonging to the low-dimensional signal vector $\mathbf{l}$ since the $q \cdot D$ entries of $\mathbf{x}$ associated with the nonzero elements of $\mathbf{s}$ will have likely already been set to zero (as long as the corruption magnitudes are sufficiently extreme relative to the signal magnitudes). As a consequence, both algorithms remove a $(\gamma - q) > 0$ proportion of elements that may no longer be used to draw the estimates $\mathbf{V}^{(t)}$ towards $\mathbf{V}^*$. On the other hand, if $\gamma < q$

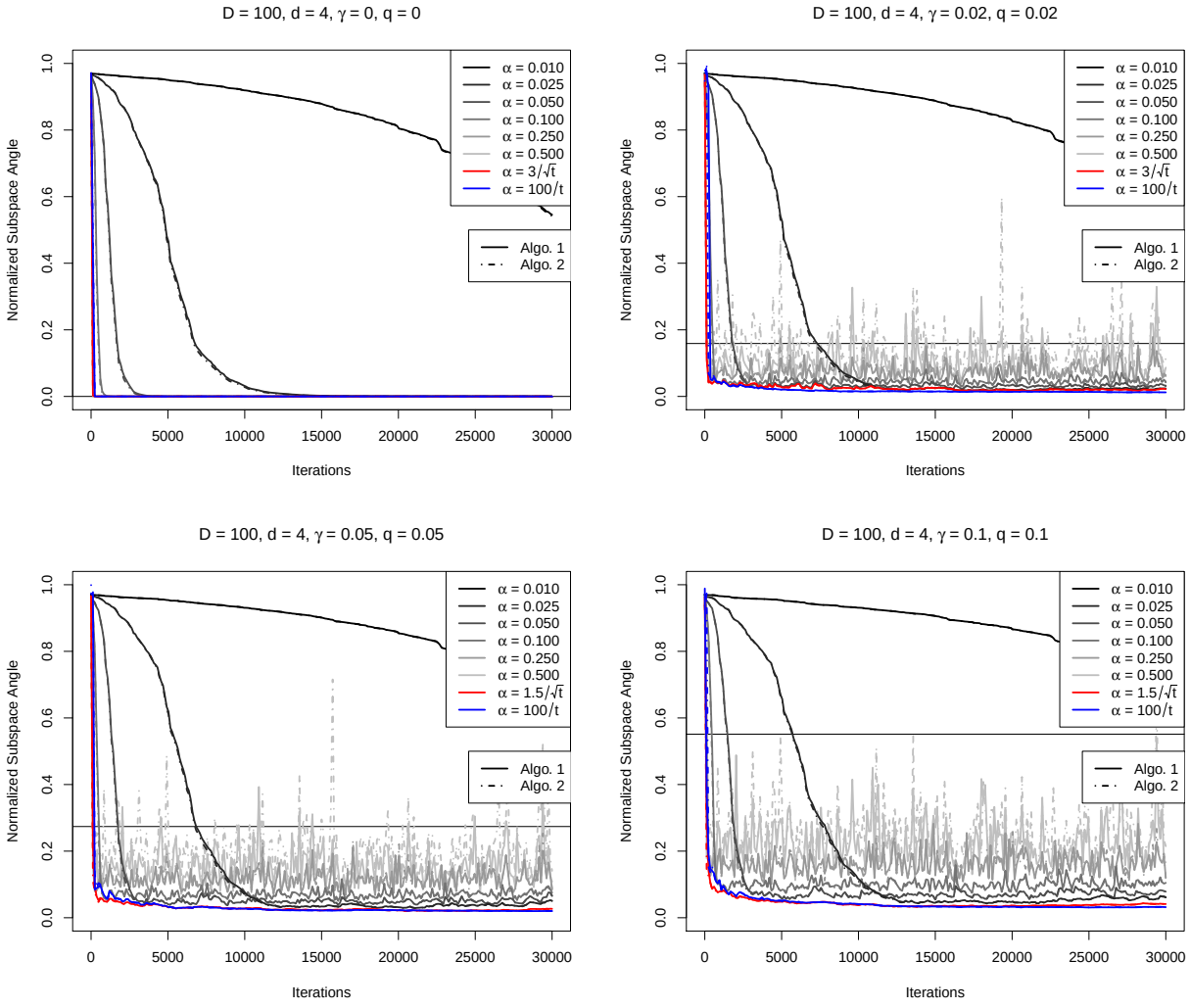

Figure 4.1: Dependence of the normalized subspace angle between the true low-rank subspace $\mathbf{V}^*$ and the estimates $\mathbf{V}^{(t)}$, $t = 1, ..., 3 \times 10^4$, when the thresholding parameter equals the noise proportion $\gamma = q$. The horizontal line is given by the classical (offline) PCA estimate of the underlying subspace which serves as a lower bound for the stochastic (classical) PCA algorithm.

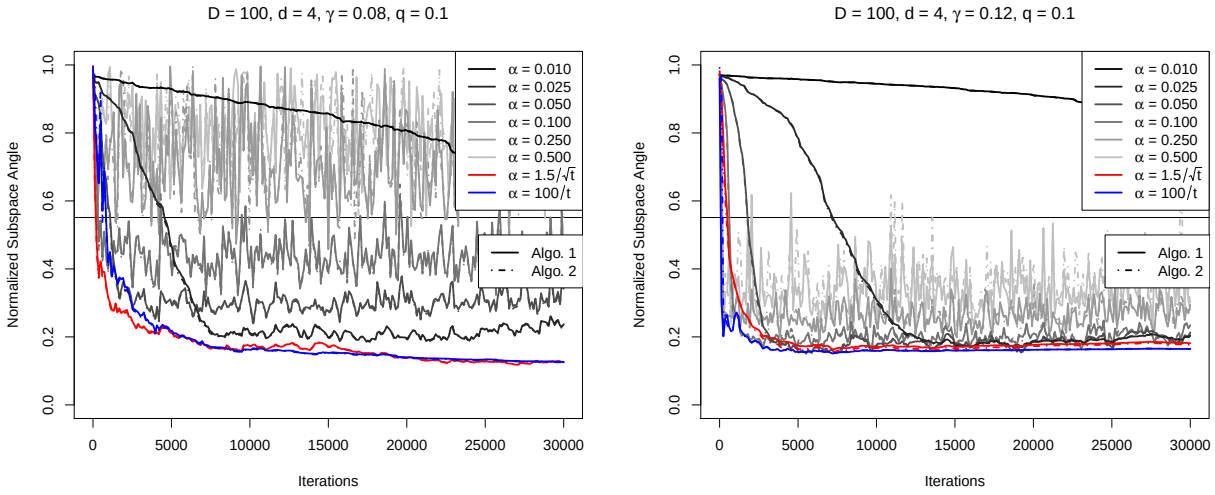

Figure 4.2: Dependence of the normalized subspace angle between the true low-rank subspace $\mathbf{V}^*$ and the estimates $\mathbf{V}^{(t)}$, $t = 1, ..., 3 \times 10^4$, when the thresholding parameter equals the noise proportion $\gamma < q$ (left) and $\gamma > q$ (right). The horizonal line is given by the classical (offline) PCA estimate of the underlying subspace which serves as a lower bound for the stochastic (classical) PCA algorithm.

then we subject the algorithm to a $(q - \gamma) > 0$ proportion of un-thresholded corrupted elements. In this case we would expect the robust streaming PCA algorithms to perform similarly to a classical streaming PCA algorithm when noise level $q' = q - \gamma$. More in-depth simulations of the (q, γ) relationship are presented in the next section.

4.1.2 Thresholding Percentile

We now wish to investigate how sensitive the estimates $\mathbf{V}^{(t)}$ are to changes in γ for some fixed corruption proportion q. Figure 4.3 shows two scenarios with fixed $q = 0.05$ and $q = 0.1$ and varying γ around these values of q. Note that Figure 4.3 is drawn using a log scale for the normalized subspace angle in order to make the separation between thresholding parameters easier to visualize.

We continue to see this tendency of $\mathbf{V}^{(t)}$ to quickly plateau when approaching $\mathbf{V}^*$ if $\gamma > q$. This effect holds even when the difference $\gamma - q = 0.01$, which corresponds to both algorithms incorrectly thresholding a single entry on each iteration under our setting $D = 100$ (for sufficiently extreme corruptions). For this reason, it appears to be worthwhile to underestimate γ when the population noise proportion is unknown.

We can more clearly visualize the (q, γ) relationship by generating a heatmap of the normalized subspace angles between $\mathbf{V}^{(T)}$ and $\mathbf{V}^*$ after $T = 3 \times 10^4$ iterations, as well as minimal angles between $\mathbf{V}^{(t)}$ and $\mathbf{V}^*$. Such a visualization of subspace angles is found in Figure 4.4, including both terminal and minimal angles, across parameter settings $\gamma \in \{0, 0.01, 0.02, ..., 0.99, 1\}$ and $q \in \{0, 0.01, 0.02, ..., 0.99, 1\}$ using step

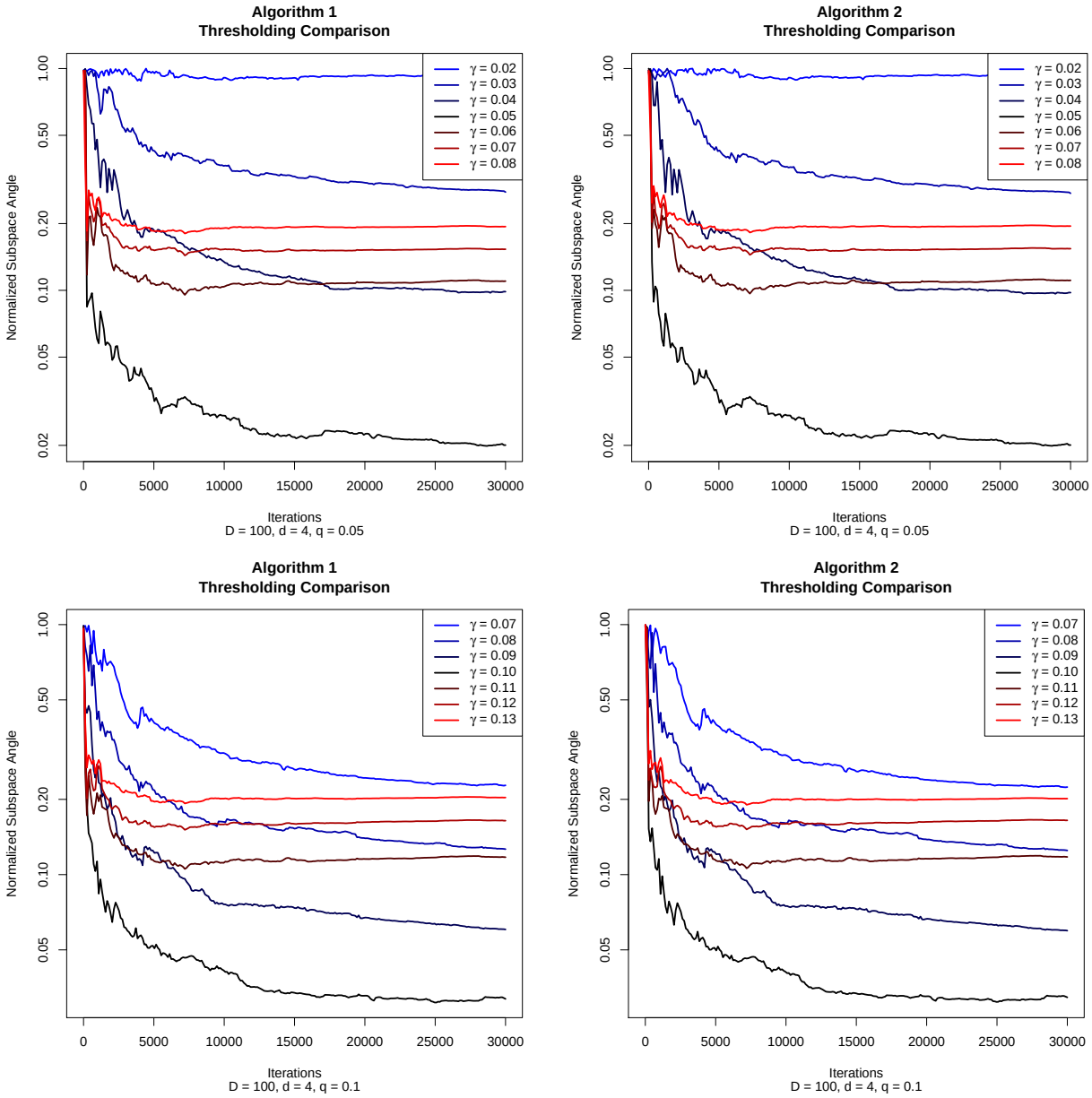

Figure 4.3: The ability of algorithm 1 (left column) and algorithm 2 (right column) to recover the target subspace fixing the noise proportion to $q = 0.05$ (top row) and $q = 0.1$ (bottom row) under a range of thresholding settings.

size $\alpha \propto \frac{1}{t}$.

Consistent with earlier figures, we find a relatively narrow band in which both Algorithm 1 and 2 are able to recover $\mathbf{V}^*$. If $\gamma < q$ then convergence is slow and only produces good estimates when γ and q are not too different. On the other hand, if $\gamma > q$ then both algorithms are able to quickly provide near-orthogonal estimates, but the quality of these estimates above the line $\gamma = q$ can be seen to plateau and have been shown to be nontrivially suboptimal from earlier figures. A feature of Figure 4.4 that is immediately noticeable is the distinction between Algorithms 1 and 2. Algorithm 1 can be seen to be more tolerant of more extreme noise proportions if q can be reliably estimated, and therefore be used to set $\gamma \approx q$. Holding γ constant at $\gamma = q$ we look restrict our investigation from Figure 4.4 to the question of how much noise either model can tolerate. This is illustrated in Figure 4.5. We see a gradual performance decay when increasing $q \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4\}$. By holding γ at q, suggested by Figure 4.4 to be the ideal thresholding parameter, we see that both algorithms seem to perform similarly, suggesting that the performance difference seems to be limited to suboptimal settings of γ.

4.2 Real Data Tests

A popular data set for testing robust matrix retrieval algorithms in the $\mathbf{X} = \mathbf{L}+\mathbf{S}$ framework is the publically available `shoppingmall` data studied in Candès et al. (2011); Yi et al. (2016); Zhang and Yang (2018). The data consists of a sequence of frames taken from a camera in a shopping mall arranged in a 1000×81920 dimensional matrix. The $n = 1000$ rows correspond to individual frames of the video feed, and the $D = 81920$ columns correspond to a vectorized arrangement of video feed's pixels, with resolution 320×256 pixels. For comparability with the offline algorithm in Zhang and Yang (2018) we set $d = 3$, as it was found that the underlying signal can be well-approximated when its subspace is assumed to be 3-dimensional. We use step size $\alpha_t = 0.01/\sqrt{t}$ for both Algorithms 1 and 2, and set $\gamma = 0.065$, hard-thresholding values in each frame if their absolute values exceed the 93.5^{th} percentile. The procedure is initialized with a random orthogonal matrix $\mathbf{V}^{(0)}$ and a random vector $\mathbf{z}^{(0)}$. Figures 4.6 and 4.7 visualize the algorithms' performance by their ability to decompose the data stream into background components (signal drawn from a low-rank space) and foreground components (sparse perturbations over the signal).

From Figures 4.6 and 4.7 we find some ability to remove foreground components from the background, particularly as the procedure iterates through later frames, but it is clear that the algorithms' are not as effective as the offline implementation in Zhang and Yang (2018). This is difference between offline and online performance is not surprising since streaming algorithms necessarily have access to considerably less information. Zhang and Yang (2018) performed 100 gradient descent steps accessing the entire 1000

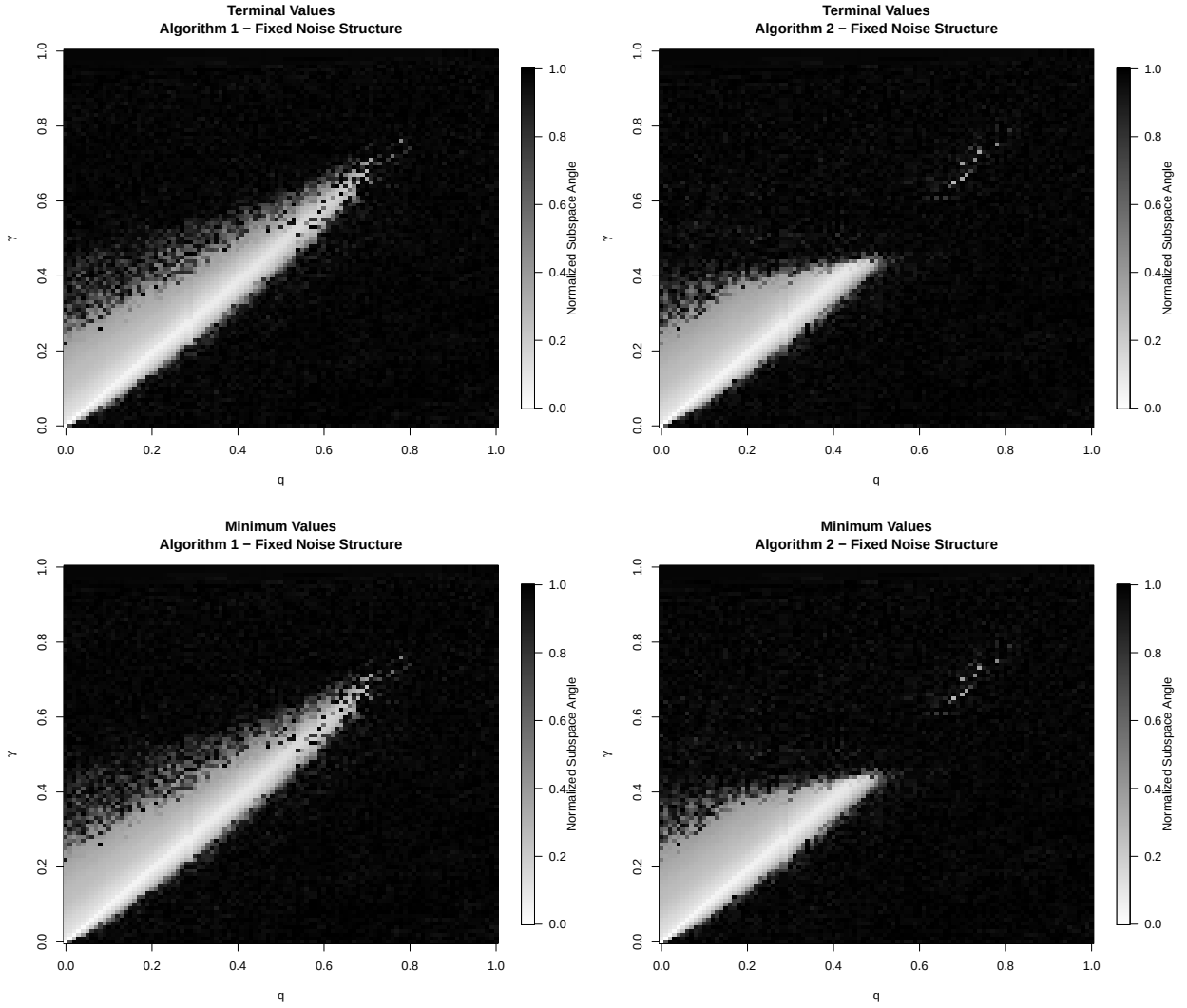

Figure 4.4: The ability of algorithm 1 (left column) and algorithm 2 (right column) to recover the target subspace across $\gamma \in \{0, 0.01, 0.02, ..., 0.99, 1\}$ and $q \in \{0, 0.01, 0.02, ..., 0.99, 1\}$ after 3×10^4 iterations. Terminal subspace angles are presented in the top row and minimal subspace angles are presented in the bottom row.

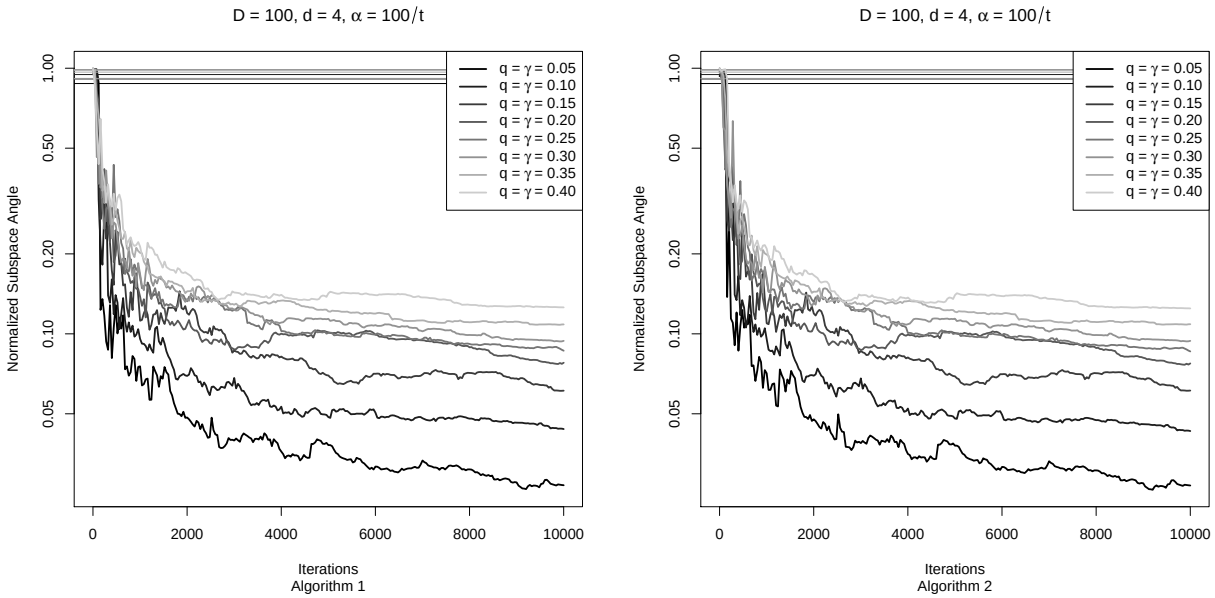

Figure 4.5: Algorithm 1 (left) and 2 (right) performance when increasing q and setting the (impressionistically) optimal $\gamma := q$. The horizontal lines correspond to the offline classical PCA estimate which provides a lower bound for classical streaming PCA estimates.

observations per update, while the present version only has access to single observations per update and only the 1000 observations available in the data.

One observation of the algorithms' performance on the `shoppingmall` data is that extreme regions of brightness/darkness found in the background appear to be having difficulty being correctly grouped with background features, and instead seem to be primarily grouped with the foreground (i.e. the bright pillar on the left, the bright counter on the right, the dark square tiles). One way to understand why this occurs in the online implementation but not its offline counterpart is via the thresholding operation F_γ. The offline algorithm thresholds a value $a_{i,j}$ to zero if its absolute value simultaneously exceeds the $(1-\gamma)^{\text{th}}$ percentile of the absolute values of row i (i.e. all other values frame i) and column j (i.e. all other values corresponding to pixel j). As a result, if a pixel beloning to the background has an extreme light/dark value then it is unlikely to be thresholded since, by virtue of belonging to the background, the pixel value will remain static across the observations. This is not the case for the streaming algorithms. The online algorithm does not have the benefit of across-observation (across-frame) information and is instead forced to make a decision on which pixels to threshold based solely upon the values within the single frame across the 81920 pixels. Therefore, if it a pixel is sufficienctly extreme relative to the other pixels in the frame, there is little to prevent the online algorithm from being forced to set it to zero.

It should be noted that there is no *ipso facto* reason that the thresholding percentile γ must remain

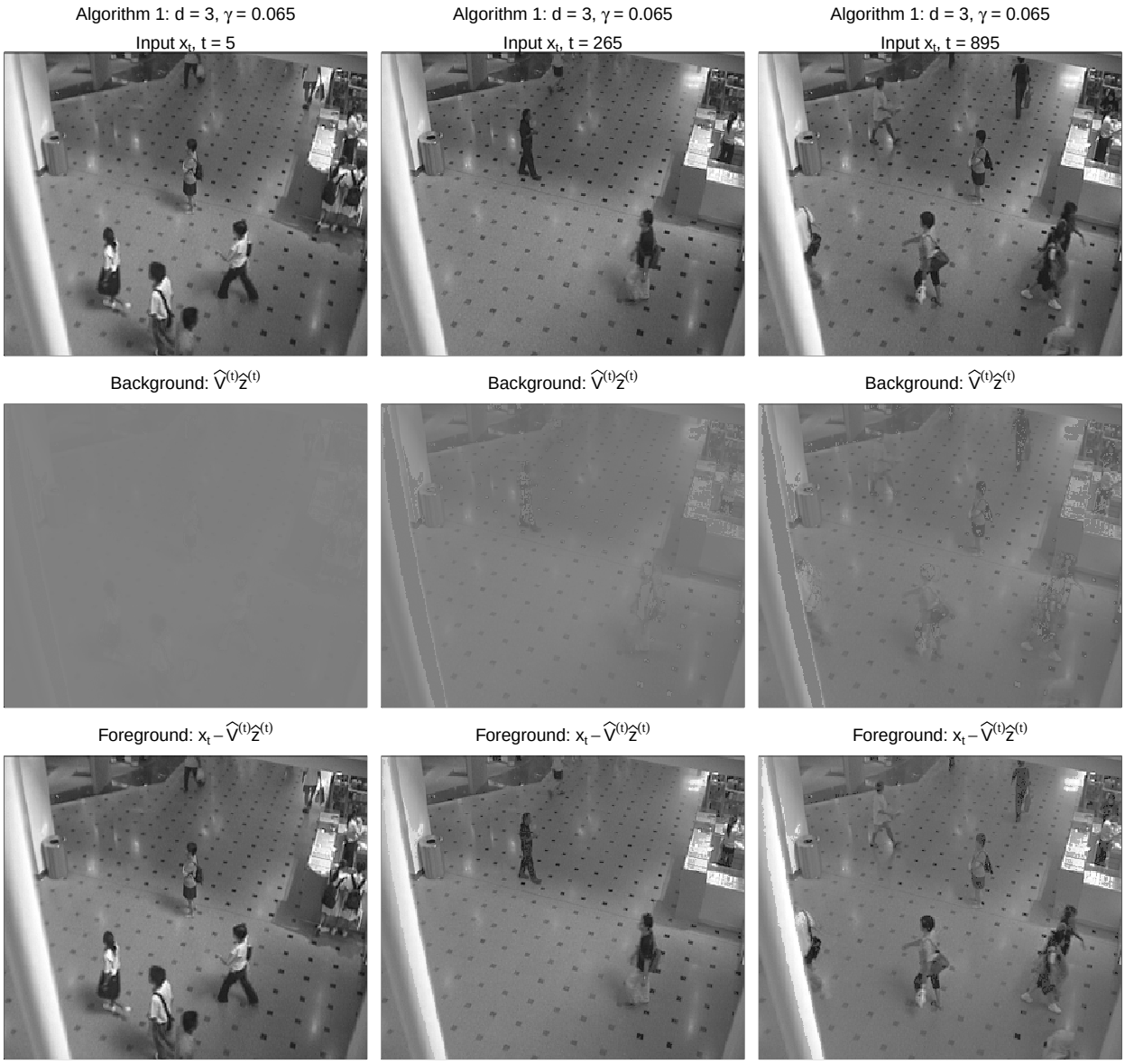

Figure 4.6: The performance of Algorithm 1 in the task of background/foreground decomposition at steps $t = 5, 265, 895$ (left to right), with low-dimension assumption $d = 3$, thresholding parameter $\gamma = 0.065$ and step size $\alpha_t = 0.01/\sqrt{t}$.

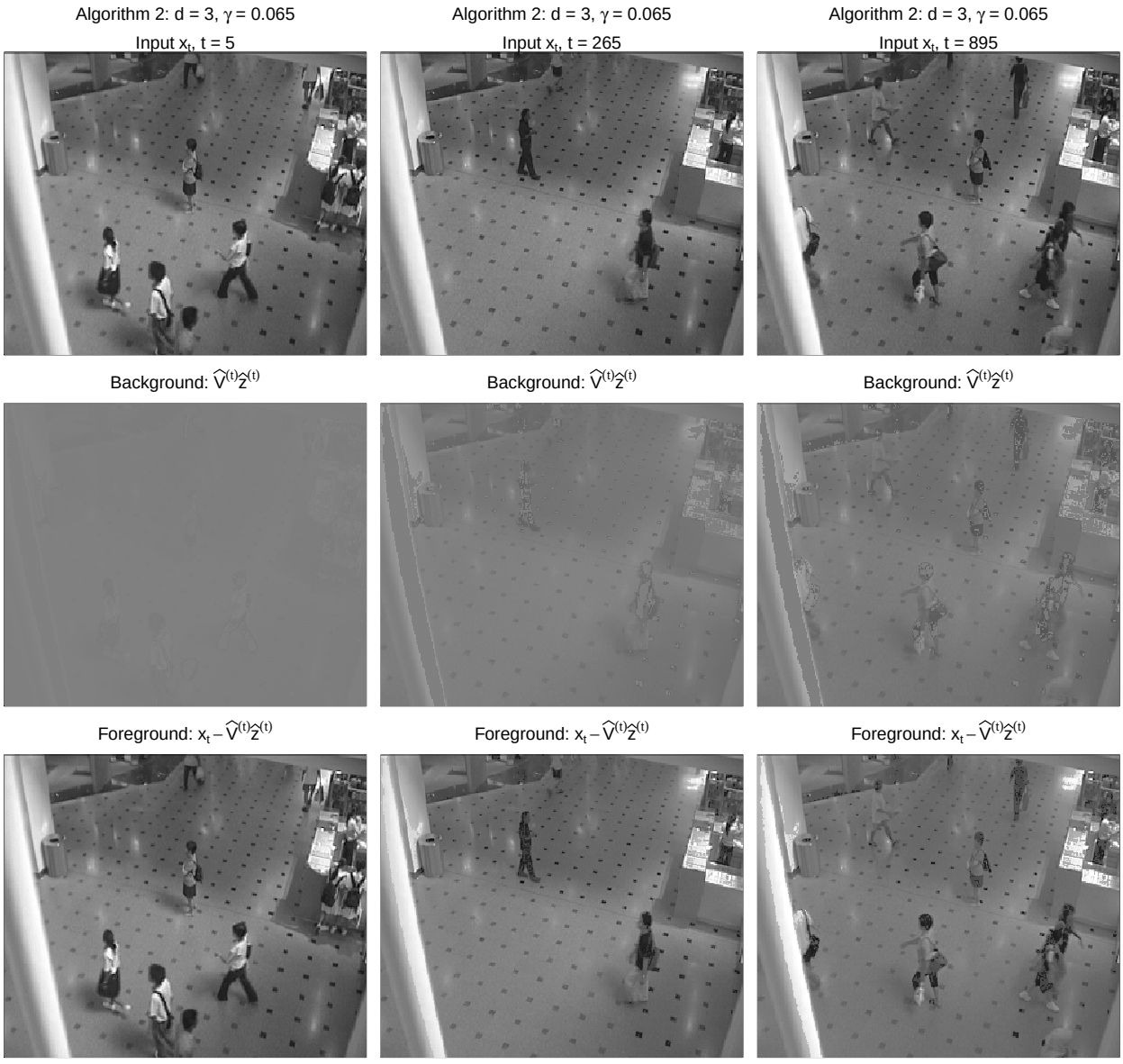

Figure 4.7: The performance of Algorithm 2 in the task of background/foreground decomposition at steps $t = 5, 265, 895$ (left to right), with low-dimension assumption $d = 3$, thresholding parameter $\gamma = 0.065$ and step size $\alpha_t = 0.01/\sqrt{t}$.

constant throughout the online procedure, and as a result it is not difficult to imagine an algorithm which takes into account frame-by-frame information to produce variable estimates for γ.

4.3 Conclusion

This thesis proposes two novel robust streaming implementations of the PCA procedure based on the offline percentile hard-thresholding algorithm in Zhang and Yang (2018). Compared to the offline algorithm with access to the full sample throughout the gradient descent procedure, the streaming alternatives present solutions that are less costly in terms of both runtime and memory requirements at the expense of accuracy. However, accuracy is a welcomed expense in situations where an offline implementation is simply infeasible. Indeed, the utility of dimensionality reduction is proportional to the dimension of the problem, and with ever increasing data volumes it has become an outstanding problem to develop streaming solutions.

References

Allen-Zhu, Z. and Li, Y. (2017) First efficient convergence for streaming k-pca: a global, gap-free, and near-optimal rate. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, 487–492. IEEE.

Arora, R., Cotter, A., Livescu, K. and Srebro, N. (2012) Stochastic optimization for pca and pls. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 861–868. IEEE.

Arora, R., Cotter, A. and Srebro, N. (2013) Stochastic optimization of pca with capped msg. In *Advances in Neural Information Processing Systems*, 1815–1823.

Arora, R. and Marinov, T. V. (2019) Efficient convex relaxations for streaming pca. In *Advances in Neural Information Processing Systems*, 10496–10505.

Beck, A. and Teboulle, M. (2003) Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, **31**, 167–175.

Bhojanapalli, S., Jain, P. and Sanghavi, S. (2015) Tighter low-rank approximation via sampling the leveraged element. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, 902–920. Society for Industrial and Applied Mathematics.

Bottou, L. (2010) Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, 177–186. Springer.

Bottou, L. and Bousquet, O. (2008) The tradeoffs of large scale learning. In *Advances in neural information processing systems*, 161–168.

Boyd, S. and Vandenberghe, L. (2004) *Convex optimization*. Cambridge university press.

Brand, M. (2002) Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision*, 707–720. Springer.

Candès, E. J., Li, X., Ma, Y. and Wright, J. (2011) Robust principal component analysis? *Journal of the ACM (JACM)*, **58**, 11.

Candès, E. J. and Tao, T. (2010) The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, **56**, 2053–2080.

Chandrasekaran, V., Sanghavi, S., Parrilo, P. A. and Willsky, A. S. (2011) Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, **21**, 572–596.

Chang, X., Nie, F., Yang, Y. and Huang, H. (2014) A convex sparse pca for feature analysis. *arXiv preprint arXiv:1411.6233*.

Chen, Y. and Wainwright, M. J. (2015) Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*.

Clarkson, K. L. and Woodruff, D. P. (2017) Low-rank approximation and regression in input sparsity time. *Journal of the ACM (JACM)*, **63**, 54.

Dattorro, J. (2010) *Convex optimization & Euclidean distance geometry*. Lulu. com.

Eckart, C. and Young, G. (1936) The approximation of one matrix by another of lower rank. *Psychometrika*, **1**, 211–218.

Elgamal, T. and Hefeeda, M. (2015) Analysis of pca algorithms in distributed environments. *arXiv preprint arXiv:1503.05214*.

Fan, K. (1949) On a theorem of weyl concerning eigenvalues of linear transformations i. *Proceedings of the National Academy of Sciences of the United States of America*, **35**, 652.

— (1950) On a theorem of weyl concerning eigenvalues of linear transformations: Ii. *Proceedings of the National Academy of Sciences of the United States of America*, **36**, 31.

Fillmore, P. and Williams, J. (1971) Some convexity theorems for matrices. *Glasgow Mathematical Journal*, **12**, 110–117.

Frieze, A., Kannan, R. and Vempala, S. (2004) Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)*, **51**, 1025–1041.

Garber, D. and Hazan, E. (2015) Fast and simple pca via convex optimization. *arXiv preprint arXiv:1509.05647*.

Goes, J., Zhang, T., Arora, R. and Lerman, G. (2014) Robust stochastic principal component analysis. *Artificial Intelligence and Statistics*, 266–274.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep learning*. MIT press.

Gu, Q., Wang, Z. W. and Liu, H. (2016) Low-rank and sparse structure pursuit via alternating minimization. In *Artificial Intelligence and Statistics*, 600–609.

Hastie, T., Tibshirani, R. and Wainwright, M. (2015) *Statistical learning with sparsity: the lasso and generalizations*. CRC press.

Hotelling, H. (1933) Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, **24**, 417.

Hu, C., Zhang, B., Yan, S., Yang, Q., Yan, J., Chen, Z. and Ma, W.-Y. (2004) Mining ratio rules via principal sparse non-negative matrix factorization. In *Fourth IEEE International Conference on Data Mining (ICDM'04)*, 407–410. IEEE.

Journée, M., Nesterov, Y., Richtárik, P. and Sepulchre, R. (2010) Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research*, **11**.

Julian, I. A. (2008) Modern multivariate statistical techniques: Regression, classification, and manifold learning.

Kim, K. I., Franz, M. O. and Scholkopf, B. (2005) Iterative kernel principal component analysis for image modeling. *IEEE transactions on pattern analysis and machine intelligence*, **27**, 1351–1366.

Knyazev, A. V. and Argentati, M. E. (2002) Principal angles between subspaces in an a-based scalar product: algorithms and perturbation estimates. *SIAM Journal on Scientific Computing*, **23**, 2009–2041.

Kotłowski, W. and Warmuth, M. K. (2015) Pca with gaussian perturbations. *arXiv preprint arXiv:1506.04855*.

Kreßner, D. (2004) Numerical methods and software for general and structured eigenvalue problems.

Lerman, G. and Maunu, T. (2018a) Fast, robust and non-convex subspace recovery. *Information and Inference: A Journal of the IMA*, **7**, 277–336.

— (2018b) An overview of robust subspace recovery. *Proceedings of the IEEE*, **106**, 1380–1410.

Lerman, G., McCoy, M. B., Tropp, J. A. and Zhang, T. (2015) Robust computation of linear models by convex relaxation. *Foundations of Computational Mathematics*, **15**, 363–410.

Li, G. and Chen, Z. (1985) Projection-pursuit approach to robust dispersion matrices and principal components: primary theory and monte carlo. *Journal of the American Statistical Association*, **80**, 759–766.

Maunu, T. and Lerman, G. (2019) Robust subspace recovery with adversarial outliers. *CoRR*, **abs/1904.03275**. URL: `http://arxiv.org/abs/1904.03275`.

Mianjy, P. and Arora, R. (2018) Stochastic pca with ℓ_2 and ℓ_1 regularization. In *International Conference on Machine Learning*, 3531–3539.

Mirsky, L. (1960) Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, **11**, 50–59.

Nemirovski, A., Juditsky, A., Lan, G. and Shapiro, A. (2009) Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, **19**, 1574–1609.

Netrapalli, P., Niranjan, U., Sanghavi, S., Anandkumar, A. and Jain, P. (2014) Non-convex robust pca. In *Advances in Neural Information Processing Systems*, 1107–1115.

Nie, J., Kotłowski, W. and Warmuth, M. K. (2016) Online pca with optimal regret. *The Journal of Machine Learning Research*, **17**, 6022–6070.

Oja, E. (1982) Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, **15**, 267–273.

Oja E., K. J. (1985) On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of mathematical analysis and applications*, **106**, 69–84.

Olfat, M. and Aswani, A. (2019) Convex formulations for fair principal component analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 663–670.

Overton, M. L. and Womersley, R. S. (1992) On the sum of the largest eigenvalues of a symmetric matrix. *SIAM Journal on Matrix Analysis and Applications*, **13**, 41–45.

Pearson, K. (1901) Principal components analysis. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, **6**, 559.

Sanger, T. D. (1989) Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural networks*, **2**, 459–473.

Schönemann, P. H. (1966) A generalized solution of the orthogonal procrustes problem. *Psychometrika*, **31**, 1–10.

Shalev-Shwartz, S. and Ben-David, S. (2014) *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Shalev-Shwartz, S., Singer, Y., Srebro, N. and Cotter, A. (2011) Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, **127**, 3–30.

Shalev-Shwartz, S. and Srebro, N. (2008) Svm optimization: inverse dependence on training set size. In *Proceedings of the 25th international conference on Machine learning*, 928–935.

Shalev-Shwartz, S. and Tewari, A. (2011) Stochastic methods for l 1-regularized loss minimization. *The Journal of Machine Learning Research*, **12**, 1865–1892.

Shamir, O. (2015) A stochastic pca and svd algorithm with an exponential convergence rate. In *International Conference on Machine Learning*, 144–152.

— (2016) Fast stochastic algorithms for svd and pca: Convergence properties and convexity. In *International Conference on Machine Learning*, 248–256.

Shamir, O. and Zhang, T. (2013) Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International conference on machine learning*, 71–79.

Tibshirani, R. (2016) Lecture 21: Alternating direction method of multipliers (ADMM). In *Convex Optimization*. Pittsburg, Pennsylvania. URL: `http://www.stat.cmu.edu/ ryantibs/convexopt/`. Accessed on 05.08.2020.

Tsuda, K., Rätsch, G. and Warmuth, M. K. (2005) Matrix exponentiated gradient updates for on-line learning and bregman projection. *Journal of Machine Learning Research*, **6**, 995–1018.

Vaswani, N., Bouwmans, T., Javed, S. and Narayanamurthy, P. (2018) Robust subspace learning: Robust pca, robust subspace tracking, and robust subspace recovery. *IEEE signal processing magazine*, **35**, 32–55.

Vu, V. Q., Cho, J., Lei, J. and Rohe, K. (2013a) Fantope projection and selection: A near-optimal convex relaxation of sparse pca. In *Advances in neural information processing systems*, 2670–2678.

Vu, V. Q., Lei, J. et al. (2013b) Minimax sparse principal subspace estimation in high dimensions. *The Annals of Statistics*, **41**, 2905–2947.

Warmuth, M. K. and Kuzmin, D. (2008) Randomized online pca algorithms with regret bounds that are logarithmic in the dimension. *Journal of Machine Learning Research*, **9**, 2287–2320.

Wright, J., Ganesh, A., Rao, S., Peng, Y. and Ma, Y. (2009) Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in neural information processing systems*, 2080–2088.

Xu, Z. and Gao, X. (2018) On truly block eigensolvers via riemannian optimization. In *International Conference on Artificial Intelligence and Statistics*, 168–177.

Yelick, K. (2017) More data, more science... and moore's law? URL: `https://people.eecs.berkeley.edu/ yelick/talks/data/MoreData-Utah17.pdf`. Invited talk given at University of Utah, Organick Lecture, Salt Lake City, October 17, 2017.

Yi, X., Park, D., Chen, Y. and Caramanis, C. (2016) Fast algorithms for robust pca via gradient descent. In *Advances in neural information processing systems*, 4152–4160.

Zhang, T. and Lerman, G. (2012) A novel m-estimator for robust pca. *stat*, **1050**, 5.

Zhang, T. and Yang, Y. (2018) Robust pca by manifold optimization. *The Journal of Machine Learning Research*, **19**, 3101–3139.