NORTHWESTERN UNIVERSITY

Optimization Methods for Scale Invariant Problems in Machine Learning

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Industrial Engineering and Management Sciences

By

Cheolmin Kim

EVANSTON, ILLINOIS

June 2020

ABSTRACT

Optimization Methods for Scale Invariant Problems in Machine Learning

Cheolmin Kim

While optimization has received much attention in the machine learning community, most of them consider unconstrained supervised learning models such as neural networks and support vector machine. In this dissertation, we introduce a new class of optimization problems called scale invariant problems that include interesting unsupervised learning models such as PCA, ICA, GMM and KL-NMF. We develep scalable optimization algorithms for scale invariant problems and provide their convergence guarantees.

The first half of this thesis develops deterministic optimization algorithms. Specifically, we develop an iterative optimization algorithm for L1-norm kernel PCA and generalizes it to solve general scale invariant problems. In the second half, we study stochastic optimization methods. We present two stochastic PCA algorithms and develop a stochastic generalization of power iteration to solve scale invariant problems with finite-sum objective functions. Numerical experiments on various scale invariant problems reveal that the proposed algorithms not only scale better than state-of-the-art algorithms but also produce excellent quality robust solutions.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Professor Diego Klabjan for his guidance throughout my doctoral studies. I have been very fortunate to have him as my advisor who has given me exciting opportunities and guided my research with constant encouragement and assistance. Without his support, I could not grow as an independent researcher.

I would like to thank Professor Sanjay Mehrotra and Professor Ermin Wei for serving as my dissertation committee members and providing constructive feedbacks. Especially, I am grateful to Professor Mehrotra for giving me the opportunity to work on exciting projects on fractional programming and Professor Wei for sharing relevant problems in distributed optimization.

I was fortunate to have wonderful collaborators. In particular, Youngseok Kim and I have collaborated on two projects in this thesis. I have benefited from his numerous insights and our countless discussions. I was lucky to have a collaborator like Youngseok. Many important results in this thesis could not be possible without his contribution. I would also like to thank Dr. Veena Mendiratta and Dr. Kibaek Kim for hosting me as a summer intern at Nokia Bell Labs and Argonne National Laboratory and giving me opportunities to work on exciting real-world projects.

My time at Northwestern was enjoyable with many friends: Eojin Han, Andrea Trevino-Gavito, Kevin Bui, Jaehoon Koo, Yintai Ma and Mark Semelhago. I am also grateful to

other friends, staffs and faculties in the IEMS family. Especially, I would like to thank Jo Ann Yablonka, Agnes Kaminski and Stephen Erik Pedersen for their warm support.

Finally, I am deeply thankful to Sylvia and my family. Sylvia always has been there for me. Thank you for being part of my life and I look forward to our future together. My special gratitude goes out to my parents, my sister Inseon and my brother Kyoungho. Without their love and support, this thesis would never have been written. I dedicate this thesis to my parents.

To my parents, Sin Kim and Youngsun Song.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

This thesis considers a class of optimization problems called *scale invariant problem* of the form

$$\max_{x} \quad f(x) \quad \text{subject to} \quad x \in \partial\mathcal{B}_d \triangleq \{x \in \mathbb{R}^d : \|x\| = 1\} \tag{1.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is a scale invariant function. A function f is called *scale invariant*, which is rigorously defined later, if its geometric surface is invariant under constant multiplication of x. Several optimization problems in statistics and machine learning have the form of (1.1), for instance, L_p-norm kernel PCA and maximum likelihood estimation of mixture proportions, to name a few. Moreover, as studied herein, independent component analysis (ICA), Gaussian mixture model (GMM), Kullback-Leibler divergence non-neative matrix factorization (KL-NMF) and the Burer-Monteiro factorization of semidefinite programming (SDP) problem are formulated as extended settings of (1.1).

In the first chapter of this thesis, we study L_1-norm kernel PCA, which is an instance of (1.1) with the objective function $f(x) = \sum_{i=1}^{n} |\Phi(a_i)^T x|$. We present an iterative algorithm to solve L_1-norm kernel PCA and provide a convergence analysis for it. While an optimal solution of L_2-norm kernel PCA can be obtained through matrix decomposition, finding that of L_1-norm kernel PCA is not trivial due to its non-convexity and non-smoothness. We provide a novel reformulation through which an equivalent, geometrically interpretable

problem is obtained. Based on the geometric interpretation of the reformulated problem, we present a "fixed-point" type algorithm that iteratively computes a binary weight for each observation. As the algorithm requires only inner products of data vectors, it is computationally efficient and the kernel trick is applicable. In the convergence analysis, we show that the algorithm converges to a local optimal solution in a finite number of steps. Moreover, we provide a rate of convergence analysis, which has been never done for any L_1-norm PCA algorithm, proving that the sequence of objective values converges at a linear rate. Numerical experiments show that the algorithm is robust in the presence of entry-wise perturbations and computationally scalable, especially in a large-scale setting. Moreover, we introduce an application to outlier detection where the model based on the proposed algorithm outperforms the benchmark algorithms.

Based on the observation that the same approach can be used to develop an algorithm for general scale invariant objective functions, we study scale invariant problems in the second chapter and derive an algorithm called *scale invariant power iteration* (SCI-PI). SCI-PI has a general form of power iteration that finds the leading eigenvector of a matrix. Since a stationary point of (1.1) is an eigenvector of the Hessian evaluated at the point, the scale invariant problem can be locally seen as a leading eigenvector problem near a local optimal solution. Our convergence analysis reveals that SCI-PI attains local linear convergence with a generalized convergence guarantee of power iteration. Moreover, we discuss some extended settings of (1.1) and provide similar convergence results. In numerical experiments, we introduce applications to ICA, GMM and KL-NMF. Experimental results demonstrate that SCI-PI is competitive to state-of-the-art benchmark algorithms and often yield better solutions.

In the third chapter, we consider the PCA problem whose objective function $f(x) = \frac{1}{2n}\sum_{i=1}^{n}(a_i^T x)^2$ consists of finitely many convex quadratic functions. This chapter presents two stochastic variance-reduced PCA algorithms and provide their convergence analyses. By deriving explicit forms of step size, epoch length and batch size to ensure the optimal runtime, we show that the proposed algorithms can attain the optimal runtime with any batch sizes. Our novel approach, which studies the optimality gap as a ratio of two expectation terms, allows us to establish global convergence of the algorithms. The framework in our analyses is general and can be used to analyze other stochastic variance-reduced PCA algorithms and improve their analyses. Moreover, we introduce practical implementations of the algorithms which do not require hyper-parameters. The experimental results show that the proposed methodsd outperform other stochastic variance-reduced PCA algorithms regardless of the batch size.

The last chapter studies a stochastic variance-reduced algorithm to solve scale invariant problems with finite-sum objective functions and provides a convergence analysis. Specifically, we develop a stochastic generalization of scale invariant power iteration, which specializes to power iteration when full-batch is used for the PCA problem. The convergence analysis that shows the expectation of the optimality gap decreases at a linear rate under some conditions on initial iterate, step size, batch size and epoch length. Numerical experiments on the KL-NMF problem using real and synthetic datasets demonstrate that the proposed stochastic approach not only converges faster than state-of-the-art deterministic algorithms but also produces excellent quality robust solutions.

CHAPTER 2

L1-norm Kernel PCA

2.1. Introduction

Principal Component Analysis (PCA) is one of the most popular dimensionality reduction techniques [**34**]. Given a large set of possibly correlated features, it attempts to find a small set of features (*principal components*) that retain as much information as possible. To generate such new dimensions, it linearly transforms original features by multiplying *loading vectors* in a way that newly generated features are orthogonal and have the largest variance.

In traditional PCA, variance is measured using the L_2-norm. This has a nice property in that although the problem itself is non-convex, an optimal solution can be easily found through matrix factorization. With this property and easy interpretability, PCA has been extensively used in a variety of applications. Nonetheless, it still has some limitations. First, since it generates a new dimension through a linear combination of features, it cannot capture non-linear relationships among features. Second, as it uses the L_2-norm for measuring variance, its outcome tends to be affected by influential outliers. In order to overcome these limitations, the following two approaches have been proposed.

Kernel PCA The idea of kernel PCA is to map original features into a high-dimensional feature space, and perform PCA in that high-dimensional feature space [**71**]. Using a non-linear mapping, it can capture non-linear relationships among features in an

efficient way using the *kernel trick*. Using the trick, principal components can be computed with no explicit mapping.

L_1-norm PCA To alleviate the effects of influential outliers, L_1-norm PCA uses the L_1-norm instead of the L_2-norm to measure variance. The L_1-norm is more advantageous than the L_2-norm in presence of observations having large feature values since it is less influenced by them. Using this property, more robust results can be obtained by L_1-norm PCA in the presence of influential outliers.

In this work, we combine the two approaches for the variance maximization version of L_1-norm PCA. In what follows, we always refer to the variance maximization version of L_1-norm PCA which is not the same as minimizing reconstruction error with respect to the L_1-norm. Compared to L_2-norm kernel PCA, the kernel version of L_1-norm PCA is a hard problem in that it is not only non-convex but also non-smooth. However, through a novel reformulation, we convert it to a geometrically interpretable problem where the objective is to minimize the L_2-norm of a vector subject to a linear constraint consisting of terms involving the L_1-norm. For the reformulated problem, we present a "fixed point" type algorithm that iteratively computes a weight of -1 or 1 for each observation using the kernel matrix and previous weights. We show that the kernel trick is applicable to this algorithm. Moreover, we prove that the algorithm converges to a local optimal solution in a finite number of steps and the sequence of objective values converges at a linear rate. In numerical experiments, we computationally investigate the robustness of the algorithm and introduce an application to outlier detection. We also provide a runtime comparison to other robust kernel PCA algorithms and L_2-norm kernel PCA. The contributions of this work are summarized as follows.

1. We provide a novel reformulation of L_1-norm kernel PCA and present an iterative algorithm based on the geometric interpretation of the reformulated problem. This approach is not specific to L_1-norm kernel PCA but can be applied to a more general problem. Particularly, its application to L_2-norm PCA results in Power iteration [**26**].
2. We not only prove convergence but also provide a rate of convergence analysis. Although many algorithms have been proposed for L_1-norm PCA, none of them provided a rate of convergence analysis. We stress that our analysis is for the kernel version which clearly covers L_1-norm PCA. Through a novel analysis, we show that the algorithm attains a linear rate of convergence.
3. We introduce a methodology based on L_1-norm kernel PCA for outlier detection and demonstrate that it outperforms the benchmark algorithms.

The work is organized as follows. Section 2.2 reviews related works and points out how our work is different. Section 2.3 introduces a novel reformulation of L_1-norm kernel PCA and provides a geometric interpretation behind it. Based on the geometric interpretation, we present an iterative algorithm in Section 2.4. Section 2.5 provides a convergence analysis for it and the experimental results are followed in Section 2.6.

2.2. Related Works

Extracting a low-rank representation from a large matrix is an important problem in statistics and machine learning. In a variety of contexts, many previous works [**15**, **16**, **51**, **78**] have been proposed to address this problem. Recovering a low-rank matrix from a sampling of its entries is studied in [**16**]. Given that the number of sampled entries is

sufficiently large, exact recovery is guaranteed with high probability by solving a simple convex optimization problem [**16**]. Assuming that a data matrix can be decomposed into the sum of a low-rank matrix L_0 and a sparse matrix S_0, a convex program (known as *robust PCA*) that minimizes a weighted combination of the nuclear norm of L_0 and the L_1 norm of S_0 is presented in [**15**]. Also, a variant of robust PCA that identifies outliers by additionally imposing a column-sparse structure on S_0 is considered in [**78**]. Under some mild conditions, exact recovery is shown for both models [**15**, **78**]. Moreover, exact recovery of mixture data is studied in [**50**–**53**]. Utilizing a dictionary matrix, low-rank representation (LRR) [**51**] is shown to better handle mixture data than robust PCA. While matrix recovery is the main focus of theses works, our work considers dimensionality reduction with emphasis on robustness, especially focusing on kernel PCA with the L_1-norm.

To reduce the number of features in a robust way, the L_1-norm has been involved in many PCA studies [**12**, **56**, **58**, **62**, **64**–**66**] and subspace estimation formulations [**19**, **37**]. Finding a subspace onto which the L_1 projections of data vectors have the smallest reconstruction error is studied in [**12**]. Based on the observation that the L_1 projection occurs along a single unit direction, it finds an optimal subspace for each unit direction by solving d least absolute deviation regression problems, each having one dimension as a dependent variable while having the other dimensions as independent variables. Using linear programming, this approach can find a global optimal subspace in polynomial time [**12**].

Minimizing reconstruction error with respect to the L_1-norm is considered in [**37**,**65**,**66**]. While the PCA problem of minimizing $\|M - XX^TM\|_1$ subject to $X^TX = I$ is considered

in [**65**], the subspace estimation problem of minimizing $E(U,V) = \|M - UV\|_1$ is studied in [**37**] where M is a data matrix. In order to solve the former problem, an iterative algorithm that computes a weight for each observation and applies L_2-norm PCA on the weighted data matrix is presented in [**65**]. On the other hand, the latter problem is solved using alternative convex minimization based on the observation that $E(U,V)$ becomes a convex function once U or V is known. It alternatively optimizes one matrix at a time while keeping the other one fixed, repeating this process until convergence. Also, a subspace estimation formulation that minimizes reconstruction error with respect to the R_1-norm, $\|M - UV\|_{R_1} = \sum_{i=1}^{n} \|x_i - U\mathrm{v}_i\|_2$ where x_i is the i^{th} column of M and v_i is that of V, is presented in [**19**]. Since this formulation minimizes the sum of distances with respect to the L_2-norm, it is different from L_2-norm PCA which minimizes the sum of squared distances with respect to the L_2-norm. Nonetheless, they share the same property that they have a unique global solution which is rotational invariant [**19**].

Maximizing variance with respect to the L_1-norm, which we refer to as L_1-norm PCA, is studied in [**56**, **58**, **62**, **64**]. Our work also considers this formulation rather than the previous two since it has a favorable structure in that an optimal solution can be represented as a linear combination of data vectors with a weight of -1 or 1. L_1-norm PCA is shown to be NP-hard in [**56**] and [**58**]. Nevertheless, an algorithm finding a global optimal solution is proposed in [**56**]. Utilizing the auxiliary-unit-vector technique [**36**], it computes a global optimal solution with complexity $\mathcal{O}(n^{pr+p-1})$ where n is the number of observations, r is the rank of the data matrix, and p is the desired number of principal components. Assuming r and p are fixed, the runtime of this algorithm is polynomial in n. However, if n, p, r are large, it can be computationally prohibitive. Instead of finding a

global optimal solution which is intractable in general, our work focuses on developing an efficient algorithm finding a local optimal solution for L_1-norm kernel PCA.

Recognizing the hardness of L_1-norm PCA, an approximation algorithm is presented in [**58**] based on the known Nesterov's theorem [**61**]. In this work, L_1-norm PCA is relaxed to a semi-definite programming (SDP) problem and alternatively, the SDP relaxation is considered. After solving the relaxed problem, it generates a random vector and uses randomized rounding to produce a feasible solution. This randomized algorithm is a $\sqrt{2/\pi}$-approximate algorithm in expectation. To achieve this approximation ratio with high probability, it performs randomized rounding multiple times and takes the one having the best objective value. Rather than providing an approximation guarantee by solving a relaxed problem, our work directly considers the kernel version of L_1-norm PCA and develops an efficient algorithm finding a local optimal solution.

Another approach utilizing a known mathematical programming model is introduced in [**64**] where the author proposes an iterative algorithm that solves a mixed integer programming problem in each iteration. Given an orthonormal matrix of loading vectors, it perturbs the matrix slightly in a way that the resulting matrix yields the largest objective value. After the perturbation, it uses singular value decomposition to recover orthogonality. The algorithm is completely different from the one proposed herein and the sequence of objective values does not necessarily improve over iterations. Unlike it, our algorithm guarantees that the sequence of objective values keeps improving and converges at a linear rate.

A simple numerical algorithm finding a local optimal solution is proposed in [**42**]. In this work, an optimal solution is assumed to have a certain form, and weights involved in

that form are updated in each iteration, improving the objective value. A similar algorithm and its extended version that finds multiple loading vectors at once are derived in [**62**] utilizing an optimization algorithm for general L_1-norm maximization problems. In the case of linear kernel, our algorithm uses the same framework as the one in [**42**] and [**62**]. However, while the algorithm in [**42**] is derived without any justification, we provide a geometric interpretation behind the algorithm, which is different from the derivation in [**62**]. Moreover, we provide a rate of convergence analysis and introduce a kernel version, which are not considered in [**42**] and [**62**].

On other hand, the kernel version of L_1-norm PCA has been rarely studied. Due to the difficulty of applying the kernel trick to L_1-norm kernel PCA, an alternative method named *nonlinear projection trick* is applied in [**43**]. Based on the finding that an optimal loading vector lies in the span of $\Phi(A)^T U \Lambda^{-1/2}$ where $\Phi(A)$ is a high-dimensionally mapped data matrix and $U\Lambda U^T$ is the eigenvalue decomposition of the kernel matrix K, it alternatively considers L_1-norm PCA having $U\Lambda^{1/2}$ in place of $\Phi(A)$ and solves it using the algorithm in [**42**]. Another kernel extension of L_1-norm PCA is studied in [**77**]. In this work, a linear system involving a kernel matrix is solved in each iteration and the resulting solution is used to update the iterate. While the algorithms in [**43**] and [**77**] entail either eigenvalue decomposition or solving a linear system, our algorithm requires only a matrix-vector multiplication in each iteration, making it suitable in a large-scale setting.

2.3. Reformulations

We consider L_1-norm PCA in a high-dimensional feature space F. Suppose we map data vectors $a_i \in \mathbb{R}^d$, $i = 1, \ldots, n$ into a feature space F by a possibly non-linear mapping

$\Phi : \mathbb{R}^d \to F$. Assuming that each feature is standardized with a mean of 0 and standard deviation of 1 and that the kernel matrix K defined by $K_{ij} = \Phi(a_i)^T\Phi(a_j)$ satisfies $K_{ii} > 0$ for $1 \leq i \leq n$ and $|K_{ij}| < \infty$ for $1 \leq i, j \leq n$, the kernel version of L_1-norm PCA is formulated as

$$\max_{x} \quad f(x) = \sum_{i=1}^{n} |\Phi(a_i)^T x| \quad \text{subject to} \quad x \in \partial\mathcal{B}_d. \tag{2.1}$$

This formulation having $\Phi(a_i)$ in place of a_i extends the variance maximization version of L_1-norm PCA in the obvious way and is also considered in [**43**, **77**]. In this formulation, we only consider extracting the first loading vector. This assumption is justifiable since the subsequent loading vectors can be found by repeatedly solving (2.1). For example, once we obtain the first loading vector x^*, we can find the second loading vector by solving (2.1) with $\Phi(a_i) - x^*(\Phi(a_i)^T x^*)$ in place of $\Phi(a_i)$.

Solving (2.1) is not trivial since it has a convex non-smooth objective function to maximize and a Euclidean unit ball constraint. In order to better understand the problem and set an algorithmic foundation, we reformulate (2.1) as

$$\min_{w} \quad g(w) = \|w\|_2 \quad \text{subject to} \quad \sum_{i=1}^{n} |\Phi(a_i)^T w| = 1. \tag{2.2}$$

In what follows, all vector norms are L_2, so we drop the subscript for notational convenience. In order to prove the equivalence of (2.1) and (2.2), we argue that an optimal solution of one formulation can be derived from an optimal solution of the other formulation by means of some mapping. Two optimization problems are equivalent if there exists some mapping h such that if x^* is an optimal solution to one problem, then $h(x^*)$ is an optimal solution to the other problem, and vice versa for a possible different mapping function [**10**].

Proposition 2.3.1. *Let x^* and w^* be an optimal solution to* (2.1) *and* (2.2)*, respectively. Then, $\hat{x} = \dfrac{w^*}{\|w^*\|}$ and $\hat{w} = \dfrac{x^*}{\sum_{i=1}^n |\Phi(a_i)^T x^*|}$ are optimal solution to* (2.1) *and* (2.2)*, respectively.*

Proof. It is obvious that $\hat{x}$ is feasible to (2.1). To derive a contradiction, suppose that $\hat{x}$ is not optimal to (2.1). Then, there exists some feasible $\bar{x}$ such that $\sum_{i=1}^n |\Phi(a_i)^T \hat{x}| < \sum_{i=1}^n |\Phi(a_i)^T \bar{x}|$. Let $\bar{w} = \dfrac{\bar{x}}{\sum_{i=1}^n |\Phi(a_i)^T \bar{x}|}$. Then, from $\|\bar{x}\| = 1$, we have

$$g(\bar{w}) = \frac{\|\bar{x}\|}{\sum_{i=1}^n |\Phi(a_i)^T \bar{x}|} = \frac{1}{\sum_{i=1}^n |\Phi(a_i)^T \bar{x}|}.$$

On the other hand, we obtain $g(w^*) = \dfrac{1}{\sum_{i=1}^n |\Phi(a_i)^T \hat{x}|}$ since $w^* = \dfrac{\hat{x}}{\sum_{i=1}^n |\Phi(a_i)^T \hat{x}|}$ and $\|\hat{x}\| = 1$. This implies that $g(w^*) > g(\bar{w})$, which contradicts the assumption that w^* is optimal to (2.2). Therefore, $\hat{x}$ is optimal to (2.1).

It is easy to check that $\hat{w}$ is a feasible solution to (2.2). Suppose that $\hat{w}$ is not optimal to (2.2). Then, there exists some feasible $\tilde{w}$ such that $\|\tilde{w}\| < \|\hat{w}\|$. As $\tilde{w}$ is feasible to (2.2), we have $\sum_{i=1}^n |\Phi(a_i)^T \tilde{w}| = 1$. Let $\tilde{x} = \dfrac{\tilde{w}}{\|\tilde{w}\|}$. Then, we have

$$f(\tilde{x}) = \sum_{i=1}^n |\Phi(a_i)^T \tilde{x}| = \frac{\sum_{i=1}^n |\Phi(a_i)^T \tilde{w}|}{\|\tilde{w}\|} = \frac{1}{\|\tilde{w}\|}.$$

In the same way, we obtain $f(x^*) = \dfrac{1}{\|\hat{w}\|}$ from $x^* = \dfrac{\hat{w}}{\|\hat{w}\|}$. This leads to $f(x^*) < f(\tilde{x})$, which contradicts the assumption that x^* is an optimal solution of (2.1). Therefore, $\hat{w}$ is optimal to (2.2) □

Let us take a look at the constraint set $\partial P = \{w \mid \sum_{i=1}^n |\Phi(a_i)^T w| = 1\}$. Geometrically, this constraint set is symmetric with respect to the origin and represents the boundary

of polytope $P = \{w \big| \sum_{i=1}^{n} |\Phi(a_i)^T w| \leq 1\}$. It is easy to check that P is a polytope since it can be written as the intersection of a finite set of linear inequalities each having the form of $\sum_{i=1}^{n} c_i \Phi(a_i)^T w \leq 1$ where $c_i \in \{-1, 1\}$. As the objective function measures the distance from the origin, formulation (2.2) can be understood as a problem of finding the closest point to the origin from the boundary of the polytope ∂P. The following proposition shows that an optimal solution w^* must be perpendicular to one of the faces of ∂P.

Proposition 2.3.2. *An optimal solution w^* is perpendicular to the face that it lies on.*

Proof. Let E be a face such that $E = \{w \mid \sum_{i=1}^{n} c_i^* \Phi(a_i)^T w = 1\} \cap \partial P$ where $c_i^* = \text{sgn}(\Phi(a_i)^T w^*)$ for $1 \leq i \leq n$. If w^* is not perpendicular to face E, then

$$z = \frac{\sum_{i=1}^{n} \Phi(a_i) c_i^*}{\| \sum_{i=1}^{n} \Phi(a_i) c_i^* \|^2}$$

is the closest point to the origin from $\{w \big| \sum_{i=1}^{n} c_i^* \Phi(a_i)^T w = 1\}$ having

$$\|z\| < \|w^*\|. \tag{2.3}$$

Let $\bar{w} = \dfrac{z}{\sum_{i=1}^{n} |\Phi(a_i)^T z|}$. Then, $\bar{w}$ is feasible to (2.2) and has the objective value of

$$\|\bar{w}\| = \frac{\|z\|}{\sum_{i=1}^{n} |\Phi(a_i)^T z|}. \tag{2.4}$$

From $\| \sum_{i=1}^{n} \Phi(a_i) c_i^* \|^2 = \sum_{i=1}^{n} \Phi(a_i)^T c_i^* \big(\sum_{j=1}^{n} \Phi(a_j) c_j^* \big)$, we have

$$\sum_{i=1}^{n} |\Phi(a_i)^T (\sum_{j=1}^{n} \Phi(a_j) c_j^*)| - \| \sum_{i=1}^{n} \Phi(a_i) c_i^* \|^2 \geq 0,$$

which results in

$$\sum_{i=1}^{n} |\Phi(a_i)^T z| = \frac{\sum_{i=1}^{n} |\Phi(a_i)^T (\sum_{j=1}^{n} \Phi(a_j) c_j^*)|}{\| \sum_{i=1}^{n} \Phi(a_i) c_i^* \|^2} \geq 1. \tag{2.5}$$

As a result, by (2.3), (2.4) and (2.5), we have $\|\bar{w}\| \leq \|z\| < \|w^*\|$, which contradicts the assumption that w^* is optimal to (2.2). Therefore, w^* must be perpendicular to E. $\square$

Proposition 2.3.2 is important since it helps to characterize the form of an optimal solution x^*. From Proposition 2.3.2, we obtain the following corollary.

Corollary 2.3.3. *An optimal solution w^* of (2.2) has the form of $w^* = \frac{y^*}{\sum_{i=1}^{n} |\Phi(a_i)^T y^*|}$ for some y^* and c^* such that $y^* = \sum_{i=1}^{n} \Phi(a_i) c_i^*$ and $c_i^* = \operatorname{sgn}(\Phi(a_i)^T y^*)$ for $1 \leq i \leq n$.*

The characterization of an optimal loading vector using a sign vector is first proposed in [**42**] without any justification. However, we provide a derivation based on the geometry of ∂P, which is different from the one in [**62**] that uses the KKT conditions. Moreover, since $\sum_{i=1}^{n} |\Phi(a_i)^T y^*| = \sum_{i=1}^{n} c_i^* \Phi(a_i)^T y^* = \| \sum_{i=1}^{n} \Phi(a_i) c_i^* \|^2$, we have

$$\|w^*\| = \frac{\|y^*\|}{\sum_{i=1}^{n} |\Phi(a_i)^T y^*|} = \frac{1}{\| \sum_{i=1}^{n} \Phi(a_i) c_i^* \|}. \tag{2.6}$$

We can further show that an optimal solution of formulation (2.2) can be found from an optimal solution of the following binary problem,

$$\max_{c} \quad \| \sum_{i=1}^{n} \Phi(a_i) c_i \|^2 \quad \text{subject to} \quad c_i \in \{-1, 1\},\ 1 \leq i \leq n. \tag{2.7}$$

Proposition 2.3.4. *Let c^* be an optimal solution of binary formulation* (2.7)*. Then, $y^* = \sum_{i=1}^n \Phi(a_i)c_i^*$ satisfies*

$$c_i^* = \operatorname{sgn}(\Phi(a_i)^T y^*), \tag{2.8}$$

for $1 \leq i \leq n$. Moreover, $w^* = \dfrac{y^*}{\sum_{i=1}^n |\Phi(a_i)^T y^*|}$ is an optimal solution to (2.2)*.*

Proof. To deduce a contradiction, let us assume that there exists some nonempty set $J \subset \{1, \ldots, n\}$ such that $c_j^* = -\operatorname{sgn}(\Phi(a_j)^T y^*)$ for $j \in J$. Since c^* is an optimal solution of (2.7), flipping the sign of c_j^* for $j \in J$ must not improve the objective value of (2.7). However, for any $j \in J$, flipping the sign of c_j^* results in $\|y^* - 2\Phi(a_j)c_j^*\|^2 > \|y^*\|^2$ since $\|y^* - 2\Phi(a_j)c_j^*\|^2 = \|y^*\|^2 + 4|\Phi(a_j)^T y^*| + 4\|\Phi(a_j)\|^2$. This contradicts the assumption that c^* is an optimal solution to (2.7). Therefore, y^* must satisfy $c_i^* = \operatorname{sgn}(\Phi(a_i)^T y^*)$ for $1 \leq i \leq n$. Since y^* and c^* satisfy (2.8) and c^* maximizes the objective value of (2.7), w^* is a minimizer of (2.2) due to Corollary 2.3.3 and (2.6). □

The following result has been shown in [**56**] for the linear kernel case but here we generalize it.

Corollary 2.3.5. *Formulation* (2.2) *is equivalent to formulation* (2.7)*.*

Proof. Based on Corollary 2.3.3 and (2.6), we can formulate (2.2) as

$$\max_{c,y} \ \|\sum_{i=1}^n \Phi(a_i)c_i\|^2 \quad \text{subject to} \quad y = \sum_{i=1}^n \Phi(a_i)c_i, \ c_i = \operatorname{sgn}(\Phi(a_i)^T y), \ 1 \leq i \leq n.$$

Since an optimal solution c^* to (2.7) satisfies the constraints of the above optimization problem by Proposition 2.3.4, the two formulations are essentially the same. □

It is interesting to note that we can reduce formulation (2.7) to the weighted max-cut problem since

$$\|\sum_{i=1}^{n}\Phi(a_i)c_i\|^2 = \sum_{i,j=1}^{n}K_{ij} + \sum_{i,j=1}^{n}(-2K_{ij})\Big(\frac{1-c_ic_j}{2}\Big). \tag{2.9}$$

Using the above reduction, we can alternatively consider the weighted max-cut problem on a complete graph with weight $w_{ij} = -K_{ij}$. Therefore, a popular approximation algorithm for the weighted max-cut problem [**25**] can be used to solve (2.7). However, due to the additional constant terms in (2.9), this does not imply a constant worst case approximation ratio algorithm for (2.7).

2.4. Algorithm

In this section, we develop an algorithm that finds a local optimal solution to (2.2) based on the findings in Section 2.3. Before giving details of the algorithm, we first provide the idea behind the algorithm.

The main idea of the algorithm is to move along the boundary of P so that the L_2-norm of w_k successively decreases. Figure 2.1 illustrates a step of the algorithm. Starting with an iterate w_k, we first identify the hyperplane h_k which the current iterate w_k lies on. After identifying the equation of h_k, we find the closest point to the origin from h_k, which we denote by z_k. After that, we obtain w_{k+1} by projecting z_k to the constraint set ∂P, which is done by multiplying an appropriate scalar between 0 and 1. We repeat this process until the sequence of iterates $\{w_k\}$ converges.

Now, we develop an algorithm based on the above idea. Given w_k, let c^k be the sign vector $c^k = [c_1^k, \ldots, c_n^k]^T$ such that $c_i^k = \text{sgn}(\Phi(a_i)^T w_k)$ for $1 \le i \le n$ and y_k be the normal

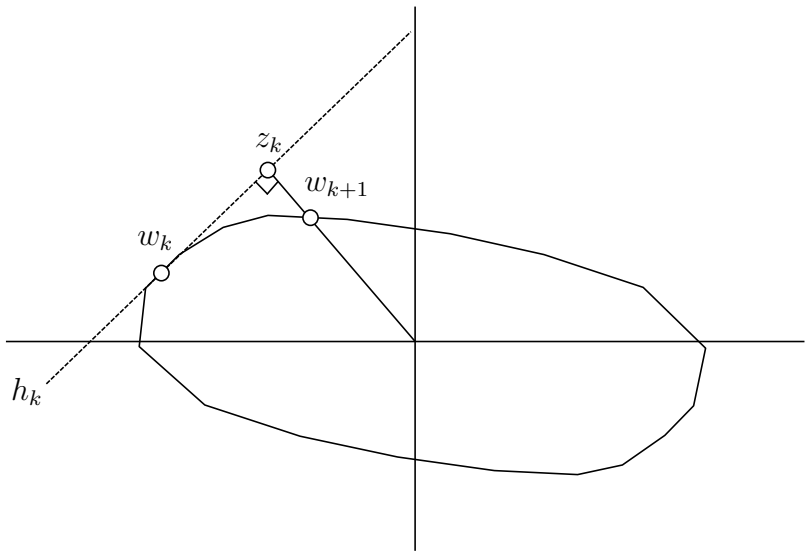

Figure 2.1. Geometric derivation of the algorithm

vector at w_k defined as

$$y_k = \sum_{i=1}^{n} \Phi(a_i) c_i^k. \tag{2.10}$$

Using the normal vector y_k at w_k, we can find the equation of hyperplane h_k as

$$y_k^T (w - w_k) = 0. \tag{2.11}$$

The closest point z_k to the origin from h_k has the form of

$$z_k = s y_k. \tag{2.12}$$

Plugging (2.12) into (2.11), we have

$$s = \frac{y_k^T w_k}{y_k^T y_k}, \quad z_k = \frac{y_k^T w_k}{y_k^T y_k} y_k. \tag{2.13}$$

Projecting z_k to ∂P, we obtain

$$w_{k+1} = \frac{z_k}{\sum_{i=1}^{n} |\Phi(a_i)^T z_k|}. \tag{2.14}$$

Using

$$y_k^T w_k = \sum_{i=1}^{n} \Phi(a_i)^T w_k c_i^k = \sum_{i=1}^{n} |\Phi(a_i)^T w_k| = 1, \tag{2.15}$$

we can further write (2.13) as

$$z_k = \frac{y_k}{\|y_k\|^2}, \tag{2.16}$$

which leads to

$$w_{k+1} = \frac{y_k}{\sum_{i=1}^{n} |\Phi(a_i)^T y_k|}. \tag{2.17}$$

Also, from (2.10) and $\sum_{i=1}^{n} |\Phi(a_i)^T y_k| = \sum_{i=1}^{n} \Phi(a_i)^T y_k c_i^k = (c^k)^T K c^k$, we can represent w_{k+1} as a function of c^k as

$$w_{k+1} = \frac{\sum_{i=1}^{n} \Phi(a_i) c_i^k}{(c^k)^T K c^k}. \tag{2.18}$$

Since $c_i^{k+1} = \text{sgn}(\Phi(a_i)^T x_{k+1}) = \text{sgn}(K_{i\cdot} c^k)$, we can update c_i^{k+1} using only K and c^k by $c^{k+1} = \text{sgn}(K c^k)$. Moreover, from

$$\|w_{k+1} - w_k\|^2 = \frac{(c^k - c^{k+1})^T K (c^k - c^{k+1})}{(c^k)^T K c^k (c^{k+1})^T K c^{k+1}},$$

the termination criteria $w_{k+1} = w_k$ can be represented by $(c^k - c^{k+1})^T K (c^k - c^{k+1}) = 0$.

Due to non-convexity of the problem, the algorithm can be stuck at a local optimum unless it is initialized close to a global optimum. In order to obtain a good initial iterate, we consider each $\Phi(a_j)$ and select the one such that $\dfrac{\Phi(a_j)}{\|\Phi(a_j)\|}$ yields the largest objective value for f, which is computed by

$$\frac{\Sigma_{i=1}^n |\Phi(a_i)^T \Phi(a_j)|}{\|\Phi(a_j)\|} = \frac{\Sigma_{i=1}^n |K_{ij}|}{\sqrt{K_{jj}}}. \tag{2.19}$$

Once we find the index j^* maximizing (2.19), we set

$$w_0 = \frac{\Phi(a_{j^*})}{\sum_{i=1}^n |\Phi(a_i)^T \Phi(a_{j^*})|}, \quad c_i^0 = \operatorname{sgn}(\Phi(a_i)^T w_0) = \operatorname{sgn}(\Phi(a_i)^T \Phi(a_{j^*})) = \operatorname{sgn}(K_{ij^*}).$$

Summarizing all the above, we obtain Algorithm 1.

Algorithm 1 L_1-norm Kernel PCA

Input: kernel matrix K
find j^* that maximizes (2.19)
initialize the sign vector c^0 with $c_i^0 = \operatorname{sgn}(K_{ij^*})$
$k \leftarrow -1$
repeat
  $k \leftarrow k + 1$
  compute $c^{k+1} = \operatorname{sgn}(Kc^k)$
until $(c^k - c^{k+1})^T K (c^k - c^{k+1}) = 0$
Output: sign vector c^*

Once we get the output c^* from Algorithm 1, we can compute principal scores with no explicit mapping. For example, the principal component of the i^{th} observation can be computed by

$$\frac{\Phi(a_i)^T x^*}{\|x^*\|} = \frac{\sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_j^*}{\sqrt{\sum_{i=1}^n \sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i^* c_j^*}} = \frac{K_{i\cdot} c^*}{\sqrt{(c^*)^T K c^*}}.$$

Also, we can proceed to find more principal components with no explicit mapping. Noting that computing a loading vector and principal components requires only the kernel matrix, it suffices to update the kernel matrix each time a new loading vector is found. Fortunately, updating the kernel matrix can be done with no explicit mapping by

$$\widetilde{K}_{ij} = \Phi(a_i)^T\Phi(a_j) - \frac{\Phi(a_i)^T x^* \Phi(a_j)^T x^*}{\|x^*\|^2} = K_{ij} - \frac{K_{i.}c^* K_{j.}c^*}{(c^*)^T K c^*},$$

which is equivalent to $\widetilde{K} = K - \dfrac{(Kc^*)(Kc^*)^T}{(c^*)^T K c^*}$ in a matrix form.

From $y_k = \nabla f(x_k)$, update rule (2.17) can be understood as projecting a gradient $\nabla f(x_k)$ to the constraint set ∂P in each iteration. In this sense, Algorithm 1 resembles Power iteration [**26**] for solving the eigenvalue problem, and interestingly, the application of our framework to the eigenvalue problem yields the same algorithm. The framework developed in this work such as reformulation, geometric interpretation and algorithm derivation is not specific to L_1-norm kernel PCA but can be extended to solve a more general problem. For the application of this approach to general scale invariant problems (1.1), see Chapter 3.

Compared to the other L_1-norm kernel PCA algorithms [**43**, **77**] considering the same formulation (2.1), Algorithm 1 is much simple and computationally efficient as it involves just one matrix-vector multiplication in each iteration. In the case of L1-KPCA [**77**], a system of linear equations having the form of $K\eta = \Sigma_{j=1}^{n} c_j^k K_{.j}$ is repeatedly solved. Solving the above linear system is not only computationally costly but also numerically unstable since it is singular due to the presence of non-trivial solution c^k. On the other hand, KPCA-L1 [**43**] requires one matrix-vector multiplication but it does not directly consider the kernel matrix K. Instead, the eigenvalue decomposition of the kernel matrix $K = U\Lambda U^T$

must be computed before starting to find each loading vector. Also, $U\Lambda^{1/2}$ is involved in computation instead of the kernel matrix K. As Algorithm 1 entails neither solving a linear system nor computing the eigenvalue decomposition of K, it is computationally more efficient than the other algorithms.

When it comes to initialization, L1-KPCA [**77**] uses the optimal loading vector from L_2-norm kernel PCA. While KPCA-L1 [**43**] finds the data vector having the largest norm and uses its normalization for the initial iterate, Algorithm 1 finds the normalized data vector with the largest objective value for f and set it to be the initial iterate. As the initialization scheme of Algorithm 1 is based on the objective fucntion f while the others are not, it is more likely to obtain a good initial iterate compared to the others.

2.5. Convergence Analysis

In this section, we provide a convergence analysis of Algorithm 1. We first prove that the algorithm converges in a finite number of iterations, and then provide a rate of convergence analysis. Before proving the finite convergence of the algorithm, we first show that the sequence $\{\|w_k\|\}$ generated by Algorithm 1 is non-increasing.

Lemma 2.5.1. *Let $\{w_k\}$ and $\{z_k\}$ be a sequence of vectors generated by Algorithm 1 and* (2.16)*, respectively. Then, we have*

$$\|w_{k+1}\| \leq \|z_k\| \leq \|w_k\|.$$

Moreover, if $\|w_k\| = \|z_k\|$, we have $w_k = ry_k$ for some $r \in \mathbb{R}$.

Proof. The inequality $\|z_k\| \leq \|w_k\|$ follows from

$$\|w_k\|^2 - \|z_k\|^2 = \|w_k\|^2 - \frac{1}{\|y_k\|^2} = \|w_k\|^2 - \frac{(y_k^T w_k)^2}{\|y_k\|^2} = \frac{\|w_k\|^2\|y_k\|^2 - (y_k^T w_k)^2}{\|y_k\|^2} \geq 0$$

where the second equality holds follows from (2.15) and the last inequality holds due to the Cauchy-Schwarz inequality. If $\|w_k\| = \|z_k\|$, the Cauchy-Schwarz inequality becomes an equality resulting in $w_k = ry_k$ for some $r \in \mathbb{R}$.

Next, from (2.14), we have

$$\|w_{k+1}\|^2 = \frac{\|z_k\|^2}{(\sum_{i=1}^n |\Phi(a_i)^T z_k|)^2}. \tag{2.20}$$

Using (2.13), we can represent the denominator as

$$\sum_{i=1}^n |\Phi(a_i)^T z_k| = \frac{\sum_{i=1}^n |\Phi(a_i)^T y_k|}{y_k^T y_k}.$$

From

$$\sum_{i=1}^n |\Phi(a_i)^T y_k| = \sum_{i=1}^n |\Phi(a_i)^T (\sum_{j=1}^n \Phi(a_j) c_j^k)| = \sum_{i=1}^n |\sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i^k c_j^k|$$

and

$$y_k^T y_k = \sum_{i=1}^n \sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i^k c_j^k,$$

we obtain

$$\sum_{i=1}^n |\Phi(a_i)^T z_k| = \frac{\sum_{i=1}^n |\sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i^k c_j^k|}{\sum_{i=1}^n \sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i^k c_j^k} \geq 1. \tag{2.21}$$

By (2.20) and (2.21), we finally have $\|w_{k+1}\|^2 \leq \|z_k\|^2$. □

Lemma 2.5.2. *If* $\|w_k\| = \|w_{k+1}\|$, *then we have* $w_k = \dfrac{y_k}{\|y_k\|^2}$ *and* $y_k = \dfrac{w_k}{\|w_k\|^2}$, *which leads to* $w_k = w_{k+1}$.

Proof. Since $\|w_k\| = \|w_{k+1}\|$, by Lemma 2.5.1, we have $\|z_k\| = \|w_k\|$ and thus $w_k = ry_k$ for some $r \in \mathbb{R}$. Using (2.15), we have $r = \dfrac{1}{\|y_k\|^2}$, which results in $w_k = \dfrac{y_k}{\|y_k\|^2}$. In the same way, we can show $y_k = \dfrac{w_k}{\|w_k\|^2}$. Since this implies $z_k = w_k$ by (2.16), we finally have

$$w_{k+1} = \frac{z_k}{\sum_{i=1}^{n} |\Phi(a_i)^T z_k|} = \frac{w_k}{\sum_{i=1}^{n} |\Phi(a_i)^T w_k|} = w_k$$

where the first equality follows from (2.14) and the last equality holds from the feasibility of w_k. □

Theorem 2.5.3. *The sequence* $\{w_k\}$ *converges in a finite number of steps.*

Proof. Suppose the sequence $\{w_k\}$ does not converge. As an iterate w_k is solely determined by a sign vector $c^k \in \{-1, +1\}^n$, the number of possible vectors that w_k can take is finite. Therefore, if the sequence $\{w_k\}$ does not converge, some vectors must appear more than once. Without loss of generality, let $w_l = w_{l+m}$. By Lemma 2.5.1, we have $\|w_{l+m}\| = \|w_l\| \geq \|w_{l+1}\| \geq ... \geq \|w_{l+m}\|$ forcing us to have $\|w_l\| = \|w_{l+1}\| = \cdots = \|w_{l+m}\|$. This implies $w_l = w_{l+1} = \cdots = w_{l+m}$ by Lemma 2.5.2, contradicting the assumption that the sequence $\{w_k\}$ does not converge. Therefore, the sequence $\{w_k\}$ generated by Algorithm 1 must converge in a finite number of steps. □

Next, we show that the sequence of $\{\|w_k\|\}$ generated by Algorithm 1 converges at a linear rate. Although Theorem 2.5.3 shows that the algorithm converges in a finite number of steps, it may take an exponential number of steps to converge, due to the combinatorial structure of the problem, making it not appropriate in a large-scale setting. To make sure that this does not happen for Algorithm 1, we additionally prove linear convergence, which ensures that the optimality gap decreases no worse than a certain rate $\rho < 1$. Since this result implies that an ϵ-optimal local solution can be attained after $\mathcal{O}\big(\frac{1}{1-\rho}\log\frac{1}{\epsilon}\big)$ iterations, we can obtain a near-optimal solution after a sufficient number of iterations without waiting for an exponential number of steps.

Theorem 2.5.4. *Let Algorithm 1 start from w_0 and terminate with w^* at iteration k^*. Then, for $k < k^*$, we have*

$$\|w_k\| - \|w^*\| \leq \rho^k(\|w_0\| - \|w^*\|)$$

where $\rho = \max\{\rho(c) \,|\, c \in \{-1, 1\}^n,\ \rho(c) < 1\}$ *where*

$$\rho(c) = \frac{\sum_{i=1}^n \sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i c_j}{\sum_{i=1}^n |\sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i c_j|}.$$

Proof. From (2.14), we have

$$\|w_k\| = \frac{\|z_{k-1}\|}{\sum_{i=1}^n |\Phi(a_i)^T z_{k-1}|}.$$

Since $\|z_{k-1}\| \leq \|w_{k-1}\|$ holds by Lemma 2.5.1, we obtain

$$\|w_k\| \leq \frac{\|w_{k-1}\|}{\sum_{i=1}^n |\Phi(a_i)^T z_{k-1}|}. \tag{2.22}$$

Subtracting $\|w^*\|$ to (2.22), we have

$$\|w_k\| - \|w^*\| \leq \frac{\|w_{k-1}\|}{\sum_{i=1}^n |\Phi(a_i)^T z_{k-1}|} - \|w^*\| \leq \frac{1}{\sum_{i=1}^n |\Phi(a_i)^T z_{k-1}|}(\|w_{k-1}\| - \|w_*\|) \tag{2.23}$$

where the last inequality follows from (2.21).

By induction on (2.23), we obtain

$$\|w_k\| - \|w^*\| \leq \prod_{l=1}^{k} \frac{1}{\sum_{i=1}^n |\Phi(a_i)^T z_{l-1}|}(\|w_0\| - \|w^*\|). \tag{2.24}$$

From (2.21), we know that

$$\sum_{i=1}^{n} |\Phi(a_i)^T z_{l-1}| \geq 1.$$

If $\sum_{i=1}^n |\Phi(a_i)^T z_{l-1}| = 1$, then we have

$$\frac{\sum_{i=1}^n |\sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i^{l-1} c_j^{l-1}|}{\sum_{i=1}^n \sum_{j=1}^n \Phi(a_i)^T \Phi(a_j) c_i^{l-1} c_j^{l-1}} = 1,$$

resulting in

$$c_i^{l-1} = \operatorname{sgn}\Big(\sum_{j=1}^{n} \Phi(a_i)^T \Phi(a_j) c_j^{l-1}\Big).$$

Since this implies $c^l = \operatorname{sgn}(Kc^{l-1}) = c^{l-1}$, we obtain $w_l = w_{l+1}$. Therefore, as long as $l < k^*$, we must have $\sum_{i=1}^n |\Phi(a_i)^T z_{j-1}| > 1$. Since this implies $\rho(c^{j-1}) = \dfrac{1}{\sum_{i=1}^n |\Phi(a_i)^T z_{j-1}|} < 1$, using (2.24), we obtain the desired result. □

As shown in Theorem 2.5.4, no matter where the algorithm starts, the sequence of objective values of (2.2) converges at a linear rate. Now, we show that we can obtain a local optimal solution of (2.1) by scaling the output of Algorithm 1.

Theorem 2.5.5. *Let the output of Algorithm 1 be w^*. Then, $x^* = \dfrac{w^*}{\|w^*\|}$ is a local optimal solution of* (2.1).

Proof. By construction, x^* is feasible. Since w^* is the output of Algorithm 1, $y^* = \dfrac{w^*}{\|w^*\|^2}$ holds by Lemma 2.5.2. Next, consider $L(\lambda, x) = \sum_{i=1}^n |\Phi(a_i)^T x| - \lambda(\|x\|^2 - 1)$. From $\nabla_x L(\lambda, x) = \sum_{i=1}^n \operatorname{sgn}(\Phi(a_i)^T x)\Phi(a_i) - 2\lambda x$, we have

$$\nabla_x L(\lambda, x^*) = \sum_{i=1}^n \operatorname{sgn}(\Phi(a_i)^T x^*)\Phi(a_i) - 2\lambda x^* = \sum_{i=1}^n \operatorname{sgn}(\Phi(a_i)^T w^*)\Phi(a_i) - 2\lambda x^*.$$

Since $\sum_{i=1}^n \operatorname{sgn}(\Phi(a_i)^T w^*)\Phi(a_i) = y^* = \dfrac{w^*}{\|w^*\|^2} = \dfrac{x^*}{\|w^*\|}$, we have

$$\nabla_x L(\lambda, x^*) = \left(\frac{1}{\|w^*\|} - 2\lambda\right) x^*$$

Therefore, with $\lambda^* = \dfrac{1}{2\|w^*\|}$, we have $\nabla_x L(\lambda^*, x^*) = 0$, meaning that (λ^*, x^*) satisfies the first order necessary conditions. Moreover, from $\nabla_{xx} L(\lambda^*, x^*) = -2\lambda^* I \prec 0$, the second order sufficient condition is also satisfied. Since (λ^*, x^*) satisfies the first and second order conditions, from the theory of constrained optimization, x^* is a local optimal solution of (2.1). □

2.6. Numerical Experiments

In this section, we assess the robustness and scalability of Algorithm 1 by running it on several tasks and compare it with other kernel PCA algorithms. First, we apply

them on datasets having entry-wise perturbations and investigate how well each algorithm extracts principal components in a noisy setting. Next, we introduce their application to outlier detection and compare their performance with other popular outlier detection models. Lastly, we provide their runtime comparison.

In addition to Algorithm 1, the two other L_1-norm kernel PCA algorithms (KPCA-L1 [**43**], L1-KPCA [**77**]), the kernel version of R_1-norm PCA (R1-KPCA [**19**]) and L_2-norm kernel PCA (L2-KPCA [**71**]) are considered in the experiments. While R_1-norm PCA [**19**] is not originally designed to incorporate kernels, we include it as it is easy to develop a kernel variant. Other L_1-norm PCA algorithms were also considered but since it is not straightforward to develop a kernel version for them, they are disregarded.

2.6.1. Robust Extraction of PCs

To measure robustness, we first run the algorithms on datasets having entry-wise perturbations (noisy datasets) to obtain loading vectors. After that, we compute how much variation in the perturbation-excluded datasets (normal datasets) is explained by the loading vectors obtained from the noisy datasets. For this experiment, we prepare synthetic datasets having entry-wise perturbations so that loading vectors obtained by running L_2-norm kernel PCA on noisy and normal datasets are different from each other.

To generate synthetic datasets, we first construct a 1000×50 data matrix with the rank of 10 following the data generation procedure in [**65**]. While the largest size in [**65**] is 300×50, we choose the size of 1000×50 to consider larger datasets. To obtain entry-wise perturbations, we corrupt $r\%$ of observations by adding some random noises. We refer to the resulting dataset as a noisy dataset and the noisy dataset without the entry-wise

perturbations as a normal dataset. For each value of $r \in \{5, 10, 15, 20, 25, 30\}$, we generate 10 instances.

Let K denote a kernel matrix of a normal dataset and $x_1, \ldots, x_p$ be p loading vectors obtained by running L_2-norm kernel PCA on K. Also, let $\tilde{K}$ be a kernel matrix of a noisy dataset and $\tilde{x}_1, \ldots, \tilde{x}_p$ be loading vectors obtained by running one of the kernel PCA algorithms (Algorithm 1, KPCA-L1, L1-KPCA, R1-KPCA, L2-KPCA) on $\tilde{K}$. Assuming that the normal dataset is standardized,

$$\sum_{j=1}^{p}\sum_{i=1}^{n}(\Phi(a_i)^T\tilde{x}_j)^2 = \sum_{j=1}^{p}\tilde{x}_j^T K\tilde{x}_j \tag{2.25}$$

represents the amount of variation in the normal dataset explained by the p loading vectors $\tilde{x}_1, \ldots, \tilde{x}_p$ where n is the number of observations in the normal dataset. After dividing (2.25) by $\sum_{j=1}^{p} x_j^T K x_j$, which is the maximum amount of variation in the normal dataset that the p orthogonal vectors can explain, and multiplying by 100, we get the following measure:

$$\text{(Total Explained Variation)} \quad 100 \times \frac{\sum_{j=1}^{p}\tilde{x}_j^T K\tilde{x}_j}{\sum_{j=1}^{p} x_j^T K x_j}. \tag{2.26}$$

Metric (2.26) captures how well the loading vectors obtained from the noisy dataset explain variation in the normal dataset with respect to the L_2-norm. Therefore, it can be used to measure the robustness of each kernel PCA algorithm in the presence of entry-wise perturbations. For example, if one algorithm has a value close to one, then it is robust with respect to entry-wise perturbations. Using this metric, we compare the robustness of Algorithm 1 with that of KPCA-L1, L1-KPCA, R1-KPCA, and L2-KPCA. For each value

of r, we compute (2.26) for the ten datasets with $p = 4$ and average them. We arbitrarily choose $p = 4$ since the result is consistent regardless of the choice of p. Figure 2.2 shows the results for the linear kernel and Figure 2.3 shows the results for the Gaussian kernel with the width parameter σ varying from 10 to 25.

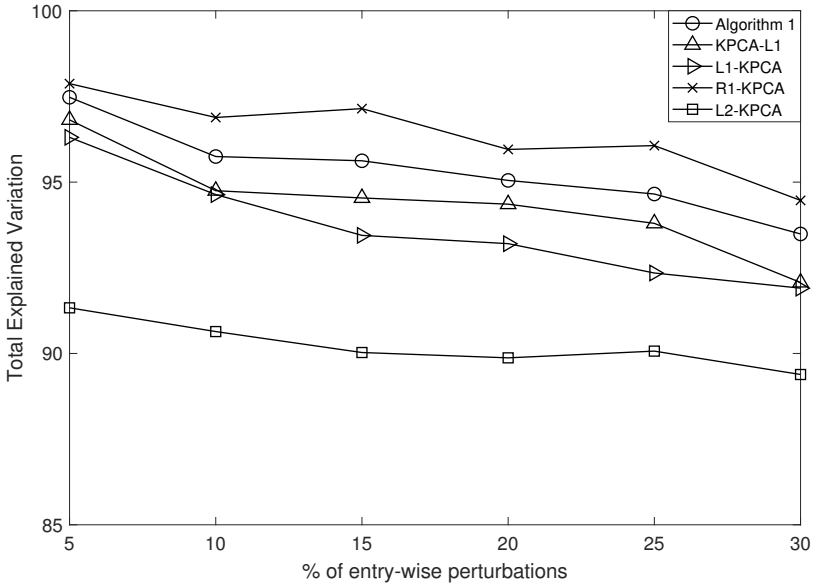

Figure 2.2. Robust extraction of PCs (linear kernel)

In the case of the linear kernel, R1-KPCA achieves the best performance for all values of r followed by the L_1-norm based kernel PCA algorithms and L2-KPCA. While the loading vectors from L2-KPCA explain about 90% of the variation, those from R1-KPCA, Algorithm 1, KPCA-L1, and L1-KPCA explain around 96%,95%,94%, and 93% of the variation, respectively. This demonstrates the robustness of the R_1-norm and L_1-norm based kernel PCA algorithms with respect to the presence of entry-wise perturbations. Among the three L_1-norm based kernel PCA algorithms, Algorithm 1 consistently outperforms KPCA-L1 and L1-KPCA by 1% and 2%, respectively. As the

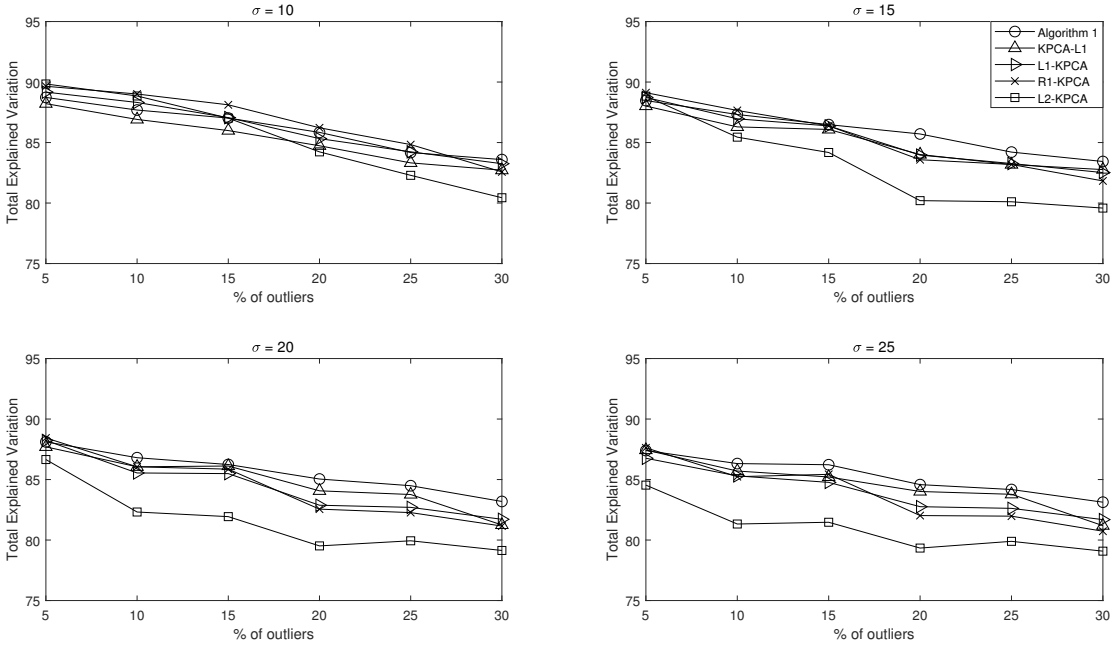

Figure 2.3. Robust extraction of PCs (Gaussian kernel with σ from 10 to 25)

percentage of corrupted observations ($r\%$) increases, the total explained variation tends to decrease for all of them but the gaps between them remain the same.

When the Gaussian kernel is used, the results are slightly different depending on the value of r and σ. If r and σ are small, the effects of entry-wise perturbations are relatively small so that all the algorithms give pretty similar results. However, if r or σ is large, the effects of entry-wise perturbations are pronounced in the kernel matrix, and therefore, the results are different depending on the robustness of the algorithms. As shown in Figure 2.3, the three L_1-norm kernel PCA algorithms and R1-KPCA outperform L2-KPCA as in the case of the linear kernel. However, while R1-KPCA achieves the best performance for the linear kernel, the L_1-norm based kernel PCA algorithms work better than R1-KPCA when the Gaussian kernel is used. Especially, Algorithm 1 outperforms all the other algorithms

if r exceeds 20. The superior performance of Algorithm 1 ranges from 1% to 5% in these cases.

2.6.2. Outlier Detection

L_2-norm PCA has been shown to be effective for anomaly detection [**74**]. The idea is to extract loading vectors using datasets consisting of only normal samples and use these loading vectors to develop a detection model. Specifically, a boundary of normal samples is constructed from the loading vectors and the boundary is used to discriminate normal and abnormal samples.

We extend this principle to outlier detection, i.e. its unsupervised counterpart. In the outlier detection setting, sample labels are not given when the model is built. Therefore, it is not possible to build a detection model solely based on normal samples. Given this context, we run robust kernel PCA algorithms on the entire dataset (with outliers) and use the resulting loading vectors to characterize a boundary of normal samples. Since these loading vectors are less influenced by outliers as illustrated in Section 2.6.1, we expect that they would better construct a normal boundary. We compare the performance of Algorithm 1 based models to that of KPCA-L1, L1-KPCA, R1-KPCA, and L2-KPCA based models as well as two other popular outlier detection models [**11**] [**49**].

2.6.2.1. Toy Examples. We first illustrate the advantage of using robust kernel PCA for outlier detection using the following two-dimensional toy examples.

Figure 2.4 displays the distribution of normal samples and outliers. As the normal samples follow a linear pattern, we run the kernel PCA algorithms with the linear kernel and represent their first loading vectors in Figure 2.4. In the figure, the first loading

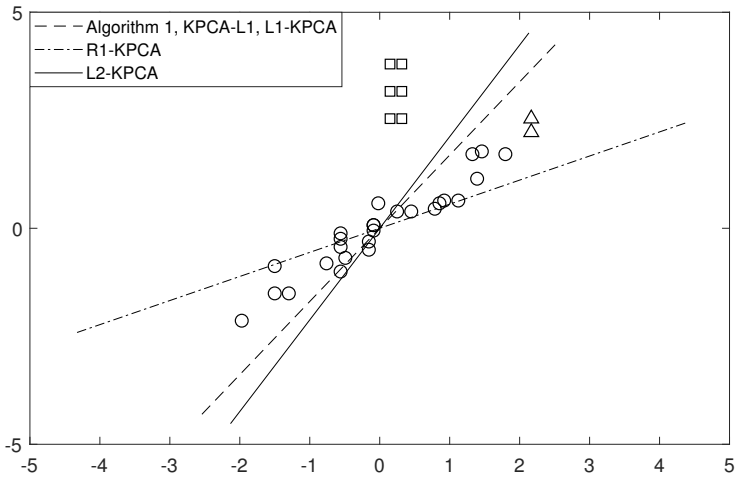

Figure 2.4. First toy example (original space)

vectors of the three L_1-norm based kernel PCA algorithms are represented using a single dashed line since they yield the same first loading vector in this example. In addition to the normal samples forming a linear pattern, there are some outliers scattered exhibiting two different patterns; the two triangle points are outliers due to their scale and the six square points are outliers since they do not follow the linear pattern. If the first loading vector exactly matches the linear pattern, outliers can be easily detected in the principal space; the triangle points can be detected due to large first principal components and the square points can be detected from large second principal components. However, due to the presence of outliers, it is impossible that the first loading vector exactly matches the linear pattern. Given this context, we use robust kernel PCA algorithms to obtain the first loading vector with lower deviation from the linear pattern.

Figure 2.5 displays the PCA results of the five kernel PCA algorithms. In the figure, the x-axis and the y-axis represents the first and the second principal component, respectively.

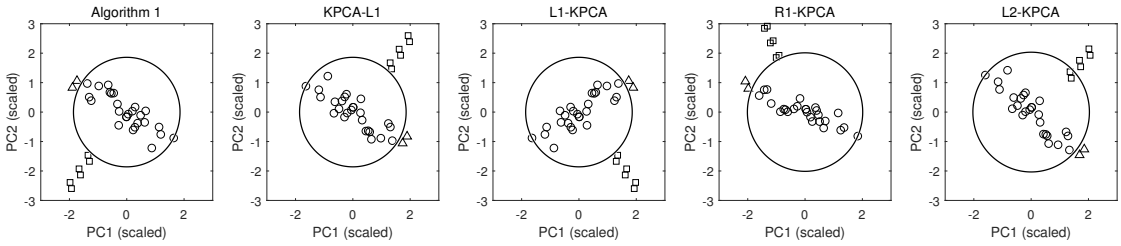

Figure 2.5. First toy example (principal space)

As shown in the figure, the triangle outliers can be easily separated by the first principal component for any kernel PCA algorithm. However, while the square outliers can be discriminated by the second principal component of the L_1-norm based kernel PCA algorithms and R1-KPCA, there exists some overlap between the normal samples and the square outliers in the range of the second principal component of L2-KPCA. As seen in the figure, two outliers appear closer to the origin than some normal samples making the circular boundary of the normal samples include them. On the other hand, all the normal samples are clearly separated from the outliers in the principal space of the L_1-norm based kernel PCA algorithms and R1-KPCA, demonstrating the advantage of using robust kernel PCA in outlier detection. This result is consistent with the findings in Figure 2.2.

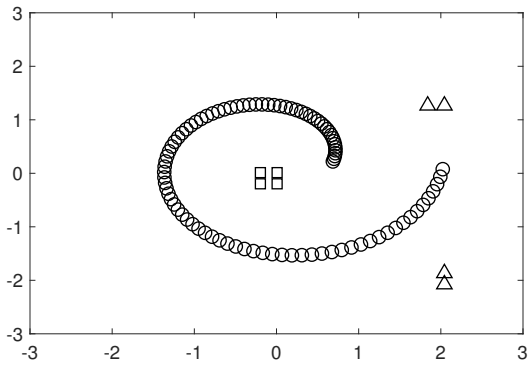

Figure 2.6. Second toy example (original space)

In order to see if the same result holds for the Gaussian kernel, we consider another example. As shown in Figure 2.6, the second example has a spiral pattern consisting of normal samples as well as two types of outliers. As in the previous example, it has both trivial outliers (the triangle points) and more challenging outliers (the square points). In order to obtain nonlinear principal components, we run the five kernel PCA algorithms with the Gaussian kernel. As Figure 2.7 displays, only Algorithm 1 succeeds to exclude the square outliers from the boundary while the other kernel PCA algorithms include them within the boundary. This superior performance of Algorithm 1 with the Gaussian kernel is consistent with the results in Section 2.6.1 and attests the effectiveness of using it for outlier detection, especially with the Gaussian kernel.

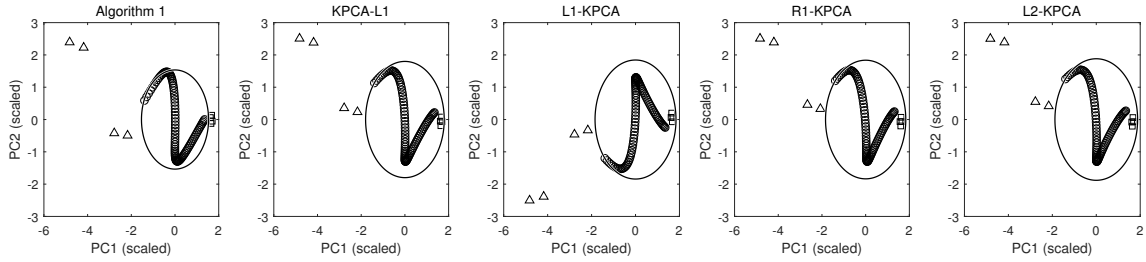

Figure 2.7. Second toy example (principal space)

2.6.2.2. Real-world Datasets. For outlier detection, we use datasets from the UCI Machine Learning Repository [**18**] and the ODDS Library [**69**], see Table 2.1.

Table 2.1. Real-world datasets for outlier detection

Data set	# samples	# features	# outliers
WBC	378	30	21 (7.6%)
Ionosphere	351	33	126 (36%)
BreastW	683	9	239 (35%)
Cardio	1831	21	176 (9.6%)
Musk	3062	166	97 (3.2%)
Mnist	7603	100	700 (9.2%)

In this experiment, we use a similar detection rule as the one in [**74**] where it is applied for anomaly detection. Let $Y \in \mathbb{R}^{n\times p}$ denote p principal components and m_j and λ_j be the mean and variance of the j^{th} principal component, respectively. To detect outliers, we consider the following detection model, which classify the i^{th} sample as an outlier if

$$\sum_{\{j:\lambda_j \geq \alpha\}} \frac{(Y_{ij} - m_j)^2}{\lambda_j} > c. \tag{2.27}$$

The metric appearing on the left-hand side of (2.27) represents the squared Euclidean distance to the origin in the standardized principal space consisting of principal components whose variance is greater than or equal to α. Therefore, our model can be understood as drawing a circular boundary (as illustrated in Figures 2.5 and 2.7) on this reduced standardized principal space. Since sample labels are unknown at the stage of building a model in the outlier detection setting, it is unclear how to choose an appropriate c. So, we compute precision and recall with varying c and evaluate the performance of each model using AUC under the precision-recall curve. We compare AUC of the Algorithm 1 based models to that of the KPCA-L1, L1-KPCA, R1-KPCA, and L2-KPCA based models as well as that of the two popular outlier detection models, Local Outlier Factor (LOF) [**11**] and Isolation Forest (iForest) [**49**].

Since principal components having small sample variance provide minor information, we only consider principal components whose sample variance is greater than or equal to some threshold value α. We set α be to the largest $\bar{\alpha}$ such that

$$0.8 \times \sum_{j=1}^{d} \lambda_j \leq \sum_{\{j:\lambda_j \geq \bar{\alpha}\}} \lambda_j$$

holds where d is the number of features. For the choice of the kernel function, we consider both the linear kernel and the Gaussian kernel with the width parameter σ of the Gaussian kernel to be equal to d. On the other hand, we set the number of nearest neighbors to 10 in LOF, and the number of trees, the size of subsample, and the number of rounds to 100, 256, and 10, respectively in iForest since these parameter values are commonly used.

Table 2.2. AUC of the outlier detection models

Datasets	AUC											
	Linear					Gaussian					[**11**]	[**49**]
	Alg 1	[**43**]	[**77**]	[**19**]	L2	Alg 1	[**43**]	[**77**]	[**19**]	L2		
WBC	0.52	0.53	0.53	0.47	0.48	**0.55**	**0.55**	**0.56**	0.47	0.49	0.35	**0.55**
Ionosphere	0.66	**0.73**	0.68	0.70	**0.71**	**0.71**	0.67	0.68	0.68	**0.71**	0.70	**0.71**
Breastw	0.93	0.91	0.92	0.92	0.92	**0.94**	0.92	0.93	**0.95**	**0.94**	0.38	**0.95**
Cardio	**0.58**	0.56	**0.58**	0.44	0.51	**0.59**	0.52	0.56	0.49	0.44	0.19	0.51
Musk	**0.99**	**0.99**	**0.99**	0.96	0.94	**0.99**	**0.99**	**0.99**	0.92	0.94	0.09	0.76
MNIST	**0.40**	**0.40**	**0.40**	**0.40**	0.39	**0.40**	**0.40**	**0.40**	0.38	0.36	0.19	0.34

Table 2.2 displays the AUCs of the 12 different detection models. The numbers in bold present the highest AUC cases (there can be several similar top performances). If outliers are obvious, any kernel PCA based model works well as seen in the case of Breastw and Musk. However, if outliers are unclear, the Algorithm 1 based detection models tend to outperform the other detection models. Especially, the Algorithm 1 based model with the Gaussian kernel consistently achieves top AUC values. Compared to the kernel PCA based models, LOF and iForest do not work well. LOF never achieves the top performance and iForest is not competitive for high-dimensional datasets such Must and MNIST although it yields the top AUC values for WBC and Breastw. As opposed to them, the Algorithm 1 based model with the Gaussian kernel consistently works well regardless of the size of the problem, demonstrating its effectiveness in outlier detection.

2.6.3. Runtime Comparison

Lastly, we compare the runtime of Algorithm 1 to that of KPCA-L1, L1-KPCA, R1-KPCA, and L2-KPCA. In order to obtain a runtime comparison, we run them on the six real-world datasets presented in Table 2.1 and measure the time taken to get all the principal components.

Table 2.3. Runtime comparison

Datasets	Runtime (minutes)									
	Linear					Gaussian				
	Alg 1	[**43**]	[**77**]	[**19**]	L2	Alg 1	[**43**]	[**77**]	[**19**]	L2
WBC	0.0	0.0	0.1	0.2	0.0	0.0	0.0	0.0	0.2	0.0
Ionosphere	0.0	0.0	0.1	0.2	0.0	0.0	0.0	0.1	0.2	0.0
Breastw	0.0	0.0	0.1	0.2	0.0	0.0	0.0	0.1	0.1	0.0
Cardio	0.0	1.9	12.3	7.6	0.1	0.0	1.8	15.3	6.2	0.1
Musk	0.5	69.6	999.1	973.5	0.3	0.5	12.9	1018.0	968.3	0.1
MNIST	1.8	558.6	9172.9	2285.3	4.4	2.0	1121.7	18231.7	2897.5	10.9

As shown in Table 2.3, the runtime largely varies across the algorithms. Among the L_1-norm based kernel PCA algorithms, Algorithm 1 has the smallest runtime for all datasets. Actually, it is much faster than the other two algorithms since it requires only one matrix-vector multiplication while the other algorithms entail either eigen-decomposition or solving a system of equations. R1-KPCA is also not as fast as Algorithm 1 since it involves QR-decomposition in each iteration to make loading vectors orthogonal. Among the robust kernel PCA algorithms, only Algorithm 1 is computationally comparable to L2-KPCA, making it the best choice for robust kernel PCA in a large-scale setting.

2.7. Final Remarks

In this work, we present a simple algorithm for L_1-norm kernel PCA and provide its convergence analysis. In order to develop it, we first reformulate L_1-norm kernel PCA

into a geometrically interpretable problem and derive a geometric interpretation behind it. Based on the geometric interpretation, we develop an algorithm to which the kernel trick is applicable. In the convergence analysis, we prove that the algorithm converges to a local optimal solution in a finite number of steps and the sequence of objective values converges at a linear rate.

The computational experiments demonstrate the robustness of the proposed algorithm in the presence of entry-wise perturbations and the runtime comparison shows that it takes much less time than the other robust kernel PCA algorithms. Also, its application to outlier detection outperforms all of the other benchmark algorithms. The model based on the proposed algorithm is not only better than that of the other kernel PCA based models but also outperforms LOF and iForest, especially when high-dimensional datasets are considered.

CHAPTER 3

Scale Invariant Power Iteration

3.1. Introduction

This chapter studies scale invariant problem (1.1) and its extended settings. In (1.1), the constraint $\partial\mathcal{B}_d$ in is not a convex set, therefore scale invariant problems are in general non-convex optimization problems. Nevertheless, it is possible to efficiently solve some cases of (1.1), for instance, the leading eigenvector problem [**26**], which is a motivating example of our study. Power iteration is an algorithm to find the leading eigenvector of a matrix A. In power iteration, the update rule $x_{k+1} \leftarrow Ax_k/\|Ax_k\|$ is repeatedly applied until some stopping criterion is satisfied. Since no hyperparameter is required, this update rule is practical yet attains global linear convergence with the rate of $|\lambda_2|/|\lambda_1|$ where $|\lambda_i|$ is the i^{th} largest absolute eigenvalue of A. This convergence result is analogous to that of gradient descent for convex optimization. Therefore, many variants including coordinate-wise [**46**], momentum [**79**], online [**9**, **24**], stochastic [**63**], stochastic variance-reduced (VR) [**72**, **73**], and stochastic VR momentum [**40**, **79**] power iterations have been developed, drawing a parallel literature to gradient descent for convex optimization.

Power iteration can be considered as a special case of the Frank-Wolfe algorithm (also called the conditional gradient method) with the step size of one. In this respect, an iterative algorithm called *generalized power method* (GPM), which computes the gradient at a current iterate and obtains the next iterate by projecting the gradient to the

constraint set is introduced in [**35**]. This idea has been used in many applications such as sparse principal component analysis (PCA) [**35**, **55**], L_1-norm kernel PCA [**39**], phase synchronization [**54**], and the Burer-Monteiro factorization of semi-definite programs [**20**]. While the linear convergence property of power iteration has been extended to some of these applications, theoretical understanding of when and how such an algorithm enjoys the attractive convergence property of power iteration is limited. For example, only global sublinear convergence of GPM has been shown for a general convex f [**35**], which does not generalize the appealing linear convergence property of power iteration.

On the other hand, we can view a scale invariant problem as an optimization problem on the real projective plane and consider an equivalent optimization problem on the embedding space ($\mathbb{R}^d$). The resulting optimization problem is unconstrained in the embedding space but have a highly non-convex structure as the maximization of the Rayleigh quotient. To find an optimal solution of the reformulated problem, one can employ general algorithms for unconstrained non-convex optimization such as gradient and Newton methods with line search, and trust region method [**1**].

In this work, rather than working in the embedding space, we focus on a generalization of power iteration to solve scale invariant problems. Specifically, we derive an algorithm called *scale invariant power iteration* (SCI-PI) and show that scale invariant problems can be efficiently solved by SCI-PI with a generalized convergence guarantee of power iteration. Having a general form of power iteration, SCI-PI requires no parameters and is thus much simpler than general non-convex algorithms yet still attains local linear convergence. To justify that SCI-PI is an appropriate algorithm for scale invariant problems, we derive an eigenvector property stating that any stationary point x^* satisfying $\nabla f(x^*) = \lambda^* x^*$ for

some λ^* is an eigenvector of $\nabla^2 f(x^*)$. Due to the eigenvector property, scale invariant problems can be locally seen as the leading eigenvector problem and thus we can expect that SCI-PI, a general form of power iteration, would efficiently solve scale invariant problems near a local optimum x^*.

In order to derive SCI-PI, we rely on a novel reformulation. By swapping the objective function and the constraint, we obtain a geometrically interpretable dual problem with the goal of finding the closest point w to the origin from the constraint $f(w) = 1$. By mapping a primal iterate x_k to the dual space, taking a descent step in the dual space and mapping it back to the original space, we provide a geometric derivation of SCI-PI, which replaces Ax_k with $\nabla f(x_k)$ in power iteration. In the convergence analysis, we show that SCI-PI converges to a local maximum x^* at a linear rate when initialized close to it. The convergence rate is proportional to $\bar{\lambda}_2/\lambda^*$ where $\bar{\lambda}_2$ is the spectral norm of $\nabla^2 f(x^*)(I - x^*(x^*)^T)$ and λ^* is the Lagrange multiplier corresponding to x^*, generalizing the convergence rate of power iteration. Moreover, under some mild conditions, we provide an explicit expression of the initial condition $\|x_0 - x^*\|$ to ensure local convergence.

In the extended settings, we discuss three variants of scale invariant problems. In the first setting, f is replaced with a sum of scale invariant functions. This setting covers a Kurtosis-based ICA and can be solved by SCI-PI with similar convergence guarantees. We also consider a block version of scale invariant problems which covers the Burer-Monteiro factorization of semi-definite programs and KL-NMF. To solve block scale invariant problems, we present a block version of SCI-PI and show that it attains linear convergence in a two-block case. Lastly, we consider partially scale invariant problems which include general mixture problems such as GMM. To solve partially scale invariant problems, we

present an alternative algorithm based on SCI-PI and the gradient method, and prove its local linear convergence. In numerical experiments, we benchmark the proposed algorithms against state-of-the-art methods for KL-NMF, GMM and ICA. The experimental results show that our algorithms are computationally competitive and result in better solutions in several cases.

Our work has the following contributions.

(1) We introduce scale invariant problems which cover interesting examples in statistics and machine learning yet can be efficiently solved by SCI-PI due to the eigenvector property.

(2) We present a geometric derivation of SCI-PI using a dual reformulation and provide a convergence analysis for it. We show that SCI-PI converges to a local maximum x^* at a linear rate when initialized close to x^*, generalizing the attractive convergence property of power iteration. Moreover, we introduce three extended settings of scale invariant problems together with their convergence analyses.

(3) We report numerical experiments including a novel reformulation of KL-NMF to extended settings of scale invariant problems. The experimental results demonstrate that SCI-PI are not only computationally competitive to state-of-the-art methods but also often yield better solutions.

The paper is organized as follows. In Section 3.2, we define scale invariance and present interesting properties of scale invariant problems including an eigenvector property and a dual formulation. We then provide a geometric derivation of SCI-PI and a convergence analysis in Section 3.3. The extended settings are discussed in Section 3.4 and we report the numerical experiments in Section 3.5.

3.2. Scale Invariant Problem

Before presenting properties of scale invariant problems, we first define scale invariant functions.

Definition 3.2.1. *We say that a function $f : \mathbb{R}^d \to \mathbb{R}$ is multiplicatively scale invariant if it satisfies*

$$f(cx) = u(c)f(x) \tag{3.1}$$

for some even function $u : \mathbb{R} \to \mathbb{R}^+$ with $u(0) = 0$. Also, we say that $f : \mathbb{R}^d \setminus \{0\} \to \mathbb{R}$ is additively scale invariant if it satisfies

$$f(cx) = f(x) + v(c) \tag{3.2}$$

for some even function $v : \mathbb{R} \setminus \{0\} \to \mathbb{R}$ with $v(1) = 0$.

The following proposition characterizes the exact form of u and v for continuous f.

Proposition 3.2.2. *If a continuous function $f \neq 0$ satisfies* (3.1) *with a multiplicative factor u, then we have*

$$u(c) = |c|^p \tag{3.3}$$

for some $p > 0$. Also, if a continuous function f satisfies (3.2) *with an additive factor v, then we have*

$$v(c) = \log_a |c| \tag{3.4}$$

for some a such that $0 < a$ and $a \neq 1$.

Proof. We first consider the multiplicative scale invariant case. Let x be a point such that $f(x) \neq 0$. Then, we have

$$f(rsx) = u(rs)f(x) = u(r)u(s)f(x),$$

which results in

$$u(rs) = u(r)u(s)$$

for all $r, s \in \mathbb{R}$. Let $g(r) = \ln(u(e^r))$. Then, we have

$$g(r+s) = \ln(u(e^{r+s})) = \ln(u(e^r e^s)) = \ln(u(e^r)) + \ln(u(e^s)) = g(r) + g(s),$$

which implies that g satisfies the first Cauchy functional equation. Since f is continuous, so is u and thus g. Therefore, by [**70**, pp. 81-82], we have

$$g(r) = rg(1) \tag{3.5}$$

for all $r \geq 0$. From the definition of g and (3.5), we have

$$u(e^r) = e^{g(r)} = (e^r)^{g(1)}. \tag{3.6}$$

Representing $r > 0$ as $r = e^{\ln(r)}$ and using (3.6), we obtain

$$u(r) = u\left(e^{\ln(r)}\right) = r^{g(1)} = r^{\ln(u(e))} = r^p.$$

Since $f(x) \neq 0$, if $p = \ln(u(e)) < 0$, then we have

$$\lim_{r\to 0_+} f(rx) = \lim_{r\to 0_+} u(r)f(x) = f(x) \cdot \lim_{r\to 0_+} r^p = f(x) \cdot \infty \neq f(0) < \infty,$$

contradicting the fact that f is continuous at 0. Also, if $p = 0$, then we get $u(r) = 1$, which contradicts $u(0) = 0$. Therefore, we must have $p > 0$. From u being an even function, we finally have

$$u(r) = |r|^p$$

for $r \in \mathbb{R}$.

Now, consider the additive scale invariant case. For any $x \in \mathrm{dom}(f)$, we have

$$f(rsx) = f(x) + v(rs) = f(x) + v(r) + v(s),$$

which results in

$$v(rs) = v(r) + v(s)$$

for all $r, s \in \mathbb{R}$. Let $g(r) = v(e^r)$. Then, we have

$$g(r + s) = v(e^{r+s}) = v(e^r e^s) = v(e^r) + v(e^s) = g(r) + g(s).$$

Since g is continuous and satisfies the second Cauchy functional equation, by [**70**, pp. 83-84], we have

$$g(r) = rg(1)$$

for all $r \geq 0$. For $r > 0$, letting $r = e^{\ln(r)}$, we have

$$v(r) = v(e^{\ln(r)}) = g(\ln(r)) = g(1)\ln(r) = v(e)\ln(r) = \log_a(r)$$

where $a = e^{\frac{1}{v(e)}}$. Note that a satisfies $0 < a$ and $a \neq 1$. From the fact that v is an even function, we finally have

$$v(r) = \log_a|r|$$

for $r \in \mathbb{R} \setminus \{0\}$. □

Using the explicit forms of u and v in Proposition 3.2.2, we establish derivative-based properties of scale invariant functions below.

Proposition 3.2.3. *Suppose that f is twice differentiable. If f satisfies* (3.1) *with a multiplicative factor $u(c) = |c|^p$, we have*

$$c\nabla f(cx) = |c|^p \nabla f(x), \quad \nabla f(x)^T x = pf(x), \quad \nabla^2 f(x)x = (p-1)\nabla f(x). \tag{3.7}$$

Also, if f satisfies (3.2) *with an additive factor $v(c) = \log_a|c|$, we have*

$$c\nabla f(cx) = \nabla f(x), \quad \nabla f(x)^T x = \log^{-1}(a), \quad \nabla^2 f(x)x = -\nabla f(x). \tag{3.8}$$

Proof. Without loss of generality, we can represent a scale-invariant function f as

$$f(cx) = u(c)f(x) + v(c) \tag{3.9}$$

since we can restore a multiplicatively or additively scale-invariant function by setting $v(c) = 0$ or $u(c) = 1$, respectively. By differentiating (3.9) with respect to x, we have

$$\nabla f(cx) = \frac{u(c)}{c} \nabla f(x).$$

On the other hand, by differentiating (3.9) with respect to c, we have

$$\nabla f(cx)^T x = u'(c) f(x) + v'(c). \tag{3.10}$$

By differentiating (3.10) with respect to x, we obtain

$$c\nabla^2 f(cx)x + \nabla f(cx) = u'(c)\nabla f(x). \tag{3.11}$$

Plugging $c = 1$ into (3.10) and (3.11) completes the proof. □

Proposition 3.2.3 states that a scale invariant function satisfies $\nabla^2 f(x) = k\nabla f(x)$ holds for some k. This relation is interesting since using the first-order optimality conditions, we can derive an eigenvector property as follows.

Proposition 3.2.4. *Suppose that f is twice differentiable and let (λ^*, x^*) be a stationary point of* (1.1) *such that*

$$\nabla f(x^*) = \lambda^* x^*.$$

If f satisfies (3.1) *with $u(c) = |c|^p$, then we have*

$$\nabla^2 f(x^*)x^* = (p-1)\lambda^* x^*.$$

Also, if f satisfies (3.2) *with $v(c) = \log_a |c|$, then we have*

$$\nabla^2 f(x^*)x^* = -\lambda^* x^*.$$

In both cases, x^* is an eigenvector of $\nabla^2 f(x^*)$. Moreover, if λ^* is greater than the largest eigenvalue of $\nabla^2 f(x^*)(I - x^*(x^*)^T)$, then x^* is a local maximum to (1.1).

Proof. Consider the Lagrangian function

$$L(x, \lambda) = f(x) + \frac{\lambda}{2}\left(1 - \|x\|^2\right)$$

and a stationary point (λ^*, x^*) satisfying

$$\nabla f(x^*) = \lambda^* x^*, \quad \|x^*\| = 1.$$

If f is multiplicative scale invariant with the degree of p, by Proposition 3.2.3, we have

$$\nabla^2 f(x^*)x^* = (p-1)\nabla f(x^*) = (p-1)\lambda^* x^*.$$

Also, by Proposition 3.2.3, if f is additive scale invariant f, we have

$$\nabla^2 f(x^*)x^* = -\nabla f(x^*) = -\lambda^* x^*.$$

Therefore, in both cases, a stationary point x^* is an eigenvector of $\nabla^2 f(x^*)$.

Suppose that λ^* is greater than the largest eigenvalue of $\nabla^2 f(x^*)(I - x^*(x^*)^T)$. For any d satisfying $d^T x^* = 0$, we have

$$d^T \nabla^2_{xx} L(x^*, \lambda^*)d = d^T \nabla^2 f(x^*)(I - x^*(x^*)^T)d - \lambda^* \|d\|^2 < 0.$$

Since the second-order sufficient condition is satisfied, x^* is a local maximum. $\square$

Proposition 3.2.4 states that a stationary point x^* is an eigenvector of $\nabla^2 f(x^*)$. Note that the Lagrange multiplier λ^* is not necessarily an eigenvalue corresponding to x^*. The eigenvalue corresponding to x^* is $(p-1)\lambda^*$ if f is multiplicatively scale invariant or $-\lambda^*$ if f is additively scale invariant. The sufficient condition for local optimality requires that the Lagrange multiplier λ^* rather than the eigenvalue corresponding to x^* is greater than the largest eigenvalue of $\nabla^2 f(x^*)(I - x^*(x^*)^T)$. Due to this eigenvector property, scale invariant problems can be considered as a generalization of the leading eigenvector problem. Next, we introduce a dual formulation of scale invariant problems.

Proposition 3.2.5. *Suppose that a continuous function f is either multiplicatively scale invariant such that $f(x^*) > 0$ or additively scale invariant with an additive factor $u(c) = \log_a |c|$ with $a > 1$. Then, solving* (1.1) *is equivalent to solving the following optimization problem*

$$\min_{w} \quad \|w\| \quad \text{subject to} \quad f(w) = 1. \tag{3.12}$$

In other words, if x^* is an optimal solution to (1.1)*, then $w^* = x^*/f(x^*)^{1/p}$ (multiplicative) or $w^* = a^{1-f(x^*)}x^*$ (additive) is an optimal solution to* (3.12)*. Conversely, if w^* is an optimal solution to* (3.12)*, $x^* = w^*/\|w^*\|$ is an optimal solution to* (1.1)*.*

Proof. First, we consider the case where an objective function f is multiplicative scale invariant with a multiplicative factor $u(c) = |c|^p$ where $p > 0$. Let w^* be an optimal solution to (3.12). From that $f(w^*) = 1$, we have $w^* \neq 0$, which leads to $\|w^*\| > 0$ and

$f\left(w^*/\|w^*\|\right) = 1/\|w^*\|^p > 0$. Suppose an optimal solution to (1.1) is y with

$$f(y) > f\left(w^*/\|w^*\|\right) > 0. \tag{3.13}$$

Let $\hat{y} = y/f(y)^{1/p}$. Then, we have $f(\hat{y}) = 1$ and $y = \hat{y}/\|\hat{y}\|$. Using $f(\hat{y}) = f(w^*) = 1$, we have

$$f(y) = f\left(\hat{y}/\|\hat{y}\|\right) = 1/\|\hat{y}\|^{1/p}, \quad f\left(w^*/\|w^*\|\right) = 1/\|w^*\|^{1/p}. \tag{3.14}$$

From (3.13) and (3.14), we obtain $\|\hat{y}\| < \|w^*\|$, which contradicts that w^* is an optimal solution to (3.12).

On the other hand, let x^* be an optimal solution to (1.1) with $f(x^*) > 0$. Suppose that an optimal solution to (3.12) is z with

$$\|z\| < \|x^*\|/f(x^*)^{1/p}. \tag{3.15}$$

Let $\hat{z} = z/\|z\|$. Then, we have $\|\hat{z}\| = 1$ and $z = \hat{z}/f(\hat{z})^{1/p}$. From that $\|\hat{z}\| = \|x^*\| = 1$, we have

$$\|z\| = \|\hat{z}\|/f(\hat{z})^{1/p} = 1/f(\hat{z})^{1/p}, \quad \|x^*\|/f(x^*)^{1/p} = 1/f(x^*)^{1/p}. \tag{3.16}$$

From (3.15) and (3.16), we have $f(x^*) < f(\hat{z})$ since $p > 0$, which contradicts the assumption that x^* is an optimal solution to (1.1).

Next, let f be an additively scale invariant function with an additive factor $v(c) = \log_a|c|$ with $a > 1$. In the same way as above, let w^* be an optimal solution to (3.12) and suppose

that an optimal solution of (1.1) is y with

$$f(y) > f\left(w^*/\|w^*\|\right). \tag{3.17}$$

Let $\hat{y} = a^{1-f(y)}y$. Then, we have $f(\hat{y}) = 1$ an $y = \hat{y}/\|\hat{y}\|$. Since $f(\hat{y}) = f(w^*) = 1$, we have

$$f(y) = f(\hat{y}) - \log_a\|\hat{y}\| = 1 - \log_a\|\hat{y}\|, \quad f\left(w^*/\|w^*\|\right) = 1 - \log_a\|w^*\|. \tag{3.18}$$

From (3.17) and (3.18), we have $\|\hat{y}\| < \|w^*\|$ due to $a > 1$, contradicting the fact that w^* is an optimal solution to (3.12).

Conversely, let x^* be an optimal solution to (1.1) and suppose that an optimal solution to (3.12) is z with

$$\|z\| < \|a^{1-f(x^*)}x^*\|. \tag{3.19}$$

Let $\hat{z} = z/\|z\|$. Then, we have $\|\hat{z}\| = 1$ and $z = a^{1-f(\hat{z})}\hat{z}$. Using $\|\hat{z}\| = \|x^*\| = 1$, we have

$$\|z\| = a^{1-f(\hat{z})}, \quad \|a^{1-f(x^*)}x^*\| = a^{1-f(x^*)}. \tag{3.20}$$

From (3.19) and (3.20), we have $f(x^*) < f(\hat{z})$ due to $a > 1$, contradicting the assumption that x^* is an optimal solution to (1.1). □

Note that a dual reformulation for a multiplicatively scale invariant f with $f(x^*) < 0$ or an additively scale invariant f with $0 < a < 1$ can be obtained by replacing $f(w) = 1$ with $f(w) = -1$ in (3.12). The dual formulation (3.12) has a nice geometric interpretation that an optimal solution w^* is the closest point to the origin from the set $\{w : f(w) = 1\}$. We use this understanding to derive SCI-PI in Section 3.3.

Lastly, we introduce two well-known examples of scale invariant problems in machine learning and statistics.

Example 3.2.6 (L_p-norm Kernel PCA)**.** *Given data vectors $a_i \in \mathbb{R}^d$ and a mapping $\Phi : \mathbb{R}^d \to F$, L_p-norm PCA considers*

$$\max_x \quad \frac{1}{n} \sum_{i=1}^n |\Phi(a_i)^T x|^p \quad \text{subject to} \quad x \in \partial\mathcal{B}_d \tag{3.21}$$

where the objective function satisfies property (3.1) *with $u(c) = |c|^p$.*

Example 3.2.7 (Estimation of Mixture Proportions)**.** *Given a design matrix $L \in \mathbb{R}^{n\times d}$ satisfying $L_{jk} \geq 0$, the problem of estimating mixture proportions seeks to find a vector π of mixture proportions on the probability simplex $\mathcal{S}^d = \{\pi : \sum_{k=1}^d \pi_k = 1, \pi \geq 0\}$ that maximizes the log-likelihood $\sum_{j=1}^n \log\left(\sum_{k=1}^d L_{jk}\pi_k\right)$. By reparametrizing π_k by x_k^2, we obtain an equivalent optimization problem*

$$\max_x \quad \frac{1}{n} \sum_{j=1}^n \log\left(\sum_{k=1}^d L_{jk} x_k^2\right) \quad \text{subject to} \quad x \in \partial\mathcal{B}_d, \tag{3.22}$$

which now satisfies property (3.2) *with $v(c) = 2\log|c|$.*

The reformulation idea in Example 3.2.7 implies that any simplex-constrained problem with scale invariant f can be reformulated to a scale invariant problem.

3.3. Scale Invariant Power Iteration

In this section, we provide a geometric derivation of SCI-PI to find a local optimal solution of (1.1). The algorithm is developed using the geometric interpretation of the dual formulation (3.12) as illustrated in Figure 3.1. Starting with an iterate $x_k \in \partial\mathcal{B}$,

we obtain a dual iterate w_k by projecting x_k to the constraint $f(w) = 1$. Given w_k, we identify the hyperplane h_k which the current iterate w_k lies on and is tangent to $f(w) = 1$. After identifying the equation of h_k, we find the closest point z_k to the origin from h_k and obtain a new dual iterate w_{k+1} by projecting z_k to the constraint $f(w) = 1$. Finally, we obtain a new primal iterate x_{k+1} by mapping w_{k+1} back to the set $\partial\mathcal{B}_d$.

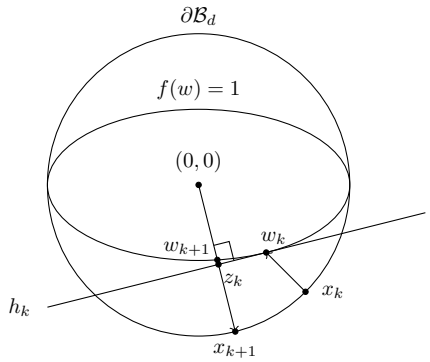

Figure 3.1. Geometric derivation of SCI-PI

Now, we develop an algorithm based on the above idea. For derivation of the algorithm, we assume that an objective function f is continuous and satisfies either (3.1) with $u(c) = |c|^p$ where $p > 0$ and $f(x) > 0$ for all $x \in \partial\mathcal{B}$ or (3.2) with $v(c) = \log_a|c|$ where $1 < a$. Under these conditions, a scalar mapping from x_k to w_k can be well defined as $w_k = x_k/f(x_k)^{1/p}$ or $w_k = a^{1-f(x_k)}x_k$, respectively. Let $w_k = c_k x_k$. Since w_k is on the constraint $f(w) = 1$, the tangent vector of the hyperplane h_k is $\nabla f(w_k)$. Therefore, we can write down the equation of the hyperplane h_k as $\{w : \nabla f(w_k)^T(w - w_k) = 0\}$. Note that z_k is a scalar multiple of $\nabla f(w_k)$ where the scalar can be determined from the requirement that z_k is on h_k. Since w_{k+1} is the projection of z_k, it must be a scalar multiple of the tangent vector $y_k = \nabla f(w_k)$. Therefore, we can write w_{k+1} as $w_{k+1} = d_k y_k$. Finally, by

projecting w_{k+1} to $\partial\mathcal{B}$, we obtain

$$x_{k+1} = \frac{w_{k+1}}{\|w_{k+1}\|} = \frac{d_k y_k}{\|d_k y_k\|} = \frac{y_k}{\|y_k\|} = \frac{\nabla f(w_k)}{\|\nabla f(w_k)\|} = \frac{\nabla f(c_k x_k)}{\|\nabla f(c_k x_k)\|} = \frac{\nabla f(x_k)}{\|\nabla f(x_k)\|}$$

where the last equality follows from Proposition 3.2.3. Summarizing all the above, we obtain SCI-PI presented in Algorithm 2.

Algorithm 2 SCI-PI

Input: initial point x_0

for $k = 0, 1, \ldots, T-1$ **do**

$x_{k+1} \leftarrow \frac{\nabla f(x_k)}{\|\nabla f(x_k)\|}$

end for

Output: x_T

Next, we provide a convergence analysis of SCI-PI. Global sublinear convergence of SCI-PI for convex f has been addressed in [**35**]. We additionally show that SCI-PI yields an ascent step even for quasi-convex f.

Proposition 3.3.1. *If f is quasi-convex and differentiable, a sequence of iterates $\{x_k\}_{k=0,1,\cdots}$ generated by SCI-PI satisfies $f(x_{k+1}) \geq f(x_k)$ for $k = 0, 1, \cdots$.*

Proof. If $f(x_{k+1}) < f(x_k)$, by the first-order condition of differentiable quasi-convex functions, we have

(3.23)

$$\nabla f(x_k)^T (x_{k+1} - x_k) = \nabla f(x_k)^T \left(\frac{\nabla f(x_k)}{\|\nabla f(x_k)\|} - x_k \right) = \|\nabla f(x_k)\| - \nabla f(x_k)^T x_k \leq 0.$$

However, since $f(x_{k+1}) \neq f(x_k)$, $\nabla f(x_k)$ is not a scalar multiple of x_k, leading to

$$\|\nabla f(x_k)\| - \nabla f(x_k)^T x_k > 0.$$

This contradicts (3.23). Therefore, we should have $f(x_{k+1}) \geq f(x_k)$. □

If f is quasi-convex, the set $\{w : f(w) \leq 1\}$ is convex, therefore, from Figure 3.1, we can expect that SCI-PI would yield an ascent step. If f is not quasi-convex, $\{f(x_k)\}_{k=0,1,\cdots}$ is not necessarily increasing, making it hard to analyze global convergence. Assuming that an initial point x_0 is close to a local maximum x^*, we study local convergence of SCI-PI as follows.

Theorem 3.3.2. *Let f be a scale invariant, twice continuously differentiable function on an open set containing $\partial\mathcal{B}_d$ and let x^* be a local maximum satisfying $\nabla f(x^*) = \lambda^* x^*$ and $\lambda^* > \bar{\lambda}_2 = \max_{2\leq i\leq d}|\lambda_i|$ where (λ_i, v_i) is an eigen-pair of $\nabla^2 f(x^*)$ with $x^* = v_1$. Then, there exists some $\delta > 0$ such that under the initial condition $1 - x_0^T x^* < \delta$, the sequence of iterates $\{x_k\}_{k=0,1,\cdots}$ generated by SCI-PI satisfies*

$$1 - (x_k^T x^*)^2 \leq \prod_{t=0}^{k-1} \left(\frac{\bar{\lambda}_2}{\lambda^*} + \gamma_t\right)^2 \left(1 - (x_0^T x^*)^2\right),$$

where

$$\frac{\bar{\lambda}_2}{\lambda^*} + \gamma_t < 1 \text{ for all } t \geq 0 \text{ and } \lim_{k\to\infty} \gamma_k = 0.$$

Moreover, if $\nabla_i f = \partial f/\partial x_i$ *has a continuous Hessian* H_i *on an open set containing* $\mathcal{B}_{d,\infty} \triangleq \{x \in \mathbb{R}^d : \|x\|_\infty \leq 1\}$*, we can explicitly write* δ *as*

$$\delta(\lambda^*, \bar{\lambda}_1, \bar{\lambda}_2, M) = \min\left\{\left(\frac{\lambda^*}{\bar{\lambda}_1 + M}\right)^2, \left(\frac{\lambda^* - \bar{\lambda}_2}{\bar{\lambda}_1 + 2M}\right)^2, 1\right\}$$

where

$$\bar{\lambda}_1 = |\lambda_1|, \quad M = \max_{x \in \partial\mathcal{B}_d,\, y \in \mathcal{B}_{d,\infty}} \sqrt{\textstyle\sum_{i=1}^d (x^T G_i(y) x)^2}, \quad G_i(y) = \textstyle\sum_{j=1}^d v_{i,j} H_j(y).$$

Proof. Since $\nabla^2 f(x^*)$ is real and symmetric, without loss of generality, we assume that $\{v_1, \ldots, v_d\}$ form an orthogonal basis in $\mathbb{R}^d$.

Since f is twice continuously differentiable on an open set containing $\partial\mathcal{B}_d$, for $x \in \partial\mathcal{B}_d$, using the Taylor expansion of $\nabla f(x)^T v_i$ at x^*, we have

$$\nabla f(x)^T v_i = \nabla f(x^*)^T v_i + (x - x^*)^T \nabla^2 f(x^*) v_i + R_i(x) \tag{3.24}$$

where

$$R_i(x) = o(\|x - x^*\|). \tag{3.25}$$

From $\nabla f(x^*) = \lambda^* x^*$ and $x^* = v_1$, we have

$$\begin{aligned} \nabla f(x)^T v_1 &= \nabla f(x^*)^T x^* + (x - x^*)^T \nabla^2 f(x^*) x^* + R_1(x) \\ &= \lambda^* - \lambda_1(1 - x^T x^*) + R_1(x) \\ &= \lambda^* + \alpha(x) \end{aligned} \tag{3.26}$$

where

$$\alpha(x) = -\lambda_1(1 - x^T x^*) + R_1(x) = o(\|x - x^*\|)$$

due to $R_1(x) = o(\|x - x^*\|)$ and $1 - x^T x^* = o(\|x - x^*\|)$.

On the other hand, for $2 \le i \le d$, due to $\nabla f(x^*) = \lambda^* x^*$, we have

$$\nabla f(x^*)^T v_i = \lambda^* (x^*)^T v_i = 0. \tag{3.27}$$

From (3.24), this results in

$$\nabla f(x)^T v_i = \lambda_i x^T v_i + R_i(x). \tag{3.28}$$

Let $\bar{R}_2(x) = \max_{2 \le i \le d} |R_i(x)|$. Note that $\bar{R}_2(x) = o(\|x - x^*\|)$. By (3.28), we obtain

$$\begin{aligned} \sum_{i=2}^{d} \left(\nabla f(x)^T v_i\right)^2 &= \sum_{i=2}^{d} \left[\lambda_i^2 (x^T v_i)^2 + 2\lambda_i (x^T v_i) R_i(x) + (R_i(x))^2\right] \\ &\le \bar{\lambda}_2^2 \sum_{i=2}^{d} (x^T v_i)^2 + 2\bar{\lambda}_2 \bar{R}_2(x) \sum_{i=2}^{d} |x^T v_i| + d\left(\bar{R}_2(x)\right)^2. \end{aligned} \tag{3.29}$$

From $x \in \partial\mathcal{B}_d$, $x^* = v_1$, and the fact that $\{v_1, \ldots, v_d\}$ forms an orthogonal basis in $\mathbb{R}^d$, we have

$$\sum_{i=2}^{d} (x^T v_i)^2 = 1 - (x^T v_1)^2 = 1 - (x^T x^*)^2 \le 2(1 - x^T x^*) = \|x - x^*\|^2.$$

Also, by the Cauchy Schwartz inequality, we have

$$\sum_{i=2}^{d} |x^T v_i| \le \sqrt{d} \sqrt{\sum_{i=2}^{d} (x^T v_i)^2} \le \sqrt{d} \|x - x^*\|.$$

Therefore, we obtain from (3.29) that

$$\begin{aligned}\sum_{i=2}^{d}\left(\nabla f(x)^T v_i\right)^2 &\le \bar{\lambda}_2^2\|x-x^*\|^2 + 2\bar{\lambda}_2\bar{R}_2(x)\sqrt{d}\|x-x^*\| + d\left(\bar{R}_2(x)\right)^2 \\ &= \left(\bar{\lambda}_2\|x-x^*\| + \beta(x)\right)^2\end{aligned} \tag{3.30}$$

where

$$\beta(x) = \sqrt{d}\bar{R}_2(x) = o(\|x-x^*\|).$$

By (3.26), (3.30), and Lemma A.1.1, we obtain the first part of the desired result.

Next, we consider the case where $\nabla_i f$ has a continuous Hessian H_i. From $\nabla_i f(x)$ being twice continuously differentiable in $\mathcal{B}_\infty$, we have

$$\nabla_i f(x_k) = \nabla_i f(x^*) + \nabla\nabla_i f(x^*)(x_k - x^*) + \frac{1}{2}\left(x_k - x^*\right)^T H_i(\hat{x}_k^i)\left(x_k - x^*\right) \tag{3.31}$$

where

$$\hat{x}_k^i \in \mathcal{N}(x_k, x^*) \triangleq \left\{x : x_s = t_s x_s^* + (1-t_s)x_{k,s},\, 0 \le t_s \le 1,\, s = 1,\ldots,d\right\}.$$

In the above, x_s^* and $x_{k,s}$ denote the s^{th} coordinates of x^* and x_k, respectively.

For each $1 \le i \le d$, we have

$$\frac{1}{2}\sum_{j=1}^{d} v_{i,j}\left(x_k - x^*\right)^T H_j(\hat{x}_k^j)\left(x_k - x^*\right) = \frac{1}{2}(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*).$$

From

$$\left|(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*)\right| = \|x_k - x^*\|^2 \left| \left[\frac{x_k - x^*}{\|x_k - x^*\|}\right]^T G_i(\hat{x}_k^j)\left[\frac{x_k - x^*}{\|x_k - x^*\|}\right]\right| \tag{3.32}$$

and

$$\max_{x\in\partial\mathcal{B}_d}|x^TG_i(\hat{x}_k^j)x| \le \max_{x\in\partial\mathcal{B}_d,\, y\in\mathcal{B}_\infty}|x^TG_i(y)x| \le \max_{x\in\partial\mathcal{B}_d,\, y\in\mathcal{B}_\infty}\sqrt{\sum_{i=1}^d\left(x^TG_i(y)x\right)^2} = M,$$

we have

$$\left|(x_k-x^*)^TG_i(\hat{x}_k^j)(x_k-x^*)\right| \le M\|x_k-x^*\|^2,$$

leading to

$$\frac{1}{2}\left|\sum_{j=1}^d v_{i,j}\,(x_k-x^*)^T\,H_j(\hat{x}_k^j)\,(x_k-x^*)\right| \le \frac{1}{2}M\|x_k-x^*\|^2. \tag{3.33}$$

From (3.31), (3.33) and that $x^* = v_1$, we have

$$\nabla f(x_k)^Tv_1 \ge \nabla f(x^*)^Tx^* + (x_k-x^*)^T\nabla^2 f(x^*)x^* - \frac{M}{2}\|x_k-x^*\|^2,$$

resulting in

$$\nabla f(x_k)^Tv_1 \ge \lambda^* - (M+|\lambda_1|)(1-x_k^Tx^*). \tag{3.34}$$

For $2 \le i \le d$, we have

$$\begin{aligned}\nabla f(x_k)^Tv_i &= \nabla f(x^*)^Tv_i + (x_k-x^*)^T\nabla^2 f(x^*)v_i + \frac{1}{2}(x_k-x^*)^TG_i(\hat{x}_k^j)(x_k-x^*)\\ &= \lambda_i x_k^Tv_i + \frac{1}{2}(x_k-x^*)^TG_i(\hat{x}_k^j)(x_k-x^*). \qquad (3.35)\end{aligned}$$

Using (3.32) and

$$\max_{x\in\partial\mathcal{B}_d}\sum_{i=2}^{d}(x^TG_i(\hat{x}_k^j)x)^2 \le \max_{x\in\partial\mathcal{B}_d,\,y\in\mathcal{B}_\infty}\sum_{i=2}^{d}(x^TG_i(y)x)^2 \le \max_{x\in\partial\mathcal{B}_d,\,y\in\mathcal{B}_\infty}\sum_{i=1}^{d}(x^TG_i(y)x)^2 \le M,$$

we have

$$\sum_{i=2}^{d}\left[(x_k-x^*)^TG_i(\hat{x}_k^j)(x_k-x^*)\right]^2 \le M^2\|x_k-x^*\|^4. \tag{3.36}$$

Using (3.35), (3.36) and the Cauchy-Schwartz inequality, we have

$$\begin{aligned}\sum_{i=2}^{d}(\nabla f(x_k)^Tv_i)^2 &\le \sum_{i=2}^{d}\left(|\lambda_i||x_k^Tv_i| + \frac{1}{2}(x_k-x^*)^TG_i(\hat{x}_k^j)(x_k-x^*)\right)^2\\ &\le \bar{\lambda}_2^2\sum_{i=2}^{d}(x_k^Tv_i)^2 + \bar{\lambda}_2M\|x_k-x^*\|^2\sqrt{\sum_{i=2}^{d}(x_k^Tv_i)^2} + \frac{M^2}{4}\|x_k-x^*\|^4\\ &= \left(\bar{\lambda}_2\sqrt{1-(x_k^Tx^*)^2} + \frac{M}{2}\|x_k-x^*\|^2\right)^2. \qquad (3.37)\end{aligned}$$

Using (3.34), (3.37), and Lemma A.1.2 with

$$A=\lambda^*,\ B=M+|\lambda_1|,\ C=0,\ D=\bar{\lambda}_2,\ E=0,\ F=M,$$

we obtain the desired result. □

Theorem 3.3.2 presents a local convergence result of SCI-PI with the rate being $\frac{\lambda^*}{\bar{\lambda}_2}$. This convergence rate generalizes that of power iteration, since it specializes to $\frac{\lambda_1}{\lambda_2}$ when it comes to the leading eigenvector problem. Note that Theorem 3.3.2 requires that a Lagrange multiplier λ^* corresponding to a local maximum x^* satisfies $\lambda^* > \bar{\lambda}_2 = \max_{2\le i\le d}|\lambda_i|$. This assumption is satisfied by all local maxima if f is convex, multiplicatively scale

invariant or concave, additively scale invariant. However, in general, not all local maxima satisfy this assumption since it is stronger than the local optimality condition stated as $\lambda^* > \max_{2\leq i\leq d} \lambda_i$. Nevertheless, by adding $\sigma\|x\|^2$ for some $\sigma > 0$ to the objective function f, we can always enforce $\lambda^* > \bar{\lambda}_2$. Conversely, by adding $\sigma\|x\|^2$ for some $\sigma < 0$, we may improve the convergence rate as in shifted power iteration.

3.4. Extended Settings

3.4.1. Sum of Scale Invariant Functions

Consider a sum of scale invariant functions having the form of $f(x) = \sum_{i=1}^m g_i(x) + \sum_{j=1}^n h_j(x)$ where g_i is a multiplicatively scale invariant function with $u(c) = |c|^{p_i}$ and h_j is an additively scale invariant function with $v(c) = \log_{a_j} |c|$. Note that this does not imply that f is scale invariant in general. Here is an example that involves a sum of scale invariant functions.

Example 3.4.1 (Kurtosis-based ICA)**.** *Given a pre-processed data matrix $W \in \mathbb{R}^{n\times d}$, Kurtosis-based ICA [**31**] solves*

$$\max_x \quad \frac{1}{n}\sum_{i=1}^n \left[(w_i^T x)^4 - 3\right]^2 \quad \text{subject to} \quad x \in \partial\mathcal{B}_d. \tag{3.38}$$

The objective function f is a sum of scale invariant functions.

By Proposition 3.2.3, the gradient of f has the form of

$$\nabla f(x) = \sum_{i=1}^m \nabla g_i(x) + \sum_{j=1}^n \nabla h_j(x) = F(x)x,$$

where

$$F(x) = \sum_{i=1}^{m} \left(\frac{1}{p_i - 1} \right) \nabla^2 g_i(x) - \sum_{j=1}^{n} \nabla^2 h_j(x).$$

Note that a stationary point x^* satisfying $\nabla f(x^*) = \lambda^* x^*$ is not necessarily an eigenvector of $\nabla^2 f(x^*)$. Instead, a stationary point x^* is an eigenvector of $F(x)$. We present a local convergence analysis of SCI-PI for a sum of scale invariant functions as follows.

Theorem 3.4.2. *Let f be a sum of scale invariant functions and twice continuously differentiable on an open set containing $\partial\mathcal{B}_d$ and let x^* be a local maximum satisfying $\nabla f(x^*) = \lambda^* x^*$ and $\lambda^* > \bar{\lambda}_2 = \|\nabla^2 f(x^*)(I - x^*(x^*)^2)\|$. Then, there exists some $\delta > 0$ such that under the initial condition $1 - x_0^T x^* < \delta$, the sequence of iterates $\{x_k\}_{k=0,1,\cdots}$ generated by SCI-PI satisfies*

$$1 - (x_k^T x^*)^2 \leq \prod_{t=0}^{k-1} \left(\frac{\bar{\lambda}_2}{\lambda^*} + \gamma_t \right)^2 \left(1 - (x_0^T x^*)^2 \right),$$

where

$$\frac{\bar{\lambda}_2}{\lambda^*} + \gamma_t < 1 \text{ for all } t \geq 0 \text{ and } \lim_{k \to \infty} \gamma_k = 0.$$

Moreover, if $\nabla_i f = \partial f / \partial x_i$ has a continuous Hessian H_i on an open set containing $\mathcal{B}_{d,\infty}$, we can explicitly write δ as

$$\delta(\lambda^*, \bar{\lambda}_1, \bar{\lambda}_2, M) = \min \left\{ \left(\frac{\lambda^*}{\bar{\lambda}_1 + M} \right)^2, \left(\frac{\lambda^* - \bar{\lambda}_2}{\bar{\lambda}_1 + \bar{\lambda}_2 + 2M} \right)^2, 1 \right\}$$

where

$$\bar{\lambda}_1 = \sqrt{2} \cdot \|\nabla^2 f(x^*) x^*\|,\ M = \max_{x \in \partial\mathcal{B}_d,\, y \in \mathcal{B}_{d,\infty}} \sqrt{\textstyle\sum_{i=1}^{d} (x^T G_i(y) x)^2},\ G_i(y) = \textstyle\sum_{j=1}^{d} v_{i,j} H_j(y).$$

Proof. Let $\{v_1, \ldots, v_d\}$ be a set of eigenvectors of $F(x^*)$ with $x^* = v_1$. Since $F(x^*)$ is real and symmetric, without loss of generality, we assume that $\{v_1, \ldots, v_d\}$ form an orthogonal basis in $\mathbb{R}^d$.

Since f is twice continuously differentiable on an open set containing $\partial\mathcal{B}_d$, for $x \in \partial\mathcal{B}_d$, using the Taylor expansion of $\nabla f(x)^T v_i$ at x^*, we have

$$\nabla f(x)^T v_i = \nabla f(x^*)^T v_i + (x - x^*)^T \nabla^2 f(x^*) v_i + R_i(x) \tag{3.39}$$

where $R_i(x) = o(\|x - x^*\|)$. Using (3.39) with $i = 1$ and $\nabla f(x^*) = \lambda^* x^*$, we obtain

$$\begin{aligned} \nabla f(x)^T v_1 &= \lambda^* (x^*)^T v_1 + (x - x^*)^T \nabla^2 f(x^*) v_1 + R_1(x) \\ &= \lambda^* + \alpha(x) \end{aligned} \tag{3.40}$$

where

$$\alpha(x) = (x - x^*)^T \nabla^2 f(x^*) v_1 + R_1(x) = o(\sqrt{\|x - x^*\|}).$$

Using (3.39) and $\nabla f(x^*) = \lambda^* x^*$ for $2 \le i \le d$, we have

$$\begin{aligned} \nabla f(x)^T v_i &= \lambda^* (x^*)^T v_i + (x - x^*)^T \nabla^2 f(x^*) v_i + R_i(x) \\ &= (x - x^*)^T \nabla^2 f(x^*) v_i + R_i(x), \end{aligned}$$

resulting in

$$\sum_{i=2}^{d} (\nabla f(x)^T v_i)^2 = \sum_{i=2}^{d} \left((x - x^*)^T \nabla^2 f(x^*) v_i + R_i(x) \right)^2. \tag{3.41}$$

Let $\bar{R}_2(x) = \max_{2 \le i \le d} |R_i(x)|$. Note that $\bar{R}_2(x) = o(\|x - x^*\|)$.

From $x^* = v_1$ and the fact that $\{v_1, \ldots, v_d\}$ forms an orthogonal basis in $\mathbb{R}^d$, we have

$$\begin{aligned}\sum_{i=2}^{d} \left((x-x^*)^T\nabla^2 f(x^*)v_i\right)^2 &= \|\nabla^2 f(x^*)(x-x^*)\|_2^2 - \left((x-x^*)^T\nabla^2 f(x^*)v_1\right)^2 \\ &= (x-x^*)^T\nabla^2 f(x^*)\left(I - x^*(x^*)^T\right)\nabla^2 f(x^*)(x-x^*) \\ &= (x-x^*)^T\nabla^2 f(x^*)\left(I - x^*(x^*)^T\right)^2\nabla^2 f(x^*)(x-x^*).\end{aligned}$$

Since

$$\begin{aligned}\|\nabla^2 f(x^*)\left(I - x^*(x^*)^T\right)^2\nabla^2 f(x^*)\| &= \|\left(I - x^*(x^*)^T\right)\nabla^2 f(x^*)\|^2 \\ &= \|\nabla^2 f(x^*)\left(I - x^*(x^*)^T\right)\|^2,\end{aligned}$$

we have

$$\sum_{i=2}^{d} \left((x-x^*)^T\nabla^2 f(x^*)v_i\right)^2 \leq \bar{\lambda}_2^2\|x-x^*\|^2. \tag{3.42}$$

Also, from (3.42) and the Cauchy-Schwartz inequality, we obtain

$$\sum_{i=2}^{d}(x-x^*)^T\nabla^2 f(x^*)v_i \leq \sum_{i=2}^{d}|(x-x^*)^T\nabla^2 f(x^*)v_i| \leq \bar{\lambda}_2\sqrt{d}\|x-x^*\|. \tag{3.43}$$

Using (3.42) and (3.43) for (3.41), we obtain

$$\sum_{i=2}^{d}(\nabla f(x)^T v_i)^2 \leq \bar{\lambda}_2^2\|x-x^*\|^2 + 2\bar{\lambda}_2\bar{R}_2(x)\sqrt{d}\|x-x^*\| + d(\bar{R}_2(x))^2,$$

resulting in

$$\sum_{i=2}^{d}(\nabla f(x)^T v_i)^2 \leq \left(\bar{\lambda}_2\|x-x^*\|^2 + \beta(x)\right)^2 \tag{3.44}$$

where $\beta(x) = \sqrt{d}\bar{R}_2(x) = o(\|x - x^*\|)$. By (3.40), (3.44), and Lemma A.1.1, we obtain the first part of the desired result.

Next, we assume that $\nabla_i f$ has a continuous Hessian H_i. By the Taylor theorem, we have

$$\nabla_i f(x_k) = \nabla_i f(x^*) + \nabla\nabla_i f(x^*)(x_k - x^*) + \frac{1}{2}(x_k - x^*)^T H_i(\hat{x}_k^i)(x_k - x^*) \tag{3.45}$$

for some $\hat{x}_k^i \in \mathcal{N}(x_k, x^*)$.

Taking the steps used to derive (3.33) and (3.36) in the proof of Theorem 3.3.2, we can derive the same inequalities

$$\frac{1}{2}\left|(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*)\right| \leq \frac{1}{2}M\|x_k - x^*\|^2 \tag{3.46}$$

and

$$\frac{1}{4}\sum_{i=2}^{d}\left[(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*)\right]^2 \leq \frac{M^2}{4}\|x_k - x^*\|^4. \tag{3.47}$$

Using (3.45), (3.47) and that $x^* = v_1$, we have

$$\nabla f(x_k)^T v_1 \geq \nabla f(x^*)^T x^* + (x_k - x^*)^T \nabla^2 f(x^*)x^* - \frac{M}{2}\|x_k - x^*\|^2$$

resulting in

$$\begin{aligned}\nabla f(x_k)^T v_1 &\geq \lambda^* - \|\nabla^2 f(x^*)x^*\|\sqrt{2(1 - x_k^T x^*)} - M(1 - x_k^T x^*) \\ &= \lambda^* - \bar{\lambda}_1\sqrt{(1 - x_k^T x^*)} - M(1 - x_k^T x^*)\end{aligned} \tag{3.48}$$

For $2 \leq i \leq d$, we have

$$\begin{aligned}\nabla f(x_k)^T v_i &\leq \nabla f(x^*)^T v_i + (x_k - x^*)^T \nabla^2 f(x^*) v_i + \frac{1}{2}(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*) \\ &= \lambda^*(x^*)^T v_i + (x_k - x^*)^T \nabla^2 f(x^*) v_i + \frac{1}{2}(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*) \\ &= (x_k - x^*)^T \nabla^2 f(x^*) v_i + \frac{1}{2}(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*). \qquad (3.49)\end{aligned}$$

From (3.49), (3.42), (3.46), (3.47) and the Cauchy-Shwartz inequality, we have

$$\begin{aligned}\sum_{i=2}^{d} (\nabla f(x_k)^T v_i)^2 &\leq \sum_{i=2}^{d} \left((x_k - x^*)^T \nabla^2 f(x^*) v_i + \frac{1}{2}(x_k - x^*)^T G_i(\hat{x}_k^j)(x_k - x^*) \right)^2 \\ &\leq \left(\bar{\lambda}_2 \|x_k - x^*\| + \frac{M}{2}\|x_k - x^*\|^2 \right)^2. \qquad (3.50)\end{aligned}$$

Using (3.48), (3.50), and Lemma A.1.2 with

$$A = \lambda^*,\ B = M,\ C = \bar{\lambda}_1,\ D = 0,\ E = \bar{\lambda}_2,\ F = M,$$

we obtain the desired result. □

Note that $\bar{\lambda}_1$ has the additional $\sqrt{2}$ factor which comes from the fact that x^* is not necessarily an eigenvector of $\nabla^2 f(x^*)$. Nonetheless, the asymptotic convergence rate in Theorem 3.4.2 provides a generalization of the convergence rate in Theorem 3.3.2.

3.4.2. Block Scale Invariant Problems

Next, consider a class of optimization problems having the form of

$$\max_{x,y} \quad f(x,y) \quad \text{subject to} \quad x \in \partial\mathcal{B}_{d_1},\ y \in \partial\mathcal{B}_{d_2}$$

where $f : \mathbb{R}^{d_1+d_2} \to \mathbb{R}$ is scale invariant in x for fixed y and vice versa. Some examples of block scale invariant problems are given next.

Example 3.4.3 (Semidefinite Programming (SDP) [**20**])**.** *Let $A, X \in \mathbb{R}^{n\times n}$. Given an SDP problem*

$$\max_X \quad \langle A, X\rangle \quad \text{subject to} \quad X_{ii} = 1,\ i \in \{1, 2, \cdots, n\},\ X \succeq 0,$$

*the Burer-Monteiro approach [**14**] yields the following block scale invariant problem*

$$\max_\sigma \quad \langle A, \sigma\sigma^T\rangle \quad \text{subject to} \quad \|\sigma_i\| = 1,\ i \in \{1, 2, \cdots, n\}.$$

Example 3.4.4 (Kullback-Leibler (KL) divergence NMF)**.** *The KL-NMF problem [**21, 45, 76**] is defined as*

(3.51)
$$\begin{aligned} \min_{W,H} \quad & D_{KL}(V\|WH) \triangleq \textstyle\sum_{i,j}\left[V_{ij}\log\frac{V_{ij}}{\sum_k W_{ik}H_{kj}} - V_{ij} + \sum_k W_{ik}H_{kj}\right] \\ \text{subject to} \quad & W_{ik} \geq 0,\ H_{kj} \geq 0,\ i \in \{1, \cdots, n\},\ j \in \{1, \cdots, m\},\ k \in \{1, \cdots, K\}. \end{aligned}$$

Many popular algorithms for the KL-NMF problem are based on alternate minimization of W and H. Given $W \geq 0$ and $j \in \{1, \cdots, m\}$, we consider a subproblem such that

$$\min_h \quad f_{KL}(h) = \textstyle\sum_i\left[v_i\log\frac{v_i}{\sum_k W_{ik}h_k} - v_i + \sum_k W_{ik}h_k\right] \quad \text{subject to} \quad h_k \geq 0 \tag{3.52}$$

where we let $v_i = V_{ij}$ and $h_k = H_{kj}$ as the objective is decomposed into m separate subproblems. Note that the KL-NMF problem in the form of (3.51) is not a block scale

invariant problem. However, using a novel reformulation, we show that the KL divergence NMF subproblem is indeed a scale invariant problem.

Lemma 3.4.5. *The KL-NMF subproblem* (3.52) *is equivalent to the following scale invariant problem*

$$\max_{\bar{h}} \quad -\textstyle\sum_i v_i \log \sum_k W_{ik}\bar{h}_k \quad \text{subject to} \quad \sum_k \bar{h}_k = 1, \quad \bar{h}_k \geq 0, \tag{3.53}$$

with the relationship $(\sum_i v_i)\bar{h}_k = (\sum_i W_{ik})h_k$.

Proof. Since a log-linear function is concave, (3.52) is a convex problem in h. Consider the Lagrangian of the original problem

$$\mathcal{L}(h, \lambda) = f_{KL}(h) - \textstyle\sum_k \lambda_k h_k \tag{3.54}$$

where $\lambda \geq 0$. By the first-order KKT conditions, we must have

$$\nabla_k f_{KL}(h^*) = \lambda_k^*, \quad \lambda_k^* h_k^* = 0, \quad \forall k = 1, \cdots, K \tag{3.55}$$

at an optimal solution (h^*, λ^*). Since (3.55) implies $\sum_k h_k^* \lambda_k^* = 0$, we have

$$\sum_k h_k^* \lambda_k^* = \sum_k h_k^* \nabla_k f_{KL}(h^*) = -\sum_{i,k} \frac{v_i W_{ik} h_k^*}{\sum_{k'} W_{ik'} h_{k'}^*} + \sum_{i,k} W_{ik} h_k^*,$$

resulting in

$$\textstyle\sum_i v_i = \sum_{i,k} W_{ik} h_k^*. \tag{3.56}$$

Next, we show that

$$\min_h \ f_{SCI}(h) = \textstyle\sum_i v_i \log \frac{v_i}{\sum_k W_{ik} h_k} \quad \text{subject to} \quad \sum_i v_i = \sum_{i,k} W_{ik} h_k, \ h_k \geq 0. \tag{3.57}$$

is equivalent to the original subproblem (5.68), due to the following:

(1) It always satisfies $f^*_{SCI} \geq f^*_{KL}$ since (3.57) has an additional constraint $\sum_i v_i = \sum_{i,k} W_{ik} h_k$ compared to (3.52).

(2) A solution h^* of (3.52) is a feasible point of (3.57) since we have shown that $\sum_i v_i = \sum_{i,k} W_{ik} h^*_k$. This implies $f^*_{KL} \geq f^*_{SCI}$.

Now, we can reparametrize h by $\bar{h}$ so that $\sum_i v_i = \sum_{i,k} W_{ik} h_k$ if and only if $\sum_k \bar{h}_k = 1$, which yields the relationship between two variables $\bar{h}_k = h_k \frac{\sum_i W_{ik}}{\sum_i v_i}$. Note that (3.53) has the optimization problem as Example 3.2.7 and thus a scale invariant problem. □

To solve block scale invariant problems, we consider an alternating maximization algorithm called *block SCI-PI*, which repeats

$$x_{k+1} \leftarrow \frac{\nabla_x f(x, y_k)}{\|\nabla_x f(x, y_k)\|}, \quad y_{k+1} \leftarrow \frac{\nabla_y f(x_k, y)}{\|\nabla_y f(x_k, y)\|}. \tag{3.58}$$

We present a local convergence result of block SCI-PI below.

Theorem 3.4.6. *Suppose that f is twice continuously differentiable on an open set containing $\partial\mathcal{B}_{d_1} \times \partial\mathcal{B}_{d_2}$ and let (x^*, y^*) be a local maximum satisfying*

$$\nabla_x f(x^*, y^*) = \lambda^* x^*, \ \lambda^* > \bar{\lambda}_2 = \max_{2 \leq i \leq d_1} |\lambda_i|, \ \nabla_y f(x^*, y^*) = s^* y^*, \ s^* > \bar{s}_2 = \max_{2 \leq i \leq d_2} |s_i|$$

where (λ_i, v_i) *and* (s_i, u_i) *are eigen-pairs of* $\nabla_x^2 f(x^*, y^*)$ *and* $\nabla_y^2 f(x^*, y^*)$*, respectively with* $x^* = v_1$ *and* $y^* = u_1$*. If*

$$\nu^2 = \|\nabla_{yx} f(x^*, y^*)\|^2 < (\lambda^* - \bar{\lambda}_2)(s^* - \bar{s}_2),$$

then for the sequence of iterates $\{(x_k, y_k)\}_{k=0,1,\cdots}$ *generated by* (3.58)*, there exists some* $\delta > 0$ *such that if* $\max\{|1 - x_0^T x^*|, |1 - y_0^T y^*|\} < \delta$*, then we have*

$$\|\Delta_k\| \leq \prod_{t=0}^{k-1} (\rho + \gamma_t) \|\Delta_0\| \text{ and } \lim_{k\to\infty} \gamma_k = 0$$

where

$$\Delta_k = \begin{bmatrix} \sqrt{1 - (x_k^T x^*)^2} \\ \sqrt{1 - (y_k^T y^*)^2} \end{bmatrix}, \ \rho = \frac{1}{2} \left[\frac{\bar{\lambda}_2}{\lambda^*} + \frac{\bar{s}_2}{s^*} + \sqrt{\left[\frac{\bar{\lambda}_2}{\lambda^*} - \frac{\bar{s}_2}{s^*} \right]^2 + \frac{4\nu^2}{\lambda^* s^*}} \right] < 1.$$

Proof. From Lemma A.1.3 with $w = x_k$, $z = y_k$, we have

$$1 - \frac{(\nabla_x f(x_k, y_k)^T x^*)^2}{\|\nabla_x f(x_k, y_k)\|^2} \leq \left(\frac{\bar{\lambda}_2}{\lambda^*} \sqrt{1 - (x_k^T x^*)^2} + \frac{\nu}{\lambda^*} \|y_k - y^*\| + \theta^x(x_k, y_k) \right)^2.$$

Since

$$x_{k+1} = \frac{\nabla_x f(x_k, y_k)}{\|\nabla_x f(x_k, y_k)\|},$$

we obtain

$$\sqrt{1 - (x_{k+1}^T x^*)^2} \leq \frac{\bar{\lambda}_2}{\lambda^*} \sqrt{1 - (x_k^T x^*)^2} + \frac{\nu}{\lambda^*} \|y_k - y^*\| + \theta^x(x_k, y_k).$$

Using

$$\|y_k - y^*\| = \sqrt{2(1 - y_k^T y^*)} = \left(1 + \frac{1 - y_k^T y^*}{1 + y_k^T y^* + \sqrt{2(1 + y_k^T y^*)}}\right) \sqrt{1 - (y_k^T y^*)^2},$$

we have

$$\sqrt{1 - (x_{k+1}^T x^*)^2} \leq \frac{\bar{\lambda}_2}{\lambda^*} \sqrt{1 - (x_k^T x^*)^2} + \frac{\nu}{\lambda^*} \sqrt{1 - (y_k^T y^*)^2} + \bar{\theta}^x(x_k, y_k) \tag{3.59}$$

where

$$\bar{\theta}^x(x_k, y_k) = \theta^x(x_k, y_k) + \frac{\left(1 - y_k^T y^*\right) \sqrt{1 - (y_k^T y^*)^2}}{1 + y_k^T y^* + \sqrt{2(1 + y_k^T y^*)}} = o\left(\left\| \begin{bmatrix} x_k - x^* \\ y_k - y^* \end{bmatrix} \right\|\right).$$

Using Lemma A.1.3 for $w = y_k$, $z = x_k$ and the definition of y_{k+1}, we have

$$\sqrt{1 - (y_{k+1}^T y^*)^2} \leq \frac{\nu}{s^*} \sqrt{1 - (x_k^T x^*)^2} + \frac{\bar{s}_2}{s^*} \sqrt{1 - (y_k^T y^*)^2} + \bar{\theta}^y(x_k, y_k) \tag{3.60}$$

where

$$\bar{\theta}^y(x_k, y_k) = \theta^y(x_k, y_k) + \frac{\left(1 - x_k^T x^*\right) \sqrt{1 - (x_k^T x^*)^2}}{1 + x_k^T x^* + \sqrt{2(1 + x_k^T x^*)}} = o\left(\left\| \begin{bmatrix} x_k - x^* \\ y_k - y^* \end{bmatrix} \right\|\right).$$

Combining (3.59) and (3.60), we obtain

$$\begin{bmatrix} \sqrt{1 - (x_{k+1}^T x^*)^2} \\ \sqrt{1 - (y_{k+1}^T y^*)^2} \end{bmatrix} \leq \begin{bmatrix} \frac{\bar{\lambda}_2}{\lambda^*} & \frac{\nu}{\lambda^*} \\ \frac{\nu}{s^*} & \frac{\bar{s}_2}{s^*} \end{bmatrix} \begin{bmatrix} \sqrt{1 - (x_k^T x^*)^2} \\ \sqrt{1 - (y_k^T y^*)^2} \end{bmatrix} + \begin{bmatrix} \bar{\theta}^x(x_k, y_k) \\ \bar{\theta}^y(x_k, y_k) \end{bmatrix} \tag{3.61}$$

$$\leq (M + N(x_k, y_k)) \begin{bmatrix} \sqrt{1 - (x_k^T x^*)^2} \\ \sqrt{1 - (y_k^T y^*)^2} \end{bmatrix} \tag{3.62}$$

where

$$M = \begin{bmatrix} \dfrac{\bar{\lambda}_2}{\lambda^*} & \dfrac{\nu}{\lambda^*} \\ \dfrac{\nu}{s^*} & \dfrac{\bar{s}_2}{s^*} \end{bmatrix}, \quad \epsilon(x,y) = \frac{\max\{\bar{\theta}^x(x,y), \bar{\theta}^y(x,y)\}}{\sqrt{2 - x^T x^* - y^T y^*}},$$

and

$$N(x,y) = \frac{\epsilon(x,y)}{\sqrt{2 - x^T x^* - y^T y^*}} \begin{bmatrix} \sqrt{\dfrac{1 - x^T x^*}{1 + x^T x^*}} & \sqrt{\dfrac{1 - y^T y^*}{1 + y^T y^*}} \\ \sqrt{\dfrac{1 - x^T x^*}{1 + x^T x^*}} & \sqrt{\dfrac{1 - y^T y^*}{1 + y^T y^*}} \end{bmatrix}.$$

Note that the spectral radius ρ of M satisfies

$$\rho = \frac{1}{2}\left(\frac{\bar{\lambda}_2}{\lambda^*} + \frac{\bar{s}_2}{s^*} + \sqrt{\left(\frac{\bar{\lambda}_2}{\lambda^*} - \frac{\bar{s}_2}{s^*} \right)^2 + \frac{4\nu^2}{\lambda^* s^*}} \right) < 1$$

due to $\nu^2 < (\lambda^* - \bar{\lambda}_2)(s^* - \bar{s}_2)$. Also, for $i, j = 1, 2$, we have $\lim_{(x,y)\to(x^*,y^*)} N_{ij}(x,y) = 0$. By Lemma A.1.5, there exists a sequence ω_t such that

$$\|M^k\| = \prod_{t=0}^{k-1} (\rho + \omega_t) \quad \text{and} \quad \lim_{t\to\infty} \omega_t = 0.$$

Let

$$\tau = \min\{k : \|M^k\| < 1\}, \quad \bar{\rho} = \frac{\|M^\tau\| + 1}{2}, \quad \rho_{\max} = \max_{1 \le k \le \tau} \|M^k\|.$$

By Lemma A.1.3, we have

$$\nabla_x f(x,y)^T v_1 = \lambda^* + (y - y^*)^T \nabla^2_{yx} f(x^*, y^*) x^* + \alpha^x(x,y)$$

$$\nabla_y f(x,y)^T u_1 = s^* + (x - x^*)^T \nabla^2_{xy} f(x^*, y^*) y^* + \alpha^y(x,y)$$

where

$$\alpha^x(x,y) = o\left(\left\|\begin{bmatrix} x_k - x^* \\ y_k - y^* \end{bmatrix}\right\|\right), \quad \alpha^y(x,y) = o\left(\left\|\begin{bmatrix} x_k - x^* \\ y_k - y^* \end{bmatrix}\right\|\right).$$

Therefore, there exists some $\delta_1 > 0$ such that if

$$x^T x^* > 0, \quad y^T y^* > 0, \quad \left\|\begin{bmatrix} \sqrt{1-(x^T x^*)^2} \\ \sqrt{1-(y^T y^*)^2} \end{bmatrix}\right\| < \delta_1,$$

then

$$\nabla_x f(x,y)^T v_1 > 0, \quad \nabla_y f(x,y)^T u_1 > 0. \tag{3.63}$$

Also, since $N_{ij}(x,y) \to 0$ as $(x,y) \to (x^*,y^*)$ for $i,j = 1,2$, there exists some $\delta_2 > 0$ such that if

$$x^T x^* > 0, \quad y^T y^* > 0, \quad \left\|\begin{bmatrix} \sqrt{1-(x^T x^*)^2} \\ \sqrt{1-(y^T y^*)^2} \end{bmatrix}\right\| < \delta_2,$$

then we have

(3.64)

$$\left\|\prod_{l=0}^{\tau-1}\left(M + N(\phi(x,y,l))\right)\right\| < \bar{\rho}, \quad \max_{0<m\leq\tau}\left\|\prod_{l=0}^{m-1}\left(M + N(\phi(x,y,l))\right)\right\| < 1 + \rho_{\max}$$

where $\phi(x,y,l)$ denotes the vector after l iterations of the algorithm starting with (x,y). To see this, let us define

$$g(x,y,m) = \left\|\prod_{l=0}^{m-1}\left(M + N(\phi(x,y,l))\right)\right\|.$$

By (3.62) and (3.63), if $x \to x^*$ and $y \to y^*$, then for any $0 \leq l \leq \tau$, we have

$$\phi(x,y,l) \to (x^*,y^*),$$

resulting in

$$g(x, y, m) \to \|M^m\|.$$

Therefore, there exists some $\delta_{2,\tau} > 0$ such that $g(x, y, \tau) < \bar{\rho}$. Also, for each $1 \le m < \tau$, there exists some $\delta_{2,m} > 0$ such that $g(x, y, m) < 1 + \rho_{\max}$. Taking the minimum of $\delta_{2,m}$ for $1 \le m \le \tau$, we obtain δ_2 satisfying (3.64).

Let

$$\delta = \frac{\bar{\delta}}{\sqrt{2}}, \quad \bar{\delta} = \min\left\{\delta_1, \frac{\delta_1}{1+\rho_{\max}}, \delta_2, 1\right\}, \quad N_k = N(x_k, y_k).$$

By mathematical induction, we show that for any $n \ge 0$, if

$$x_{n\tau}^T x^* > 0, \quad y_{n\tau}^T y^* > 0, \quad \Delta_{n\tau} < \bar{\delta}, \tag{3.65}$$

then for $0 \le m \le \tau$, we have

$$x_{n\tau+m}^T x^* > 0, \quad y_{n\tau+m}^T y^* > 0, \quad \Delta_{n\tau+m} \le (1+\rho_{\max})\Delta_{n\tau} < \delta_1. \tag{3.66}$$

By (3.65), it is obvious that we have (3.66) for $m = 0$. This proves the base case. Next, suppose that we have (3.66) for $0 \le m < \tau$. Then, by the definition of δ_1, we have

$$x_{n\tau+m+1}^T x^* = x_{n\tau+m+1}^T v_1 = \frac{\nabla_x f(x_{n\tau+m}, y_{n\tau+m})^T v_1}{\|\nabla_x f(x_{n\tau+m}, y_{n\tau+m})\|} > 0$$

and

$$y_{n\tau+m+1}^T y^* = y_{n\tau+m+1}^T u_1 = \frac{\nabla_y f(x_{n\tau+m}, y_{n\tau+m})^T u_1}{\|\nabla_y f(x_{n\tau+m}, y_{n\tau+m})\|} > 0.$$

Also, by (3.62), (3.65) and (3.64), we have

$$\Delta_{n\tau+m+1} \leq \|\prod_{l=0}^{m}(M+N_{n\tau+l})\|\,\Delta_{n\tau} \leq (1+\rho_{\max})\Delta_{n\tau} < \delta_1.$$

This completes the induction proof.

Suppose that (x_0, y_0) satisfies $\max\{|1-x_0^T x^*|, |1-y_0^T y^*|\} < \delta$. Then, we have

$$x_0^T x^* > 0, \quad y_0^T y^* > 0, \quad \Delta_0 < \bar{\delta}. \tag{3.67}$$

Now, we show

$$x_{n\tau}^T x^* > 0, \quad y_{n\tau}^T y^* > 0, \quad \Delta_{n\tau} \leq \bar{\rho}^n \Delta_0. \tag{3.68}$$

For $n = 0$, we have (3.68) by (3.67). This proves the base case. Next, suppose that we have (3.68) for n. Then, since (3.68) implies that $\Delta_{n\tau} \leq \bar{\rho}^n \Delta_0 < \bar{\delta}$, by (3.66), we have

$$x_{(n+1)\tau}^T x^* > 0, \quad y_{(n+1)\tau}^T y^* > 0.$$

Moreover, using (3.62) and (3.64), we have

$$\Delta_{(n+1)\tau} \leq \left\|\prod_{l=0}^{\tau-1}(M+N_{n\tau+l})\right\|\Delta_{n\tau} \leq \bar{\rho}\Delta_{n\tau} < \bar{\rho}^{n+1}\Delta_0,$$

which completes the induction proof. By repeatedly applying (3.68), we have

$$(x_{n\tau}, y_{n\tau}) \to (x^*, y^*) \text{ as } n \to \infty.$$

Furthermore, due to (3.66), we have

$$(x_{n\tau+m}, y_{n\tau+m}) \to (x^*, y^*) \text{ for every } 0 < m \leq \tau,$$

indicating that

$$(x_k, y_k) \to (x^*, y^*).$$

This in turn implies that $N_k \to 0$. Letting

$$\eta_k = \frac{\| \prod_{t=0}^{k}(M + N_t)\|}{\| \prod_{t=0}^{k-1}(M + N_t)\|} - \frac{\|M^{k+1}\|}{\|M^k\|}, \quad \gamma_k = \omega_k + \eta_k,$$

we have

$$\| \prod_{t=0}^{k-1}(M + N_t)\| = \prod_{t=0}^{k-1}(\rho + \omega_t + \eta_t) = \prod_{t=0}^{k-1}(\rho + \gamma_t). \tag{3.69}$$

Since $\eta_k \to 0$ as $N_k \to 0$, we have $\lim \gamma_k = 0$. This concludes the proof. □

If x and y are independent ($\nu = 0$), we have $\rho = \max\{\bar{\lambda}_2/\lambda^*, \bar{s}_2/s^*\}$. Otherwise, ρ increases as ν increases. Note that the result of Theorem 3.3.2 can be restored by dropping x or y in Theorem 3.4.6. While we consider the two-block case, the algorithm and the convergence analysis can be easily generalized to more than two blocks.

3.4.3. Partially Scale Invariant Problems

Lastly, we consider a class of optimization problems of the form

$$\max_{x,y} \quad f(x, y) \quad \text{subject to} \quad x \in \partial\mathcal{B}_{d_1}$$

where $f(x,y):\mathbb{R}^{d_1+d_2}\to\mathbb{R}$ is a scale invariant function in x for each $y\in\mathbb{R}^{d_2}$. A partially scale invariant problem has the form of (1.1) with respect to x once y is fixed. If x is fixed, we obtain an unconstrained optimization problem with respect to y.

Example 3.4.7 (Gaussian Mixture Model (GMM))**.** *The GMM problem is defined as*

$$\max_{x} \quad \sum_{i=1}^{n}\log\sum_{k=1}^{d}x_k^2\mathcal{N}(x_i;\mu_k,\Sigma_k) \quad \text{subject to} \quad x\in\partial\mathcal{B}_d.$$

Note that the objective function is scale invariant in x for fixed μ_k and Σ_k, and μ_k is unconstrained. If we assume some structure on Σ_k, estimation of Σ_k can also be unconstrained. For general Σ_k, semi-positive definiteness is necessary for Σ_k.

To solve partially scale invariant problems, we consider an alternative maximization algorithm based on SCI-PI and the gradient method as

$$x_{k+1}\leftarrow\frac{\nabla_x f(x_k,y_k)}{\|\nabla_x f(x_k,y_k)\|},\quad y_{k+1}\leftarrow y_k+\alpha\nabla_y f(x_k,y_k). \tag{3.70}$$

While the gradient method is used in (3.70), any method for unconstrained optimization can replace it. We present a convergence analysis of (3.70) below.

Theorem 3.4.8. *Suppose that $f(x,y)$ is scale invariant in x for each $y\in\mathbb{R}^{d_2}$, μ-strongly concave in y with an L-Lipschitz continuous $\nabla_y f(x,y)$ for each $x\in\partial\mathcal{B}_{d_1}$, and three-times continuously differentiable on an open set containing $\partial\mathcal{B}_{d_1}\times\mathbb{R}^{d_2}$. Let (x^*,y^*) be a local maximum satisfying*

$$\nabla f(x^*)=\lambda^* x^*,\ \lambda^*>\bar{\lambda}_2=\max_{2\le i\le d}|\lambda_i|$$

where (λ_i, v_i) *is an eigen-pair of* $\nabla^2 f(x^*)$ *with* $x^* = v_1$. *If*

$$\nu^2 = \|\nabla^2_{yx} f(x^*, y^*)\|^2 < \mu(\lambda^* - \bar{\lambda}_2),$$

then for the sequence of iterates $\{(x_k, y_k)\}_{k=0,1,\cdots}$ *generated by* (3.70) *with* $\alpha = \dfrac{2}{L+\mu}$, *there exists some* $\delta > 0$ *such that if* $\max\{|1 - x_0^T x^*|, \|y - y^*\|\} < \delta$, *then we have*

$$\|\Delta_k\| \leq \prod_{t=0}^{k-1} (\rho + \gamma_t) \|\Delta_0\| \text{ and } \lim_{k\to\infty} \gamma_k = 0$$

where

$$\Delta_k = \begin{bmatrix} \sqrt{1 - (x_k^T x^*)^2} \\ \|y_k - y^*\| \end{bmatrix}, \ \rho = \frac{1}{2}\left[\frac{\bar{\lambda}_2}{\lambda^*} + \frac{L-\mu}{L+\mu} + \sqrt{\left(\frac{\bar{\lambda}_2}{\lambda^*} - \frac{L-\mu}{L+\mu}\right)^2 + \frac{8\nu^2}{\lambda^*(L+\mu)}}\right] < 1.$$

Proof. Using Lemma A.1.3 for $w = x_k$, $z = y_k$ and the definition of x_{k+1}, we have

$$\sqrt{1 - (x_{k+1}^T x^*)^2} \leq \frac{\bar{\lambda}_2}{\lambda^*}\sqrt{1 - (x_k^T x^*)^2} + \frac{\nu}{\lambda^*}\|y_k - y^*\| + \theta^x(x_k, y_k). \tag{3.71}$$

where

$$\theta^x(x_k, y_k) = o\left(\left\|\begin{bmatrix} x_k - x^* \\ y_k - y^* \end{bmatrix}\right\|\right).$$

By Lemma A.1.4 with $w = x_k, z = y_k$, we also have

$$\|y_{k+1} - y^*\| \leq \left(\frac{2\nu}{L+\mu}\right)\|x_k - x^*\| + \left(\frac{L-\mu}{L+\mu}\right)\|y_k - y^*\| + \theta^y(x_k, y_k). \tag{3.72}$$

Using

$$\bar{\theta}^y(x_k, y_k) = \theta^y(x_k, y_k) + \frac{(1 - x_k^T x^*)\sqrt{1 - (x_k^T x^*)^2}}{1 + x_k^T x^* + \sqrt{2(1 + x_k^T x^*)}} = o\left(\left\|\begin{bmatrix} x_k - x^* \\ y_k - y^* \end{bmatrix}\right\|\right),$$

we can write (3.72) as

$$\|y_{k+1} - y^*\| \leq \left(\frac{2\nu}{L+\mu}\right)\sqrt{1-(x_k^T x^*)^2} + \left(\frac{L-\mu}{L+\mu}\right)\|y_k - y^*\| + \bar{\theta}^y(x_k, y_k). \tag{3.73}$$

Combining (3.71) and (3.73), we obtain

$$\begin{bmatrix} \sqrt{1-(x_{k+1}^T x^*)^2} \\ \|y_{k+1} - y^*\| \end{bmatrix} \leq \begin{bmatrix} \frac{\bar{\lambda}_2}{\lambda^*} & \frac{\nu}{\lambda^*} \\ \frac{2\nu}{L+\mu} & \frac{L-\mu}{L+\mu} \end{bmatrix} \begin{bmatrix} \sqrt{1-(x_k^T x^*)^2} \\ \|y_k - y^*\| \end{bmatrix} + \begin{bmatrix} \theta^x(x_k, y_k) \\ \bar{\theta}^y(x_k, y_k) \end{bmatrix} \tag{3.74}$$

$$\leq (M + N(x_k, y_k)) \begin{bmatrix} \sqrt{1-(x_k^T x^*)^2} \\ \|y_k - y^*\| \end{bmatrix} \tag{3.75}$$

where

$$M = \begin{bmatrix} \frac{\bar{\lambda}_2}{\lambda^*} & \frac{\nu}{\lambda^*} \\ \frac{2\nu}{L+\mu} & \frac{L-\mu}{L+\mu} \end{bmatrix}, \quad \epsilon(x,y) = \frac{\max\{\theta^x(x,y), \bar{\theta}^y(x,y)\}}{\sqrt{1 - x^T x^* + \|y - y^*\|^2}}$$

and

$$N(x,y) = \frac{\epsilon(x,y)}{\sqrt{1 - x^T x^* + \|y - y^*\|^2}} \begin{bmatrix} \sqrt{\frac{1-x^T x^*}{1+x^T x^*}} & \|y - y^*\| \\ \sqrt{\frac{1-x^T x^*}{1+x^T x^*}} & \|y - y^*\| \end{bmatrix}.$$

Since $\nu^2 < \mu(\lambda^* - \bar{\lambda}_2)$, the spectral radius ρ of M satisfies

$$\rho = \frac{1}{2}\left(\frac{\bar{\lambda}_2}{\lambda^*} + \frac{L-\mu}{L+\mu} + \sqrt{\left(\frac{\bar{\lambda}_2}{\lambda^*} - \frac{L-\mu}{L+\mu}\right)^2 + \frac{8\nu^2}{\lambda^*(L+\mu)}}\right) < 1.$$

The rest of the proof is the same as the steps taken in the proof of Theorem 3.4.6. □

As in the result of Theorem 3.4.6, the rate ρ increases as ν increases and is equal to $\max\{\bar{\lambda}_2/\lambda^*, (L-\mu)/(L+\mu)\}$ when $\nu = 0$. Also, by dropping y, we can restore the convergence result of Theorem 3.3.2.

3.5. Numerical Experiments

We test the proposed algorithms on real-world data sets. All experiments are implemented on a standard laptop (2.6 GHz Intel Core i7 processor and 16GM memory) using the Julia programming language. Let us emphasize that scale invariant problems frequently appear in many important applications in statistics and machine learning. We select three important applications, KL-NMF, GMM and ICA. A description of the data sets is provided below.

3.5.1. Description of Data Sets

Table 3.1. Summary of datasets for KL-NMF

Name	# of samples	# of features	# of nonzeros	Sparsity
WIKI	8,274	8,297	104,000	0.999
NIPS	1,500	12,419	280,000	0.985
KOS	3,430	6,906	950,000	0.960
WT	287	19,200	5,510,000	0.000

For KL-NMF (Section 3.5.2), we use four public real data sets available online[1] and summarized in Table 3.1. Waving Trees (WT) has 287 images, each having 160×120 pixels. KOS and NIPS are sparse, large matrices implemented for topic modeling. WIKI is a large binary matrix having values 0 or 1 representing the adjacency matrix of a directed graph.

[1]These are obtained from `https://www.microsoft.com/en-us/research/project`, `https://archive.ics.uci.edu/ml/datasets/bag+of+words` and `https://snap.stanford.edu/data/wiki-Vote.html`.

Table 3.2. Summary of datasets for GMM

Name	# of classes	# of samples	Dimension
Sonar	2	208	60
Ionosphere	2	351	34
HouseVotes84	2	435	16
BrCancer	2	699	10
PIDiabetes	2	768	8
Vehicle	4	846	18
Glass	6	214	9
Zoo	7	101	16
Vowel	11	990	10
Servo	51	167	4

For GMM (Section 3.5.3), we use 10 public real data sets, corresponding to all small and moderate data sets provided by the `mlbench` package in R. We select data sets for multi-class classification problems and run EM and SCI-PI for the given number of classes without class labels. In Table 3.2, the sample size varies from 101 to 990, the dimension varies from 2 to 60, and the number of classes varies from 2 to 51. Only a small portion of entries are missing, if missing data exists, and we simply impute by mean.

Table 3.3. Summary of datasets for ICA

Name	# of samples	# of features
Wine	178	14
Soybean	683	35
Vehicel	846	18
Vowel	990	10
Cardio	2,126	22
Satellite	6,435	37
Pendigits	10,992	17
Letter	20,000	16
Shuttle	58,000	9

For ICA, we use nine public data sets (see Table 3.3) from the UCI Machine Learning repository[2]. The sample size varies from 178 to 58,000 and the dimension varies from 9 to 37.

3.5.2. KL-divergence Nonnegative Matrix Factorization

We perform experiments on the KL-divergence NMF (KL-NMF) problem (3.51) described in Example 3.4.4. Let us recall that the original KL-NMF problem can be solved via block SCI-PI where in each iteration the algorithm solves the subproblem of the form (5.69). Our focus is to compare this algorithm with other well-known alternating minimization algorithms listed below, updating H and W alternatively. To lighten the notation, let $\odot$, $\oslash$ and $(\cdot)^{\odot 2}$ denote element-wise product, division and square, respectively. We let $z = V \oslash (Wh)$ and $\mathbb{1}_n$ denote a vector of ones.

- Projected gradient descent (PGD): It iterates $h^{\text{new}} \leftarrow h - \eta \odot W^T(z - \mathbb{1}_n)$ followed by projection onto the simplex, where $\eta \propto h$ is an appropriate learning rate [**48**].
- Multiplicative update (MU): A famous multiplicative update algorithm is originally suggested by [**45**], which iterates $h^{\text{new}} \leftarrow h \odot (W^T z) \oslash (W^T \mathbb{1}_n)$ and is learning rate free.
- Our method (SCI-PI): It iterates $h^{\text{new}} \leftarrow h \odot (\sigma + W^T z)^{\odot 2}$ and rescales h, where σ is a shift parameter. We simply use $\sigma = 1$ for preconditioning.
- Sequential quadratic programming (MIXSQP): It exactly solves each subproblem via a convex solver `mixsqp` [**41**]. This algorithm performs sequential non-negative least squares.

[2] `https://archive.ics.uci.edu/ml/index.php`

KL-NMF Subproblem. Note that the KL-NMF subproblem (3.53) has exactly the same form of the estimation of mixture proportions (3.22) described in the Example 3.2.7.

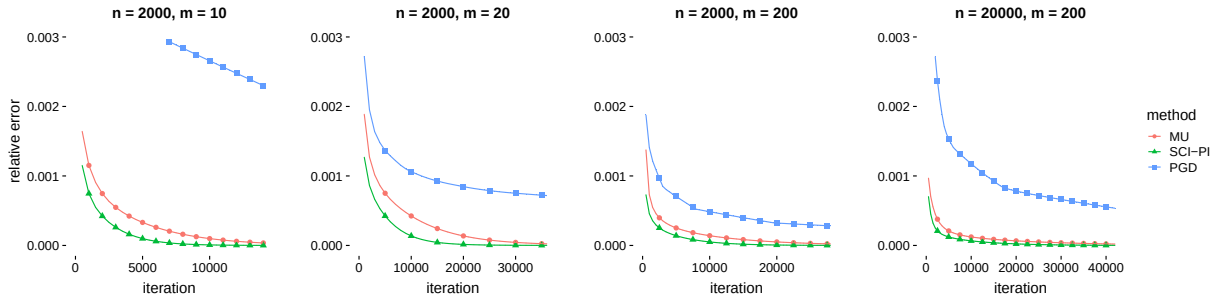

Figure 3.2. Convergence plots for the KL-NMF subproblem. n/m: the number of samples/features of the data matrix.

To study the convergence rate for the KL-NMF subproblems, we use the four data sets studied in [**41**]. We study MU, PGD and SCI-PI since they have the same order of computational complexity per iteration, but omit MIXSQP since it is a second-order method which cannot be directly compared. For PGD, the learning rate is optimized by grid search. The stopping criterion is $\|f(x_k) - f^*\| \leq 10^{-6} f^*$ where f^* is the solution obtained by MIXSQP after extensive computation time. The average runtime for aforementioned 3 methods are 33, 33 and 30 seconds for 10,000 iterations, respectively. The result is shown in Figure 3.2[3]. It shows that SCI-PI outperforms the other 2 for all simulated data sets. Also, all methods seems to exhibit linear convergence.

KL-NMF on real-world datasets. Next, we test the four algorithms on the data sets in Table 3.1. We estimate $k = 20$ factors. At each iteration, all four algorithms solve m subproblems simultaneously for W and then alternatively for H.

[3]For each evaluation, we randomly draw 10 initial points and report the averaged relative errors with respect to f^*. The initial input for the KL-NMF problem is a one-step MU update of a Unif$(0, 1)$ random matrix.

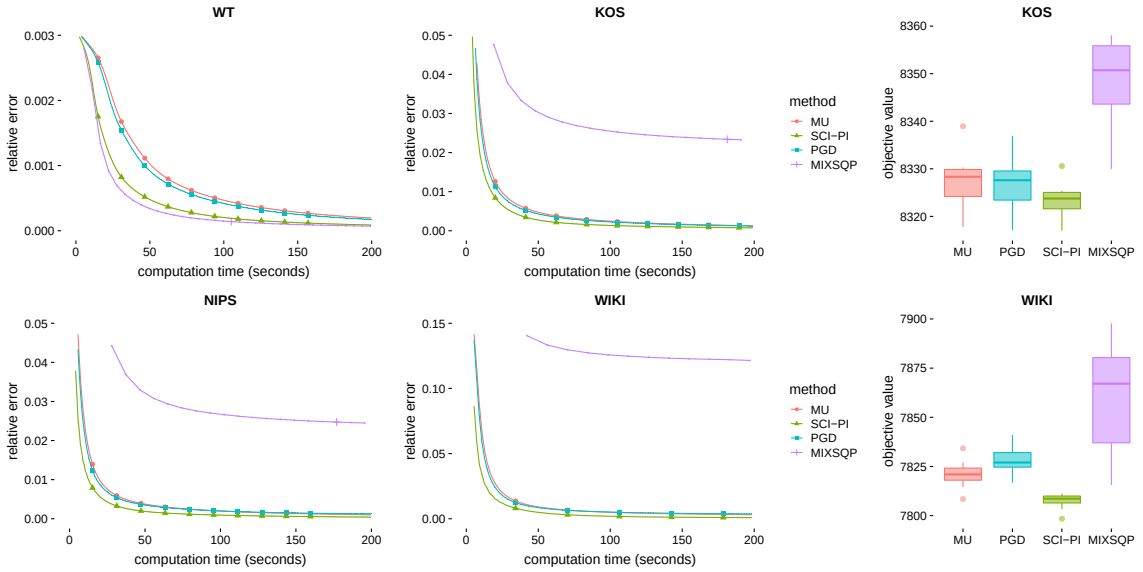

Figure 3.3. *(Left)* Convergence plots for the KL-NMF problem. *(Right)* Boxplots containing ten objective values achieved after 400 seconds.

The result is summarized in Figure 3.3[4]. The convergence plots are based on the average relative errors over 10 repeated runs with random initializations. The result shows that SCI-PI is an overall winner, showing faster convergence rates. The stopping criterion is the same as above. To assess the overall performance when initialized differently, we select KOS and WIKI and run MU, PGD, SCI-PI, and MIXSQP 10 times. The three algorithms except MIXSQP have (approximately) the same computational cost per iteration, take runtime of 391, 396, 408 seconds for KOS data and 372, 390, 418 seconds for WIKI data, respectively for 200 iterations. MIXSQP has a larger per iteration cost. After 400 seconds, SCI-PI achieves lowest objective values in all cases but one for each data set (38 out of 40 in total). Thus it clearly outperforms other methods and also achieves the lowest variance. Unlike the other three algorithms, SCI-PI is not an ascent algorithm but an

[4]In all plots we do not show the first few iterations. The initial random solutions have the gap of approximately 50% which drops to a few percent after 10 iterations where the plots start.

eigenvalue-based fixed-point algorithm. We observe that sometimes SCI-PI converges to a better solution due to this fact. Admittedly, non-monotone convergence of SCI-PI can hurt reliability of the solution but for the KL-NMF problem its performance turns out to be stable.

3.5.3. Gaussian Mixture Model and Independent Component Analysis

In this subsection, we study the empirical performance of SCI-PI when it is applied to GMM and ICA.

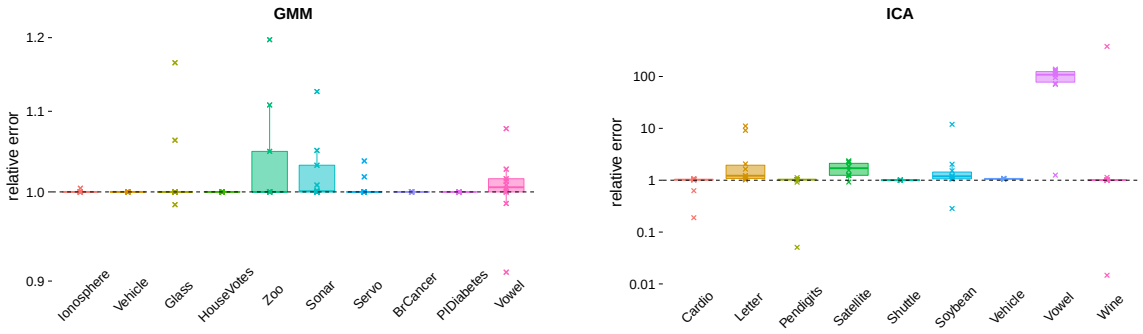

Figure 3.4. Box plots showing relative errors of *(Left)* $f^*_{\text{SCI-PI}}/f^*_{\text{EM}}$ for GMM, *(Right)* $f^*_{\text{SCI-PI}}/f^*_{\text{FastICA}}$ for ICA.

GMM. GMM fits a mixture of Gaussian distributions to the underlying data. Let $L_{ik} = \mathcal{N}(x_i; \mu_k, \Sigma_k)$ where i is the sample index and k the cluster index and let π be the actual mixture proportion vector. GMM fits into our restricted scale invariant setting (Section 3.4.3) with reparametrization, but the gradient update for μ_k, Σ_k is replaced by the exact coordinate ascent step. The EM and SCI-PI updates for π can be written

respectively as

$$(3.76)\quad r = \mathbb{1} \oslash (L\pi), \quad \pi_k^{\text{new}} \propto \pi \odot (L^T r) \quad \text{(EM)}, \quad \pi_k^{\text{new}} \propto \pi \odot (\alpha + L^T r)^{\odot 2} \quad \text{(SCI-PI)}.$$

We compare SCI-PI and EM for different real-world data sets from Table 3.2. All the algorithms initialize from the same standard Gaussian random variable, repeatedly for 10 times. The result is summarized in the left panel in Figure 3.4. The stopping criterion is $\|x_{k+1} - x_k\| < 10^{-8}$. In some cases, SCI-PI achieves much larger objective values even if initialized the same. In many cases the two algorithms exhibit the same performance. This is because estimation of μ_k's and Σ_k's are usually harder than estimation of π, and EM and SCI-PI have the same updates for μ and Σ. For a few cases EM outperforms SCI-PI. Let us mention that SCI-PI and EM have the same order of computational complexity and require 591 and 590 seconds of total computation time, respectively.

ICA. We implement SCI-PI on the Kurtosis-based ICA problem [**30**] and compare it with the benchmark algorithm FastICA [**29**], which is the most popular algorithm. Given a pre-processed[5] data matrix $W \in \mathbb{R}^{n\times d}$, we seek to maximize an approximated negative entropy $f(x) = \sum_{i=1}^n \left[(w_i^T x)^4 - 3\right]^2$ subject to $x \in \partial\mathcal{B}_d$, for maximizing Kurtosis-based non-Gaussianity [**31**]. This problem fits into the sum of scale invariant setting (Section 3.4.1). SCI-PI iterates $x_{k+1} \leftarrow W^T[(Wx_k)^{\odot 4} - 3\mathbb{1}_n) \odot (Wx_k)^{\odot 3}]$ and FastICA iterates $x_{k+1} \leftarrow W^T(Wx_k)^{\odot 3} - 3(\mathbb{1}^T(Wx_k)^{\odot 2})x_k$, both followed by normalization.

In Figure 3.4 (right panel), we compare SCI-PI and FastICA on the data sets in Table 3.3. The majority of data points (81 out of 100 in total) show that SCI-PI tends to find a better solution with a larger objective value, but in a few cases SCI-PI converges to

[5]A centered matrix $\widetilde{W} = n^{1/2}UDV^T$ is pre-processed by $W = \widetilde{W}VD^{-1}V^T$ so that $W^TW = nVV^T$.

a sub-optimal point. Both algorithms are fixed-point based and thus have no guarantee of global convergence but overall SCI-PI outperforms FastICA. SCI-PI and FastICA have the same order of computational complexity and require 11 and 12 seconds of total computation time, respectively.

3.6. Final Remarks

In this paper, we propose a new class of optimization problems called the scale invariant problems, together with a generic solver SCI-PI, which is indeed an eigenvalue-based fixed-point iteration. We showed that SCI-PI directly generalizes power iteration and enjoys similar properties such as that SCI-PI has local linear convergence under mild conditions and its convergence rate is determined by eigenvalues of the Hessian matrix at a solution. Also, we extend scale invariant problems to problems with more general settings. We show by experiments that SCI-PI can be a competitive option for numerous important problems such as KL-NMF, GMM and ICA. Finding more examples and extending SCI-PI further to a more general setting is a promising direction for future studies.

CHAPTER 4

Stochastic Power Iterations

4.1. Introduction

Principal component analysis (PCA) [**34**] is a fundamental tool for dimensionality reduction in machine learning and statistics. Given a data matrix $A = [a_1 a_2 \dots a_n] \in \mathbb{R}^{d \times n}$ consisting of n data vectors $a_1, a_2, \dots, a_n$ in $\mathbb{R}^d$, PCA finds a direction x onto which the projections of the data vectors have the largest variance. Assuming that the data vectors are standardized with a mean of zero and standard deviation of one, the PCA problem can be formulated as

$$\max_x \quad f(x) = \frac{1}{2n} \sum_{i=1}^{n} (a_i^T x)^2 = \frac{1}{2} x^T C x \quad \text{subject to} \quad x \in \partial \mathcal{B}_d \tag{4.1}$$

where $C = \frac{1}{n} A A^T \in \mathbb{R}^{d \times d}$ is the covariance matrix of data matrix A. Since the largest eigenvector u_1 of C maximizes $f(x)$, (4.1) can be solved by the singular value decomposition (SVD) of A. However, the runtime of SVD is $\mathcal{O}(\min\{nd^2, n^2 d\})$, which can be expensive in a large-scale setting. An alternative way to solve (4.1) is to use power iteration [**26**] which repeatedly applies $x_{t+1} = Cx_t / \|Cx_t\|$ at each iteration. The sequence of iterates $\{x_t\}$ generated by power iteration is guaranteed to obtain an ϵ-optimal solution after $\mathcal{O}\big(\frac{1}{\Delta} \log \frac{1}{\epsilon}\big)$ iterations where $\lambda_1 > \lambda_2 \geq \dots \geq \lambda_d \geq 0$ are the eigenvalues of C and $\Delta = 1 - \lambda_2/\lambda_1$. Since each iteration involves multiplying vector x_t with the matrix C, the runtime becomes $\mathcal{O}\big(nd \frac{1}{\Delta} \log \frac{1}{\epsilon}\big)$. When n and d are both large, the runtime of power iteration is better than

that of SVD. Nonetheless, it still largely depends on n and can be prohibitive when Δ is small.

In order to reduce the dependence on Δ or n, many variants have been developed. To improve the dependence on Δ, [**79**] propose power iteration with momentum (Power+M) based on the momentum idea of [**67**]. With the optimal choice of the momentum parameter $\beta = \lambda_2^2/4$, the total runtime reduces to $\mathcal{O}\big(nd\frac{1}{\sqrt{\Delta}}\log\frac{1}{\epsilon}\big)$. Also, a stochastic algorithm utilizing a stochastic gradient rather than a full gradient Cw_t is introduced in [**63**]. Since it requires just one data vector at a time, the computational cost per iteration is significantly reduced. However, due to the variance of stochastic gradients, a sequence of diminishing step sizes needs to be adopted, making its progress slow near the optimum.

Table 4.1. Comparison of stochastic variance-reduced PCA algorithms and their convergence analyses. Types of convergence and complexity results are summarized. "Local" means that there is a restriction on the angle between an initial iterate and the first eigenvector u_1 and "global" implies no such restriction. For VR Power and VR HB Power, $\mu \geq 0$ is a parameter that controls the progress of the algorithms through step size $\eta = \Delta^\mu$.

Algorithm	Convergence	Iteration	Batch Size	Total Runtime
VR-PCA [**72**]	Local	$\mathcal{O}\left(\frac{1}{\Delta^2}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}(1)$	$\mathcal{O}\left(d\left(n+\frac{1}{\Delta^2}\right)\log\frac{1}{\epsilon}\right)$
VR Power+M [**79**]	Local	$\mathcal{O}\left(\frac{1}{\Delta^{1/2}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{\sqrt{d}}{\Delta^{3/2}}\right)$	$\mathcal{O}\left(d\left(n+\frac{\sqrt{d}}{\Delta^2}\right)\log\frac{1}{\epsilon}\right)$
Fast PCA [**22**]	Global	$\mathcal{O}\left(\frac{1}{\Delta^2}\text{poly}\left(\log\frac{1}{\epsilon}\right)\right)$	$\mathcal{O}(1)$	$\mathcal{O}\left(d\left(n+\frac{1}{\Delta^2}\right)\text{poly}\left(\log\frac{1}{\epsilon}\right)\right)$
VR Power	**Global**	$\mathcal{O}\left(\frac{1}{\Delta^{1+\mu}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{\Delta^{1-\mu}}\right)$	$\mathcal{O}\left(d\left(n+\frac{1}{\Delta^2}\right)\log\frac{1}{\epsilon}\right)$
VR HB Power	**Global**	$\mathcal{O}\left(\frac{1}{\Delta^{1/2+\mu}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{1}{\Delta^{3/2-\mu}}\right)$	$\mathcal{O}\left(d\left(n+\frac{1}{\Delta^2}\right)\log\frac{1}{\epsilon}\right)$

Built on the recent stochastic variance-reduced gradient (SVRG) technique [**33**], [**72**,**73**] propose a stochastic variance-reduced version of Oja's algorithm (VR-PCA) and its extension for finding $k \geq 1$ principal components. Utilizing stochastic variance-reduced gradients, VR-PCA works with a constant step size and converges at an exponential rate, reducing the total runtime to $\mathcal{O}(d(n+\frac{1}{\Delta^2})\log\frac{1}{\epsilon})$. The analysis of VR-PCA considers a

batch of size one. While this implies that it works with any batch size, conditions for the step size and the epoch size are not precisely given, making it hard to attain the theoretically guaranteed performance in practice.

A stochastic variance-reduced version of Power+M (VR Power+M) is introduced by [**79**]. Due to the momentum term, the iteration complexity is improved to $\mathcal{O}\left(\frac{1}{\Delta^{1/2}}\log\frac{1}{\epsilon}\right)$. However, a batch size of $\mathcal{O}\left(\frac{\sqrt{d}}{\Delta^{3/2}}\right)$ is required to achieve such iteration complexity, leading to the total runtime of $\mathcal{O}\left(d\left(n+\frac{\sqrt{d}}{\Delta^2}\right)\log\frac{1}{\epsilon}\right)$. This runtime is worse than that of VR-PCA due to the extra dependency on $\sqrt{d}$. Moreover, if the batch size is not sufficiently large, VR Power+M may diverge, which makes it hard to use.

On other other hand, [**22**] reduce the PCA problem into a sequence of convex optimization problems. Each convex optimization problem has the form of the least square problem and amounts to one step of inverse power iteration [**26**]. Due to the finite sum structure of the objective function, the SVRG algorithm [**33**] can be used to solve the least square problem. However, solving this strongly convex optimization problem can be as hard as the original PCA problem since the objective function is $(\lambda_1 - \lambda_2)$-stronly convex and $(2\lambda_1 - \lambda_2 - \lambda_d)$-smooth in the accurate regime. Through inexactly solving these problems, an ϵ-optimal solution can be obtained after a poly-logarithmic number of iterations.

The shifted-and-inverted approach is also introduced for the leading eigenvector problem [**23**] and numerous solvers such as coordinate-descent [**75**], SVRG [**23**], accelerated gradient descent, accelerated SVRG [**2**] and Riemannian gradient descent [**80**] have been developed to solve the least square problem. Other works on power iteration include the noisy [**27**] and coordinate-wise [**46**] power methods. The noisy power method considers the power

method in a noise setting, which [**5**] extend to provide an improved gap-dependency analysis. Moreover, power iteration has been analyzed for incremental or online PCA in many works [**3**, **4**, **6**, **9**, **32**, **47**, **59**].

In this paper, we introduce two mini-batch stochastic variance-reduced PCA algorithms (VR Power, VR HB Power) and provide their convergence analyses. They are mini-batch stochastic variance-reduced variants of power iteration [**26**] and power with momentum method [**79**]. While VR-PCA [**72**] takes a data vector at a time, VR Power works with any batch sizes and the accompanying analysis reveals that whatever the batch size is, VR Power attains the optimal runtime by appropriately choosing the step size and epoch length. Explicit conditions of the step size, epoch length and batch size to ensure the optimal runtime of VR Power are derived. On the other hand, VR HB Power is an enhanced algorithm of VR Power+M. By adding the step size, VR HB Power works with any batch sizes while VR Power+M can fail unless the batch size is sufficiently large. For any batch sizes, VR HB Power can achieve the optimal runtime if we appropriately choose the step size, epoch length and momentum parameter. We derive explicit expressions for theses parameters. Our analysis improves the analysis of VR Power+M by removing the dependency on $\sqrt{d}$ for the batch size. For the comparison of stochastic variance-reduced PCA algorithms and their convergence analyses see Table 4.1.

In the convergence analyses, we introduce a novel framework of analyzing stochastic variance-reduced algorithms for PCA. For an inner-loop iterate x_t, we decompose $E[(u_k^T x_t)^2]$ with u_k an eigenvector with respect to λ_k into two parts where the first one is the expectation term and the second one is the variance term. To obtain a tight bound for the variance term, we analyze its growth over an epoch rather than focusing on iteration-by-iteration behavior.

Based on the Binomial expansion of matrices, we come up with a compact bound of the variance term, which is used to establish an upper bound of $\sum_{k=2}^{d} E[(u_k^T x_t)^2]/E[(u_1^T x_t)^2]$ and derive conditions for the step size, epoch length and batch size to ensure its sufficient decrease.

The concept of representing the optimality gap as the ratio of two expectations has been never used for analyzing stochastic PCA algorithms. However, it results in much simpler convergence statements than probabilistic statements in [**72**, **79**]. Note that we are able to obtain probabilistic statements from the expectation bounds using the Chebyshev inequality. Using the expectation bounds, we can establish global convergence of stochastic PCA algorithms. Although stochastic PCA algorithms have been observed to work well with random initialization [**72**], an initial condition such as $|u_1^T \tilde{w}_0| \geq 1/2$ is required in previous probabilistic analyses. In our framework, such condition is not necessary and the rate of convergence does not depend on how far an iterate is from u_1 but is kept the same across iterations, as in the case of deterministic power iteration. The framework introduced in this work is not specific to the proposed algorithms; it can be applied to analyze other stochastic variance-reduced PCA algorithms such as VR-PCA or VR Power+M, deriving expectation bounds for them and resolving their initialization issues.

This work has the following contributions.

(1) We introduce two mini-batch stochastic variance-reduced PCA algorithms. Regardless of the batch size, our algorithms can attain the optimal runtime by appropriately choosing algorithm parameters. Explicit expressions for these parameters are provided.

(2) We provide novel convergence analyses for the algorithms where we establish global convergence by deriving a bound for the ratio of two expectation terms. The framework in our convergence analyses is general and can be used to analyze other stochastic variance-reduced PCA algorithms. To this end, we are the first to establish convergence of VR-PCA and VR Power+M for any initial vector and in expectation.

(3) We present practical implementations of the algorithms and report numerical experiments. The experimental results on real-world datasets show that our algorithms outperform other stochastic variance-reduced algorithms for any batch size.

This chapter is organized as follows. We introduce the algorithms in Section 4.2 and the convergence analyses in Section 4.3. Some practical considerations regarding the implementations of the algorithms are discussed in Section 4.4 and the experimental results are followed in Section 4.5.

4.2. Algorithms

We consider two mini-batch stochastic variance-reduced algorithms for PCA. The first one is a mini-batch version of VR-PCA [**72**] and the second one is an enhanced version of VR Power+M [**79**] with a step size incorporated. For eigenpairs (λ_k, u_k) of C, we assume that the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_d$ satisfy $\lambda_1 > \lambda_2 \geq \ldots \geq \lambda_d \geq 0$ and the eigenvectors $u_1, u_2, \ldots, u_d$ form an orthonormal basis. Since a symmetric matrix is orthogonally diagonalizable, we can assume that such eigenvectors exist without loss of generality. We assume that all norms are L_2 for vectors and spectral for matrices.

Variance reduction algorithms have an outer loop and an inner loop. They periodically compute exact gradients at each outer iteration and use it in inner iterations to reduce the variance of stochastic gradients. Let $\tilde{x}_s$ and x_t denote an outer-loop and inner-loop iterate, respectively. To get a stochastic variance-reduced gradient of an inner loop iterate x_t, we first decompose the inner loop iterate x_t it into two parts as

$$x_t = \frac{\tilde{x}_s^T x_t}{\|\tilde{x}_s\|^2}\tilde{x}_s + \left(I - \frac{\tilde{x}_s\tilde{x}_s^T}{\|\tilde{x}_s\|^2}\right)x_t$$

using the outer loop iterate $\tilde{x}_s$. In the above decomposition, the former term represents the projection of x_t on $\tilde{x}_s$ while the latter term represents the remaining vector. Utilizing the exact gradient $\tilde{g}_s$ at $\tilde{x}_s$, the exact gradient at the first term can be computed as

$$\nabla f\left(\frac{\tilde{x}_s^T x_t}{\|\tilde{x}_s\|^2}\tilde{x}_s\right) = \frac{\tilde{x}_s^T x_t}{\|\tilde{x}_s\|^2}C\tilde{x}_s = \frac{\tilde{x}_s^T x_t}{\|\tilde{x}_s\|^2}\tilde{g}_s.$$

On the other hand, a stochastic sample S_t is used to compute a stochastic gradient at the second term as

$$\frac{1}{|S_t|}\sum_{l\in S_t} a_l a_l^T\left(I - \frac{\tilde{x}_s\tilde{x}_s^T}{\|\tilde{x}_s\|^2}\right)x_t.$$

This results in the following stochastic variance-reduced gradient g_t at x_t as

$$g_t = \frac{\tilde{x}_s^T x_t}{\|\tilde{x}_s\|^2}\tilde{g}_s + \frac{1}{|S_t|}\sum_{l\in S_t} a_l a_l^T\left(I - \frac{\tilde{x}_s\tilde{x}_s^T}{\|\tilde{x}_s\|^2}\right)x_t. \tag{4.2}$$

4.2.1. VR Power

Using the stochastic variance-reduced gradient g_t, we obtain a stochastic variance reduced version of Power iteration as

$$x_{t+1} \leftarrow (1-\eta)x_t + \eta g_t. \tag{4.3}$$

This update rule has a similar form as the one in VR-PCA, which repeats

$$x_{t+1} \leftarrow w_t + \bar{\eta}\left(a_{i_t}(a_{i_t}^T x_t - a_{i_t}^T \tilde{x}_s) + \tilde{g}_s\right). \tag{4.4}$$

Note that (4.3) generalizes (4.4) in the following two senses. First, we can obtain an update rule of (4.4) by letting $\eta = (1+\bar{\eta})/\bar{\eta}$ in (4.3). Second, with the choice of $\eta = 1$, we can recover deterministic power iteration from (4.3) while (4.4) does not. Using update rule (4.3), we have VR Power exhibited in Algorithm 3.

Algorithm 3 VR Power

Parameters: step size η, mini-batch size $|S|$, epoch length m
Input: data vectors $a_i, i = 1, \ldots, n$
randomly initialize outer iterate $\tilde{x}_0$
for $s = 0, 1, \ldots$ **do**
  $\tilde{g} \leftarrow C\tilde{x}_s$
  $x_0 \leftarrow \tilde{x}_s$
  $x_1 \leftarrow (1-\eta)x_0 + \eta\tilde{g}$
  for $t = 1, 2, \ldots, m-1$ **do**
    sample a mini-batch $S_t \subset \{1, \cdots, n\}$ of size $|S|$ uniformly at random
    $g_t \leftarrow \frac{1}{|S_t|}\sum_{l\in S_t} a_l a_l^T\big(I - \frac{x_0 x_0^T}{\|x_0\|^2}\big)x_t + \frac{(x_t^T x_0)}{\|x_0\|^2}\tilde{g}$
    $x_{t+1} \leftarrow (1-\eta)x_t + \eta g_t$
  end for
  $\tilde{x}_{s+1} \leftarrow x_m$
end for

When per sample cost is as expensive as per iteration cost, VR Power is an efficient algorithm since it attains the optimal sample complexity. However, if per sample cost is cheap, it might not be effective since its iteration complexity does not improve beyond $\mathcal{O}(\frac{1}{\Delta}\log(\frac{1}{\epsilon}))$. For this reason, we introduce VR HB Power which works better in the latter setting.

4.2.2. VR HB Power

Using g_t, we obtain a stochastic variance-reduced heavy ball power iteration as

$$x_{t+1} \leftarrow 2\big((1-\eta)x_t + \eta g_t\big) - \beta x_{t-1} \tag{4.5}$$

where $\eta \in (0, 1]$ is the step size and β is the momentum parameter. Note that we can recover the deterministic heavy ball power iteration from (4.5) when the step size η is set to 1 and the exact gradient $g_t = Cx_t$ is used. The mechanism of controlling the progress of the algorithm using the step size η is not present in VR Power+M [**79**]. As a result, it fails to converge unless the mini-batch size $|S|$ is sufficiently large. To the contrary, our algorithm works with any mini-batch size $|S|$ due to the presence of the step size η. By selecting an appropriate value of η depending on the size of $|S|$ and m, we can always ensure that the variance terms do not grow faster than expectation terms. Having update rule (4.5), VR HB Power is described in Algorithm 4.

4.3. Convergence Analyses

In this section, we provide convergence analyses for VR Power and VR HB Power. Before presenting the convergence analyses, we first introduce some notation.

Algorithm 4 VR HB Power

Parameters: step size η, momentum β, mini-batch size $|S|$, epoch length m
Input: data vectors $a_i, i = 1, \ldots, n$
randomly initialize outer iterate $\tilde{x}_0$
for $s = 0, 1, \ldots$ **do**
  $\tilde{g} \leftarrow C\tilde{x}_s$
  $x_0 \leftarrow \tilde{x}_s$
  $x_1 \leftarrow (1-\eta)x_0 + \eta\tilde{g}$
  for $t = 1, 2, \ldots, m-1$ **do**
    sample a mini-batch $S_t \subset \{1, \cdots, n\}$ of size $|S|$ uniformly at random
    $g_t \leftarrow \frac{1}{|S_t|}\sum_{l\in S_t} a_l a_l^T \big(I - \frac{x_0 x_0^T}{\|x_0\|^2}\big)x_t + \frac{(x_t^T x_0)}{\|x_0\|^2}\tilde{g}$
    $x_{t+1} \leftarrow 2\big((1-\eta)x_t + \eta g_t\big) - \beta x_{t-1}$
  end for
  $\tilde{x}_{s+1} \leftarrow x_m$
end for

4.3.1. Notation

Let C_t and P be the sample covariance matrix at inner iteration t and the projection matrix to the space orthogonal to the outer iterate $x_0 = \tilde{x}_s$ as

$$C_t = \frac{1}{|S_t|}\sum_{l\in S_t} a_l a_l^T, \quad P = I - \frac{x_0 x_0^T}{\|x_0\|^2}. \tag{4.6}$$

Using (4.6), we can write g_t as $g_t = \eta C x_t + \eta(C_t - C)Px_t$. Next, we characterize the variance of sample covariance matrix C_t as

$$K = E[\|(C_t - C)^2\|], \quad \sigma^2 = E[\|a_{i_t}a_{i_t}^T - C\|^2].$$

Then, for $M_k = E[(C_t - C)u_k u_k^T (C_t - C)]$, we have

$$\|M_k\| \le K = \frac{\sigma^2}{|S|}. \tag{4.7}$$

For the analysis of VR HB Power, we define

$$\alpha_k(\eta) = 4(1-\eta+\eta\lambda_k)^2,\ \beta(\eta) = (1-\eta+\eta\lambda_2)^2. \tag{4.8}$$

Also, we let $p_t(\alpha,\beta)$ and $q_t(\alpha,\beta)$ be the Chebyshev polynomials of the first and the second kind [**57**] respectively such that

$$p_t(\alpha,\beta) = (\alpha-\beta)p_{t-1}(\alpha,\beta) - \beta(\alpha-\beta)p_{t-2}(\alpha,\beta) + \beta^3 p_{t-3}(\alpha,\beta), \tag{4.9}$$

$$q_t(\alpha,\beta) = (\alpha-\beta)q_{t-1}(\alpha,\beta) - \beta(\alpha-\beta)q_{t-2}(\alpha,\beta) + \beta^3 q_{t-3}(\alpha,\beta) \tag{4.10}$$

for $t \geq 3$ and

$$p_0(\alpha,\beta) = 1,\ p_1(\alpha,\beta) = \frac{\alpha}{4},\ p_2(\alpha,\beta) = \left(\frac{\alpha}{2}-\beta\right)^2, \tag{4.11}$$

$$q_0(\alpha,\beta) = 1,\ q_1(\alpha,\beta) = \alpha,\ q_2(\alpha,\beta) = (\alpha-\beta)^2. \tag{4.12}$$

Since the first eigenvector u_1 of the covariance matrix C is an optimal solution to (4.1), the optimality gap is measured as $\sum_{k=2}^{d}(u_k^T x_t)^2/(u_1^T x_t)^2$, representing how closely x_t is aligned with u_1. Note that this ratio is zero if $x_t = u_1$. Our analysis studies it in expectation, providing a bound for $\theta_t = \sum_{k=2}^{d} E[(u_k^T x_t)^2]/E[(u_1^T x_t)^2]$ given fixed s and $\tilde{\theta}_s = \sum_{k=2}^{d} E[(u_k^T \tilde{x}_s)^2]/E[(u_1^T \tilde{x}_s)^2]$ for an inner loop iterate x_t and an outer loop iterate $\tilde{x}_s$, respectively.

4.3.2. VR Power

In Lemmas 4.3.1, 4.3.2 and 4.3.3, we consider a single epoch, which corresponds to one inner loop iteration starting with x_0.

Lemma 4.3.1. *For any $\eta \in (0,1]$, $1 \le k \le d$ and $1 \le t \le m$, we have*

$$E[(u_k^T x_t)^2] = (1-\eta+\eta\lambda_k)^{2t} E[(u_k^T x_0)^2] + \eta^2 \sum_{i=1}^{t-1} (1-\eta+\eta\lambda_k)^{2(t-i-1)} E[x_i^T P M_k P x_i].$$

Proof. Since $Px_0 = \left(I - x_0 x_0^T\right) x_0 = 0$, we have

$$u_k^T x_1 = (1-\eta) u_k^T x_0 + \eta u_k^T C x_0 + \eta u_k^T (C_0 - C) P x_0 = (1-\eta+\eta\lambda_k) u_k^T x_0. \tag{4.13}$$

Taking the expectation of the square of (4.13), we obtain

$$E[(u_k^T x_1)^2] = (1-\eta+\eta\lambda_k)^2 E[(u_k^T x_0)^2]. \tag{4.14}$$

For $t \ge 2$, we have

$$u_k^T x_t = (1-\eta+\eta\lambda_k) u_k^T x_{t-1} + \eta u_k^T (C_{t-1} - C) P x_{t-1}. \tag{4.15}$$

Since S_t is sampled uniformly at random, C_t is independent of $S_1, \ldots, S_{t-1}$ and x_0 with $E[C_t] = C$, leading to

$$\begin{aligned} E[u_k^T x_{t-1} u_k^T (C_{t-1} - C) P x_t] &= E[E[u_k^T x_{t-1} u_k^T (C_{t-1} - C) P x_t | x_0, S_1, \ldots, S_{t-2}]] \\ &= E[u_k^T x_{t-1} u_k^T E[C_{t-1} - C] P x_t] = 0. \end{aligned}$$

Therefore, taking the expectation of the square of (4.15), we have

$$\begin{aligned} E[(u_k^T x_t)^2] &= (1-\eta+\eta\lambda_k)^2 E[(u_k^T x_{t-1})^2] + \eta^2 E[x_{t-1}^T P (C_{t-1} - C) u_k u_k^T (C_{t-1} - C) P x_{t-1}] \\ &= (1-\eta+\eta\lambda_k)^2 E[(u_k^T x_{t-1})^2] + \eta^2 E[x_{t-1}^T P M_k P x_{t-1}] \end{aligned} \tag{4.16}$$

where the last equality follows from

$$\begin{aligned}
&E[x_{t-1}^T P(C_{t-1}-C)u_k u_k^T (C_{t-1}-C)Px_{t-1}] \\
&= E[E[x_{t-1}^T P(C_{t-1}-C)u_k u_k^T (C_{t-1}-C)Px_{t-1}|x_0, S_1, \ldots, S_{t-2}]] \\
&= E[x_{t-1}^T PE[(C_{t-1}-C)u_k u_k^T (C_{t-1}-C)]Px_{t-1}] \\
&= E[x_{t-1}^T PM_k Px_{t-1}].
\end{aligned}$$

Repeatedly applying (4.16) and using (4.14), we obtain

$$E[(u_k^T x_t)^2] = (1-\eta+\eta\lambda_k)^{2t} E[(u_k^T x_0)^2] + \eta^2 \sum_{i=1}^{t-1} (1-\eta+\eta\lambda_k)^{2(t-i-1)} E[x_i^T PM_k Px_i].$$

□

Lemma 4.3.1 decomposes $E[(u_k^T x_t)^2]$ into two parts. The first part represents the expectation term which grows at a rate of $(1-\eta+\eta\lambda_k)^2$ and the second part is the variance term which increases as x_t strides away from x_0 as captured by $E[x_t^T PM_k Px_t]$.

Lemma 4.3.2. *For any $\eta \in (0,1]$, $1 \le k \le d$ and $1 \le t \le m$, we have*

$$\sum_{k=2}^{d} E[x_t^T PM_k Px_t] \le 2K \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot \left[(1-\eta+\eta\lambda_1)^2 + \eta^2 K\right]^t.$$

Moreover, if $0 < \dfrac{\eta^2 Km}{(1-\eta+\eta\lambda_1)^2} < 1$, then we have

$$\theta_m \le \left[\left(\frac{1-\eta+\eta\lambda_2}{1-\eta+\eta\lambda_1}\right)^{2m} + \frac{4\eta^2 Km}{(1-\eta+\eta\lambda_1)^2}\right] \cdot \theta_0.$$

Proof. By Lemma A.2.2, we have

(4.17)
$$\sum_{k=2}^{d} E[x_t^T P M_k P x_t] = \sum_{k=2}^{d} E[x_t^T P M_k P x_t] = E[x_t^T P \sum_{k=2}^{d} M_k P x_t] \leq \| \sum_{k=2}^{d} M_k \| \cdot E[\|Px_t\|^2].$$

Using the Jensen's inequality and the fact that $\| \sum_{k=2}^{d} u_k u_k^T \| = 1$, we have

$$\| \sum_{k=2}^{d} M_k \| = \| \sum_{k=2}^{d} E[(C_t - C) u_k u_k^T (C_t - C)] \| \leq E[\|C_t - C\|^2] = E[\|(C_t - C)^2\|] = K,$$

resulting in

$$\sum_{k=2}^{d} E[x_t^T P M_k P x_t] \leq K E[\|Px_t\|^2]. \tag{4.18}$$

Let

$$B = (1-\eta)I + \eta C, \quad B_i = (1-\eta)I + \eta C + \eta(C_i - C)P.$$

Since $Px_0 = 0$ and

$$\begin{aligned}
&\prod_{i=t-1}^{0} B_i = \prod_{i=t-1}^{1} B_i \eta (C_0 - C) P + \prod_{i=t-1}^{1} B_i \left((1-\eta)I + \eta C \right) \\
&= \prod_{i=t-1}^{1} B_i \eta (C_0 - C) P + \sum_{j=1}^{t-1} \prod_{i=t-1}^{j+1} B_i \eta (C_j - C) P \left[(1-\eta)I + \eta C \right]^j + \left[(1-\eta)I + \eta C \right]^t \\
&= \prod_{i=t-1}^{1} B_i \eta (C_0 - C) P + \sum_{j=1}^{t-1} \prod_{i=t-1}^{j+1} B_i \eta (C_j - C) P B^j + B^t,
\end{aligned}$$

which can be seen by elementary manipulation, we have

$$x_t = \prod_{i=t-1}^{0} B_i x_0 = \left[\sum_{j=1}^{t-1} \prod_{i=t-1}^{j+1} B_i \eta (C_j - C) P B^j + B^t \right] x_0,$$

resulting in

$$Px_t = \left[\sum_{j=1}^{t-1} P \prod_{i=t-1}^{j+1} B_i \eta (C_j - C) P B^j + P B^t\right] x_0. \tag{4.19}$$

Since $C_0, \cdots, C_{t-1}$ are independent with $E[C_i] = C$ for all $1 \leq i \leq t-1$, we obtain

$$E\Big[x_0^T B^t P^2 \prod_{i=t-1}^{j+1} B_i \eta (C_j - C) P B^j x_0\Big] = 0 \tag{4.20}$$

$$E\Big[x_0^T B^{j_1} P (C_{j_1} - C) \eta \prod_{i=j_1+1}^{t-1} B_i P^2 \prod_{i=t-1}^{j_2+1} B_i \eta (C_{j_2} - C) P B^{j_2} x_0\Big] = 0 \tag{4.21}$$

where $1 \leq j, j_1, j_2 \leq t-1$ and $j_1 \neq j_2$. Therefore, we have

$$E[\|Px_t\|^2] = \sum_{j=1}^{t-1} E\big[\big\|P \prod_{i=t-1}^{j+1} B_i \eta (C_j - C) P B^j x_0\big\|^2\big] + E[\|PB^t x_0\|^2] \tag{4.22}$$

due to cross-terms being 0 from (4.20) and (4.21) when "squaring" (4.19). Using Lemma A.2.1 with $x = x_0/\|x_0\|$ and the fact that $\|x_0\|^2(1-(u_1^T x_0)^2/\|x_0\|^2) = \sum_{k=2}^{d}(u_k^T x_0)^2$, we have

$$E[\|PB^t x_0\|^2] \leq 2(1 - \eta + \eta\lambda_1)^{2t} \sum_{k=2}^{d} E[(u_k^T x_0)^2]. \tag{4.23}$$

By Lemma A.2.2 and $\|P\| = 1$, we have

$$\big\|P \prod_{i=t-1}^{j+1} B_i \eta (C_j - C) P B^j x_0\big\|^2 \leq \eta^2 \big\| \prod_{i=t-1}^{j+1} B_i (C_j - C) P B^j x_0\big\|^2. \tag{4.24}$$

Moreover, by repeatedly using first the property that B_i is independent of $x_0, C_j, B_{j+1}, \cdots, B_{i-1}$ and Lemma A.2.2, we have

$$
\begin{aligned}
&E\big[\big\| \prod_{i=t-1}^{j+1} B_i(C_j - C)PB^j x_0\big\|^2\big] \\
&= E[x_0^T B^j P(C_j - C)\Big(\prod_{i=t-2}^{j+1} B_i\Big)^T B_{t-1}^T B_{t-1} \prod_{i=t-2}^{j+1} B_i P(C_j - C)B^j x_0] \\
&= E[x_0^T B^j P(C_j - C)\Big(\prod_{i=t-2}^{j+1} B_i\Big)^T E[B_{t-1}^T B_{t-1}] \prod_{i=t-2}^{j+1} B_i P(C_j - C)B^j x_0] \\
&\leq \|E[B_{t-1}^T B_{t-1}]\| \cdot E\big[\big\| \prod_{i=t-2}^{j+1} B_i(C_j - C)PB^j x_0\big\|^2\big] \\
&\leq \prod_{i=t-1}^{j+1} \big\|E[B_i^T B_i]\big\| \cdot E[\|(C_j - C)PB^j x_0\|^2].
\end{aligned}
$$

In the same way, using the fact that C_j is independent of x_0 and Lemma A.2.2, we have

$$E[\|(C_j - C)PB^j x_0\|^2] \leq \|E[(C_j - C)^2]\| \cdot E[\|PB^j x_0\|^2],$$

resulting in

(4.25)

$$E\big[\big\| \prod_{i=t-1}^{j+1} B_i(C_j - C)PB^j x_0\big\|^2\big] \leq \prod_{i=t-1}^{j+1} \|E[B_i^T B_i]\| \cdot \|E[(C_j - C)^2]\| \cdot E[\|PB^j x_0\|^2].$$

Since C_i is independent of x_0 and $E[C_i] = C$, we have

$$\|E[B_i^T B_i]\| \leq \|B^2\| + \eta^2 \|E[P(C_i - C)^2 P]\|.$$

Since all induced norms are convex, using the Jensen's inequality, we have

$$\|E[P(C_i - C)^2 P]\|] \leq E[\|P(C_i - C)^2 P\|] \leq E[\|(C_i - C)^2\|] = K,$$

leading to

$$\|E[B_i^T B_i]\| \leq \|B^2\| + \eta^2 \|E[P(C_i - C)^2 P]\| \leq (1 - \eta + \eta\lambda_1)^2 + \eta^2 K. \tag{4.26}$$

In the same way, we obtain

$$\|E[(C_j - C)^2]\| \leq E[\|(C_j - C)^2\|] = K. \tag{4.27}$$

Using (4.26), (4.27) and (4.23) for (4.25), we have

$$\begin{aligned} E\big[\big\| \prod_{i=t-1}^{j+1} B_i (C_j - C) P B^j x_0 \big\|^2\big] \leq K \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot (1 - \eta + \eta\lambda_1)^{2j} \\ \cdot \left[(1 - \eta + \eta\lambda_1)^2 + \eta^2 K\right]^{t-j-1}. \end{aligned} \tag{4.28}$$

From (4.22), (4.23), (4.24) and (4.28), we finally have

$$\begin{aligned} E[\|Px_t\|^2] &\leq 2\eta^2 K \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot \sum_{j=1}^{t-1} \left[(1 - \eta + \eta\lambda_1)^2 + \eta^2 K\right]^{t-j-1} (1 - \eta + \eta\lambda_1)^{2j} \\ &\quad + 2 \cdot (1 - \eta + \eta\lambda_1)^{2t} \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \\ &\leq 2 \left[(1 - \eta + \eta\lambda_1)^2 + \eta^2 K\right]^t \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2], \end{aligned}$$

where the last inequality can be checked by elementary manipulation. This results in

$$\sum_{k=2}^{d} E[x_t^T P M_k P x_t] \leq 2K \left[(1-\eta+\eta\lambda_1)^2 + \eta^2 K\right]^t \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2]. \tag{4.29}$$

This proves the first part of the proof.

Next, we have

$$\begin{aligned}
&\sum_{k=2}^{d}\sum_{i=1}^{t-1}(1-\eta+\eta\lambda_k)^{2(t-i-1)} E[x_i^T P M_k P x_i] \\
&\leq (1-\eta+\eta\lambda_1)^{2t} \cdot \sum_{i=1}^{t-1}(1-\eta+\eta\lambda_1)^{-2(i+1)} \sum_{k=2}^{d} E[x_i^T P M_k P x_i]
\end{aligned}$$

and

$$\begin{aligned}
&\sum_{i=1}^{t-1}(1-\eta+\eta\lambda_1)^{-2(i+1)} \left[(1-\eta+\eta\lambda_1)^2 + \eta^2 K\right]^i \\
&\leq \frac{1}{(1-\eta+\eta\lambda_1)^2} \sum_{i=1}^{t-1} \left(\frac{(1-\eta+\eta\lambda_1)^2 + \eta^2 K}{(1-\eta+\eta\lambda_1)^2}\right)^i \\
&\leq \frac{1}{\eta^2 K} \left[\left(1 + \frac{\eta^2 K}{(1-\eta+\eta\lambda_1)^2}\right)^{t-1} - 1\right] \left(1 + \frac{\eta^2 K}{(1-\eta+\eta\lambda_1)^2}\right) \\
&\leq \frac{1}{\eta^2 K} \left[\exp\left(\frac{\eta^2 K t}{(1-\eta+\eta\lambda_1)^2}\right) - 1\right].
\end{aligned}$$

Using the condition that

$$0 < \frac{\eta^2 K m}{(1-\eta+\eta\lambda_1)^2} < 1$$

and the fact $\exp(x) - 1 \leq 2x$ for all $x \in (0, 1)$, we further obtain

$$\sum_{i=1}^{t-1}(1-\eta+\eta\lambda_1)^{-2(i+1)}\left[(1-\eta+\eta\lambda_1)^2+\eta^2 K\right]^i \leq \frac{2t}{(1-\eta+\eta\lambda_1)^2}.$$

Combined with (4.29), this results in

$$\begin{aligned}
&\eta^2\sum_{k=2}^{d}\sum_{i=1}^{m-1}(1-\eta+\eta\lambda_k)^{2(m-i-1)}E[x_i^T P M_k P x_i] \\
&\leq \eta^2\sum_{i=1}^{m-1}(1-\eta+\eta\lambda_k)^{2(m-i-1)}\sum_{k=2}^{d}E[x_i^T P M_k P x_i] \\
&\leq 4\eta^2 K m(1-\eta+\eta\lambda_1)^{2(m-1)}\cdot\sum_{k=2}^{d}E[(u_k^T x_0)^2].
\end{aligned}$$

Using Lemma 4.3.1 for $t = m$ and the fact that $(1-\eta+\eta\lambda_k)^{2m} \leq (1-\eta+\eta\lambda_2)^{2m}$ for $k \geq 2$, we finally have

$$\begin{aligned}
\sum_{k=2}^{d}E[(u_k^T x_m)^2] &= \sum_{k=2}^{d}(1-\eta+\eta\lambda_k)^{2m}E[(u_k^T x_0)^2] \\
&\quad+\eta^2\sum_{k=2}^{d}\sum_{i=1}^{m-1}(1-\eta+\eta\lambda_k)^{2(m-i-1)}E[x_i^T P M_k P x_i] \\
&\leq \left((1-\eta+\eta\lambda_2)^{2m}+4\eta^2 K m(1-\eta+\eta\lambda_1)^{2(m-1)}\right)\cdot\sum_{k=2}^{d}E[(u_k^T x_0)^2]. \qquad (4.30)
\end{aligned}$$

On the other hand, by Lemma 4.3.1 and the fact that PM_kP is positive semi-definite, we have

$$(1-\eta+\eta\lambda_1)^{2m}E[(u_1^T x_0)^2] \leq E[(u_1^T x_m)^2]. \qquad (4.31)$$

Combining (4.31) with (4.30), we obtain

$$\frac{\sum_{k=2}^{d} E[(u_k^T x_m)^2]}{E[(u_1^T x_m)^2]} \leq \left[\left(\frac{1-\eta+\eta\lambda_2}{1-\eta+\eta\lambda_1} \right)^{2m} + \frac{4\eta^2 K m}{(1-\eta+\eta\lambda_1)^2} \right] \cdot \frac{\sum_{k=2}^{d} E[(u_k^T x_0)^2]}{E[(u_1^T x_0)^2]}.$$

□

Lemma 4.3.2 provides a bound for $\sum_{k=2}^{d} E[x_t^T P M_k P x_t]$, which grows at a rate not greater than $(1-\eta+\eta\lambda_1)^2 + \eta^2 K$. Using this bound and assuming some condition on η, K, and m, a bound on θ_m is derived as a function of θ_0, η, m, and K. In Lemma 4.3.3, we present explicit conditions for η, m, and $|S|$ to ensure a sufficient decrease of θ_m.

Lemma 4.3.3. *Let $\eta = \Delta^\mu$ for some $\mu \geq 0$. If m and $|S|$ satisfy*

$$m = \left\lceil \frac{(1-\eta+\eta\lambda_1)\log 2}{2\eta\lambda_1\Delta} \right\rceil \tag{4.32}$$

and

$$|S| \geq \frac{16\eta^2\sigma^2 m}{(1-\eta+\eta\lambda_1)^2}, \tag{4.33}$$

then we have $\theta_m \leq \frac{3}{4} \cdot \theta_0$.

Proof. From the conditions on η, m and $|S|$, we have

$$0 < \frac{\eta^2 K m}{(1-\eta+\eta\lambda_1)^2} < \frac{1}{16}.$$

Therefore, using Lemma 4.3.2, we have

$$\frac{\sum_{k=2}^{d} E[(u_k^T x_m)^2]}{E[(u_1^T x_m)^2]} \leq \left[\left(\frac{1-\eta+\eta\lambda_2}{1-\eta+\eta\lambda_1} \right)^{2m} + \frac{4\eta^2 K m}{(1-\eta+\eta\lambda_1)^2} \right] \cdot \frac{\sum_{k=2}^{d} E[(u_k^T x_0)^2]}{E[(u_1^T x_0)^2]}.$$

By the choice of η and m, we have

$$\left(\frac{1-\eta+\eta\lambda_2}{1-\eta+\eta\lambda_1}\right)^{2m} = \left(1-\frac{\eta(\lambda_1-\lambda_2)}{1-\eta+\eta\lambda_1}\right)^{2m} \leq \exp\left(-\frac{2\eta(\lambda_1-\lambda_2)m}{1-\eta+\eta\lambda_1}\right) \leq \frac{1}{2}.$$

Also, by the choice of η, m and $|S|$, we have

$$\frac{4\eta^2 Km}{(1-\eta+\eta\lambda_1)^2} = \frac{4\sigma^2\eta^2 m}{|S|(1-\eta+\eta\lambda_1)^2} \leq \frac{1}{4}.$$

Therefore, we have

$$\frac{\sum_{k=2}^{d} E[(u_k^T x_m)^2]}{E[(u_1^T x_m)^2]} \leq \frac{3}{4} \cdot \frac{\sum_{k=2}^{d} E[(u_k^T x_0)^2]}{E[(u_1^T x_0)^2]}.$$

□

For any $\mu \geq 0$ such that $\eta = \Delta^\mu$, Lemma 4.3.3 provides explicit values of m and $|S|$ to ensure a sufficient decrease of θ_m. In the analysis of VR-PCA, exact values of η and m to ensure the optimal runtime have not been provided. Instead, only the orders of η and m have been provided such that $\eta = c_1\Delta$ and $m = c_2/\Delta^2$, making it hard to obtain the optimal runtime in practice. Contrary to it, our analysis provides explicit expressions for m and $|S|$, being more practical. Moreover, since the term on the right-hand side of (4.33) goes to zero as μ increases, it can be also stated that for any $|S| \geq 1$, there exists some $\mu \geq 0$ and thus $\eta = \Delta^\mu$ and m (see (4.33)) such that $\theta_m \leq 3/4 \cdot \theta_0$ holds. This implies that VR Power can always attain a sufficient decrease of θ_m no matter what $|S|$ is used. We next give the main result.

Theorem 4.3.4. *Suppose that an initial vector $\tilde{x}_0$ satisfies $u_1^T\tilde{x}_0 \neq 0$ and let $\tilde{\theta}_0 = (1-(u_1^T\tilde{x}_0)^2)/(u_1^T\tilde{x}_0)^2 \geq \epsilon$ for some $\epsilon > 0$. If $\eta = \Delta^\mu$ and m and $|S|$ satisfy (4.32) and (4.33), after $\tau = \lceil \log(\tilde{\theta}_0/\epsilon)/\log(4/3) \rceil$ epochs of VR Power, we have $\tilde{\theta}_\tau \leq \epsilon$.*

Proof. By repeatedly applying Lemma 4.3.3, we have

$$\frac{\sum_{k=2}^{d} E[(u_k^T \tilde{x}_\tau)^2]}{E[(u_1^T \tilde{x}_\tau)^2]} \leq \left(\frac{3}{4}\right)^\tau \frac{\sum_{k=2}^{d} E[(u_k^T \tilde{x}_0)^2]}{E[(u_1^T \tilde{x}_0)^2]} = \left(\frac{3}{4}\right)^\tau \tilde{\theta}_0.$$

Since $\tau = \lceil \log(\tilde{\theta}_0/\epsilon)/\log(4/3) \rceil$, we have

$$\tau \log\left(\frac{3}{4}\right) \leq \log\left(\frac{\epsilon}{\tilde{\theta}_0}\right),$$

resulting in

$$\frac{\sum_{k=2}^{d} E[(u_k^T \tilde{x}_\tau)^2]}{E[(u_1^T \tilde{x}_\tau)^2]} \leq \epsilon.$$

□

Theorem 4.3.4 present a convergence result for τ epochs. Note that our result requires only a trivial assumption on $\tilde{\theta}_0$ and thus establishes global convergence. Also, since $\tau = \mathcal{O}(\log(\frac{1}{\epsilon}))$, only a logarithmic number of inner loops is needed to be performed to obtain ϵ-accuracy.

4.3.3. VR HB Poxer

The folloxing Lemmas 4.3.5, 4.3.6 and 4.3.7 are counterparts of Lemmas 4.3.1, 4.3.2 and 4.3.3 for VR HB Power. For the momentum parameter β, we let $\beta = \beta(\eta)$ which is defined in (4.8). As in the analysis of VR Power, we first consider a single epoch with an initial inner loop iterate x_0.

Lemma 4.3.5. *For any $\eta \in (0, 1]$, $1 \leq k \leq d$ and $1 \leq t \leq m$, we have*

$$E[(u_k^T x_t)^2] = p_t(\alpha_k(\eta), \beta(\eta)) E[(u_k^T x_0)^2] + 4\eta^2 \sum_{r=1}^{t-1} q_{t-r-1}(\alpha_k(\eta), \beta(\eta)) E[x_r^T P M_k P x_r].$$

Proof. From $x_1 = (1-\eta)x_0 + \eta\tilde{g} = (1-\eta)x_0 + \eta C x_0$, we have

$$u_k^T x_1 = (1-\eta)u_k^T x_0 + \eta u_k^T C x_0 = (1-\eta+\eta\lambda_k)u_k^T x_0. \tag{4.34}$$

Taking the expectation of the square of (4.34), we obtain

$$E[(u_k^T x_1)^2] = (1-\eta+\eta\lambda_k)^2 E[(u_k^T x_0)^2] = \frac{\alpha_k(\eta)}{4} E[(u_k^T x_0)^2]. \tag{4.35}$$

Next, from (4.5), we have

$$\begin{aligned}
x_{t+1} &= 2\bigg((1-\eta)x_t + \eta\frac{1}{|S_t|}\sum_{i_t\in S_t} a_{i_t}a_{i_t}^T\Big(x_t - \frac{(x_t^T x_0)}{\|x_0\|^2}x_0\Big) + \frac{(x_t^T x_0)}{\|x_0\|^2}\tilde{g}\bigg) - \beta(\eta)x_{t-1} \\
&= 2\bigg((1-\eta)x_t + \eta\frac{1}{|S_t|}\sum_{i_t\in S_t} a_{i_t}a_{i_t}^T\Big(I - \frac{x_0x_0^T}{\|x_0\|^2}\Big)x_t + C\frac{x_0x_0^T}{\|x_0\|^2}x_t\bigg) - \beta(\eta)x_{t-1} \\
&= 2\bigg((1-\eta)x_t + \eta C x_t + \eta\frac{1}{|S_t|}\sum_{i_t\in S_t}(a_{i_t}a_{i_t}^T - C)\Big(I - \frac{x_0x_0^T}{\|x_0\|^2}\Big)x_t\bigg) - \beta(\eta)x_{t-1} \\
&= 2\big((1-\eta)x_t + \eta C x_t + \eta(C_t - C)Px_t\big) - \beta(\eta)x_{t-1}, \qquad (4.36)
\end{aligned}$$

leading to

$$u_k^T x_{t+1} = 2\big((1-\eta+\eta\lambda_k)u_k^T x_t + \eta u_k^T(C_t - C)Px_t\big) - \beta(\eta)u_k^T x_{t-1}. \tag{4.37}$$

Taking the square of (4.37), we have

$$\begin{aligned}(u_k^T x_{t+1})^2 &= 4(1-\eta+\eta\lambda_k)^2(u_k^T x_t)^2 \\ &+ 4\eta^2 x_t^T P(C_t - C)u_k u_k^T (C_t - C)Px_t + (\beta(\eta))^2(u_k^T x_{t-1})^2 \\ &+ 8\eta(1-\eta+\eta\lambda_k)u_k^T x_t u_k^T (C_t - C)Px_t - 4(1-\eta+\eta\lambda_k)\beta(\eta)u_k^T x_t u_k^T x_{t-1} \\ &- 4\eta\beta(\eta)u_k^T (C_t - C)Px_t u_k^T x_{t-1}.\end{aligned}$$

Since S_t is sampled uniformly at random, C_t is independent of $S_1, \ldots, S_{t-1}$ and identically distributed with $E[C_t] = C$. Therefore,

$$E[u_k^T x_t u_k^T (C_t - C)Px_t] = E[u_k^T x_t u_k^T E[C_t - C]Px_t] = 0.$$

Similarly, we have

$$E[u_k^T (C_t - C)Px_t u_k^T x_{t-1}] = 0. \tag{4.38}$$

As a result, we obtain

$$\begin{aligned}E[(u_k^T x_{t+1})^2] &= \alpha_k(\eta)E[(u_k^T x_t)^2] - 2\sqrt{\alpha_k(\eta)}\beta(\eta)E[(u_k^T x_t)(u_k^T x_{t-1})] + (\beta(\eta))^2 E[(u_k^T x_{t-1})^2] \\ &+ 4\eta^2 E[x_t^T P M_k P x_t]. \qquad (4.39)\end{aligned}$$

Using (4.34) and (4.35) in (4.39) for $t = 1$, we have

$$E[(u_k^T x_2)^2] = \Big(\frac{\alpha_k(\eta)}{2} - \beta(\eta)\Big)^2 E[(u_k^T x_0)^2] + 4\eta^2 E[x_1^T P M_k P x_1]. \tag{4.40}$$

Moreover, by using (4.37) with $t-1$, multiplying it with $u_k^T x_{t-1}$, taking expectation and using (4.38) with x_t being x_{t-1} (which can be derived in the same way as (4.38)) , we have

$$E[(u_k^T x_t)(u_k^T x_{t-1})] = \sqrt{\alpha_k(\eta)} E[(u_k^T x_{t-1})^2] - \beta(\eta) E[(u_k^T x_{t-1})(u_k^T x_{t-2})]. \tag{4.41}$$

Using (4.41), we can further write (4.39) as

$$\begin{aligned} E[(u_k^T x_{t+1})^2] &= \alpha_k(\eta) E[(u_k x_t)^2] - \beta(\eta)(2\alpha_k(\eta) - \beta(\eta)) E[(u_k^T x_{t-1})^2] \\ &\quad + 2\sqrt{\alpha_k(\eta)}(\beta(\eta))^2 E[(u_k^T x_{t-1})(u_k^T x_{t-2})] + 4\eta^2 E[x_t^T P M_k P x_t]. \end{aligned} \tag{4.42}$$

With $t-1$ in (4.39), we have

$$\begin{aligned} E[(u_k^T x_t)^2] &= \alpha_k(\eta) E[(u_k^T x_{t-1})^2] - 2\sqrt{\alpha_k(\eta)}\beta(\eta) E[(u_k^T x_{t-1})(u_k^T x_{t-2})] \\ &\quad + (\beta(\eta))^2 E[(u_k^T x_{t-2})^2] + 4\eta^2 E[x_{t-1}^T P M_k P x_{t-1}]. \end{aligned}$$

Adding (4.43) multiplied by $\beta(\eta)$ to (4.42), we obtain

$$\begin{aligned} E[(u_k^T x_{t+1})^2] &= (\alpha_k(\eta) - \beta(\eta)) E[(u_k^T x_t)^2] - \beta(\eta)(\alpha_k(\eta) - \beta(\eta)) E[(u_k^T x_{t-1})^2] \\ &\quad + (\beta(\eta))^3 E[(u_k^T x_{t-2})^2] + 4\eta^2 E[x_t^T P M_k P x_t] + 4\eta^2 \beta(\eta) E[x_{t-1}^T P M_k P x_{t-1}]. \end{aligned} \tag{4.43}$$

With $t-1$ in (4.43), we finally have

$$\begin{aligned} E[(u_k^T x_t)^2] &= (\alpha_k(\eta) - \beta(\eta)) E[(u_k^T x_{t-1})^2] - \beta(\eta)(\alpha_k(\eta) - \beta(\eta)) E[(u_k^T x_{t-2})^2] \\ &\quad + (\beta(\eta))^3 E[(u_k^T x_{t-3})^2] + 4\eta^2 E[x_{t-1}^T P M_k P x_{t-1}] + 4\eta^2 \beta(\eta) E[x_{t-2}^T P M_k P x_{t-2}]. \end{aligned} \tag{4.44}$$

Using Lemma A.2.4 for $E[(u_k^T x_t)^2]$ defined by (4.35), (4.40) and (4.44) with

$$\alpha = \alpha_k(\eta), \quad \beta = \beta(\eta), \quad L_0 = E[(u_k^T x_0)^2], \quad L_t = 4\eta^2 E[x_t^T P M_k P x_t],$$

we have

$$E[(u_k^T x_t)^2] = p_t(\alpha_k(\eta), \beta(\eta)) E[(u_k^T x_0)^2] + 4\eta^2 \sum_{r=1}^{t-1} q_{t-r-1}(\alpha_k(\eta), \beta(\eta)) E[x_r^T P M_k P x_r].$$

□

Lemma 4.3.5 breaks $E[(u_k^T x_t)^2]$ into the sum of expectation part and variance part. While the expectation term is a function of the Chebyshev polynomial of the first kind, the variance part is a function of the Chebyshev polynomials of the second kind. That being said, the variance term grows faster and thus we need a careful analysis for it.

Lemma 4.3.6. *For any $\eta \in (0, 1]$, $1 \le k \le d$, and $1 \le t \le m$, we have*

$$\sum_{k=2}^{d} E[x_t^T P M_k P x_t] \le 4K \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot \left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{t-1} \cdot \left(\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2}\right)^{2t}.$$

Moreover, if $0 < \dfrac{4\eta^2 K m}{\alpha_1(\eta) - 4\beta(\eta)} < 1$, then we have

$$\theta_m \le \left(\frac{p_m(\alpha_2(\eta), \beta(\eta))}{p_m(\alpha_1(\eta), \beta(\eta))} + \frac{128\eta^2 K m}{\alpha_1(\eta) - 4\beta(\eta)}\right) \cdot \theta_0.$$

Proof. Since $\| \sum_{k=2}^{d} u_k u_k^T \| \le 1$, we have $\| \sum_{k=2}^{d} M_k \| = \| \sum_{k=2}^{d} E[(C_t - C) u_k u_k^T (C_t - C)] \| \le E[\|C_t - C\|^2] = E[\|(C_t - C)^2\|] = K$.

By Lemma A.2.2, this leads to

$$\sum_{k=2}^{d} E[x_t^T P M_k P x_t] = E[x_t^T P \sum_{k=2}^{d} M_k P x_t] \leq \| \sum_{k=2}^{d} M_k \| E[\|Px_t\|^2] \leq K E[\|Px_t\|^2]. \tag{4.45}$$

Let

$$F = \begin{bmatrix} I \\ 0 \end{bmatrix}, \quad G = \begin{bmatrix} 2\left[(1-\eta)I + \eta C\right] & -\beta(\eta) I \\ I & 0 \end{bmatrix}$$

and

$$G_0 = \begin{bmatrix} (1-\eta)I + \eta C & -\beta(\eta) I \\ I & 0 \end{bmatrix}, \quad H_t = 2\eta \begin{bmatrix} (C_t - C)P & 0 \\ 0 & 0 \end{bmatrix}.$$

From the update rule in Algorithm 4 expressed in (4.36), we can write

$$x_t = F^T (G + H_{t-1})(G + H_{t-2}) \cdots (G + H_1)(G_0 + H_0) F x_0.$$

Using Lemma A.2.3 for the expansion of $(G + H_{t-1})(G + H_{t-2}) \cdots (G + H_1)(G_0 + H_0)$, we have

$$Px_t = PF^T \left(G^{t-1} G_0 + \sum_{i=1}^{t-1} \left[\prod_{j=t-1}^{i+1} (G + H_j) H_i G^{i-1} G_0 \right] + \prod_{j=t-1}^{1} (G + H_j) H_0 \right) F x_0. \tag{4.46}$$

Since $C_0, C_1, \cdots, C_{t-1}$ are independent and identically distributed with mean C, so are $H_0, H_1, \cdots, H_{t-1}$ with mean 0. Therefore, the expectation of all cross-terms in the "square" of (4.46) are zero. Using the fact that $H_0 F x_0 = 0$, we have

$$E[\|Px_t\|^2] = E[\|PF^T G^{t-1} G_0 F x_0\|^2] + \sum_{i=1}^{t-1} E\Big[\Big\| PF^T \prod_{j=t-1}^{i+1} (G + H_j) H_i G^{i-1} G_0 F x_0 \Big\|^2 \Big]. \tag{4.47}$$

Note that this result is analogous to (4.22) in the analysis of VR Power. From $F^T G^{t-1} G_0 F = Y_t((1-\eta)I + \eta C, \beta(\eta))$ (see (A.36) for the definition of Y_t) and (A.40b) in Lemma A.2.1 with $x = x_0/\|x_0\|$ and the fact that $\|x_0\|^2(1 - (u_1^T x_0)^2/\|x_0\|^2) = \sum_{k=2}^d (u_k^T x_0)^2$, we have

$$E\big[\|PF^T G^{t-1} G_0 F x_0\|^2\big] = 4p_t(\alpha_1(\eta), \beta(\eta)) \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2]. \tag{4.48}$$

Using Lemma A.2.2, $\|P\| = 1$, $H_t = 2\eta F(C_t - C)PF^T$, we have

(4.49)

$$\begin{aligned}
&E\big[\big\|PF^T \prod_{j=t-1}^{i+1} (G + H_j) H_i G^{i-1} G_0 F x_0\big\|^2\big] \\
&\le 4\eta^2 \|P\|^2 \cdot E\big[\big\|F^T \prod_{j=t-1}^{i+1} (G + H_j) F(C_i - C) PF^T G^{i-1} G_0 F x_0\big\|^2\big] \\
&\le 4\eta^2 \cdot \big\|E\big[F^T\big[\prod_{j=t-1}^{i+1} (G + H_j)\big]^T FF^T \prod_{j=t-1}^{i+1} (G + H_j) F\big]\big\| E\big[\big\|(C_i - C) PF^T G^{i-1} G_0 F x_0\big\|^2\big].
\end{aligned}$$

Using mathematical induction on i, we prove that

$$\begin{aligned}
&E\big[\big[\prod_{j=t-1}^{i+1} (G + H_j)\big]^T FF^T \prod_{j=t-1}^{i+1} (G + H_j)\big] \\
&= \sum_{(v_{i+1}, \cdots, v_{t-1}) \in \{0,1\}^{t-i-1}} E\big[\big[\prod_{j=t-1}^{i+1} H_j^{1-v_j} G^{v_j}\big]^T FF^T \prod_{j=t-1}^{i+1} H_j^{1-v_j} G^{v_j}\big]
\end{aligned} \tag{4.50}$$

for any $i \le t-2$ and fixed $t \ge 2$. Since $E[H_{t-1}] = 0$, we have

$$E[(G^T + H_{t-1}^T) FF^T (G + H_{t-1})] = G^T FF^T G + E[H_{t-1}^T FF^T H_{t-1}].$$

This proves the base case for $i = t-2$.

Suppose that (4.50) holds for $i = k$. Then, since H_k is independent from $H_{k+1}, \cdots, H_{t-1}$ and $E[H_k] = 0$, we have

$$\begin{aligned}
&E\big[\big[\prod_{j=t-1}^{k}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{k}(G+H_j)\big] \\
&= G^T E\big[\big[\prod_{j=t-1}^{k+1}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{k+1}(G+H_j)\big]G \\
&\quad + E\big[H_k^T\big[\prod_{j=t-1}^{k+1}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{k+1}(G+H_j)H_k\big].
\end{aligned}$$

From (4.50), we have

$$\begin{aligned}
&G^T E\big[\big[\prod_{j=t-1}^{k+1}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{k+1}(G+H_j)\big]G \\
&= \sum_{(v_{k+1},\cdots,v_{t-1})\in\{0,1\}^{t-k-1}} E\big[\big[\big(\prod_{j=t-1}^{k+1} H_j^{1-v_j}G^{v_j}\big)G\big]^T FF^T\big(\prod_{j=t-1}^{k+1} H_j^{1-v_j}G^{v_j}\big)G\big].
\end{aligned}$$

Also, by the independence of H_k from $H_{k+1}, \cdots, H_{t-1}$ and (4.50), we have

$$\begin{aligned}
&E\big[H_k^T\big[\prod_{j=t-1}^{k+1}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{k+1}(G+H_j)H_k\big] \\
&= E\big[H_k^T E\big[\big[\prod_{j=t-1}^{k+1}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{k+1}(G+H_j)\big]H_k\big] \\
&= E\big[H_k^T \sum_{(v_{k+1},\cdots,v_{t-1})\in\{0,1\}^{t-i-1}} E\big[\big[\prod_{j=t-1}^{k+1} H_j^{1-v_j}G^{v_j}\big]^T FF^T \prod_{j=t-1}^{k+1} H_j^{1-v_j}G^{v_j}\big]H_k\big] \\
&= \sum_{(v_{k+1},\cdots,v_{t-1})\in\{0,1\}^{t-k-1}} E\big[\big[\big(\prod_{j=t-1}^{k+1} H_j^{1-v_j}G^{v_j}\big)H_k\big]^T FF^T\big(\prod_{j=t-1}^{k+1} H_j^{1-v_j}G^{v_j}\big)H_k\big].
\end{aligned}$$

Therefore, we have

$$E\big[\big[\prod_{j=t-1}^{k}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{k}(G+H_j)\big]$$
$$= \sum_{(v_k,\cdots,v_{t-1})\in\{0,1\}^{t-k}} E\big[\big[\prod_{j=t-1}^{k} H_j^{1-v_j}G^{v_j}\big]^T FF^T \prod_{j=t-1}^{k} H_j^{1-v_j}G^{v_j}\big],$$

which completes the proof of (4.50).

Using the Jensen's inequality and the norm property of a symmetric matrix, we have

(4.51)
$$\|E\big[F^T\big[\prod_{j=t-1}^{i+1}[H_j^{1-v_j}G^{v_j}]\big]^T FF^T \prod_{j=t-1}^{i+1}[H_j^{1-v_j}G^{v_j}]F\big]\| \le E\big[\|F^T\prod_{j=t-1}^{i+1}[H_j^{1-v_j}G^{v_j}]F\|^2\big].$$

For $(v_{i+1},\cdots,v_{t-1})\in\{0,1\}^{t-i-1}$, let $J=\{j_1,j_2,\cdots,j_{\bar{k}}\}$ be a set of indices such that $j_1<j_2<\cdots<j_{\bar{k}}$ and $v_j=0$ if $j\in J$ and $v_j=1$ otherwise. Also, let $j_0=i$. Using that $H_j=FF^TH_jFF^T$, we have

$$E\big[\|F^T\prod_{j=t-1}^{i+1}[H_j^{1-v_j}G^{v_j}]F\|^2\big] = E\big[\|F^TG^{t-j_{\bar{k}}-1}F\prod_{l=\bar{k}}^{1}\big(F^TH_{j_l}FF^TG^{j_l-j_{l-1}-1}F\big)\|^2\big]$$
$$\le E\big[\|F^TG^{t-j_{\bar{k}}-1}F\|^2\prod_{l=\bar{k}}^{1}\|F^TH_{j_l}F\|^2\|F^TG^{j_l-j_{l-1}-1}F\|^2\big]. \tag{4.52}$$

Since $F^TG^tF=Z_t((1-\eta)I+\eta C,\beta(\eta))$, using (A.40c) in Lemma A.2.1, we have

$$\|F^TG^tF\|^2 \le q_t(\alpha_1(\eta),\beta(\eta)). \tag{4.53}$$

Also, from that $F^T H_t F = 2\eta(C_t - C)P$, we have

$$E\big[\|F^T H_t F\|^2\big] \le 4\eta^2 E\big[\|(C_t - C)P\|^2\big] \le 4\eta^2 E\big[\|(C_t - C)\|^2\big] = 4\eta^2 K. \tag{4.54}$$

where the last inequality follows from $\|P\| = 1$ and the second last equality follows from the symmetry of $C_t - C$. Using (4.53) and Lemma A.2.5, we have

$$\|F^T G^{t-j_{\bar{k}}-1} F\|^2 \prod_{l=\bar{k}}^{1} \|F^T G^{j_l - j_{l-1}-1} F\|^2 \le \left(\frac{1}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{\bar{k}} q_{t-i-1}(\alpha_1(\eta), \beta(\eta)). \tag{4.55}$$

We use Lemma A.2.5 $\bar{k}$ times to obtain the term on the right-hand side.

Using (4.51), (4.52), (4.55), and the independence of $C_0, C_1, \cdots, C_{t-1}$, we obtain

$$\begin{aligned}
&\|E\big[F^T\big[\prod_{j=t-1}^{i+1} [H_j^{1-v_j} G^{v_j}]\big]^T F F^T \prod_{j=t-1}^{i+1} [H_j^{1-v_j} G^{v_j}] F\big]\| \\
&\le \left(\frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{\bar{k}} q_{t-i-1}(\alpha_1(\eta), \beta(\eta)).
\end{aligned}$$

Combined with (4.50), this results in

$$\begin{aligned}
&\|E\big[F^T\big[\prod_{j=t-1}^{i+1} (G + H_j)\big]^T F F^T \prod_{j=t-1}^{i+1} (G + H_j) F\big]\| \\
&= \Big\| \sum_{(v_{i+1}, \cdots, v_{t-1}) \in \{0,1\}^{t-i-1}} E\big[F^T\big[\prod_{j=t-1}^{i+1} [H_j^{1-v_j} G^{v_j}]\big]^T F F^T \prod_{j=t-1}^{i+1} [H_j^{1-v_j} G^{v_j}] F\big] \Big\| \\
&\le \sum_{(v_{i+1}, \cdots, v_{t-1}) \in \{0,1\}^{t-i-1}} \big\| E\big[F^T\big[\prod_{j=t-1}^{i+1} [H_j^{1-v_j} G^{v_j}]\big]^T F F^T \prod_{j=t-1}^{i+1} [H_j^{1-v_j} G^{v_j}] F\big] \big\| \\
&\le \sum_{\bar{k}=0}^{t-i-1} \binom{t-i-1}{\bar{k}} \left(\frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{\bar{k}} q_{t-i-1}(\alpha_1(\eta), \beta(\eta)).
\end{aligned}$$

From

$$\sum_{\bar{k}=0}^{t-i-1} \binom{t-i-1}{\bar{k}} \left(\frac{4\eta^2 K}{\alpha_1(\eta)-4\beta(\eta)}\right)^{\bar{k}} \leq \left(1+\frac{4\eta^2 K}{\alpha_1(\eta)-4\beta(\eta)}\right)^{t-i-1},$$

we further have

$$\begin{aligned} &\|E\big[F^T\big[\prod_{j=t-1}^{i+1}(G+H_j)\big]^T FF^T \prod_{j=t-1}^{i+1}(G+H_j)F\big]\| \\ &\leq q_{t-i-1}(\alpha_1(\eta),\beta(\eta))\left(1+\frac{4\eta^2 K}{\alpha_1(\eta)-4\beta(\eta)}\right)^{t-i-1} \end{aligned} \tag{4.56}$$

On the other hand, using Lemma A.2.2 and (4.48) for $t=i$, we have

$$\begin{aligned} &\eta^2 E\big[\big\|(C_i-C)PF^T G^{i-1}G_0 F x_0\big\|^2\big] \\ &= \eta^2 E[x_0 F^T G_0^T (G^{i-1})^T FP^T E[(C_i-C)^2]PF^T G^{i-1}G_0 F x_0] \\ &\leq \eta^2 \|E[(C_i-C)^2]\| E[\|PF^T G^{i-1}G_0 F x_0\|^2] \\ &\leq 4\eta^2 K \cdot p_i(\alpha_1(\eta),\beta(\eta)) \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2]. \end{aligned} \tag{4.57}$$

Using (4.56) and (4.57) to bound (4.49), we have

$$\begin{aligned} E\big[\big\|PF^T \prod_{j=t-1}^{i+1}(G+H_j)H_i G^{i-1}G_0 F x_0\big\|^2\big] &\leq 16\eta^2 K \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \\ \cdot p_i(\alpha_1(\eta),\beta(\eta)) \cdot q_{t-i-1}(\alpha_1(\eta),\beta(\eta)) &\cdot \left(1+\frac{4\eta^2 K}{\alpha_1(\eta)-4\beta(\eta)}\right)^{t-i-1}. \end{aligned} \tag{4.58}$$

Using (4.48) and (4.58) for (4.47), we finally have

$$E[\|Px_t\|^2] \leq \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot \Bigg[4p_t(\alpha_1(\eta), \beta(\eta)) \\ + 16\eta^2 K \sum_{i=1}^{t-1} p_i(\alpha_1(\eta), \beta(\eta)) \cdot q_{t-i-1}(\alpha_1(\eta), \beta(\eta)) \left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{t-i-1} \Bigg].$$

By (A.52) and (A.53) in Lemma A.2.4, we have

$$p_t(\alpha_1(\eta), \beta(\eta)) \leq \left(\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right)^{2t},$$

$$q_t(\alpha_1(\eta), \beta(\eta)) \leq \left(\frac{1}{\alpha_1(\eta) - \beta(\eta)} \right) \left(\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right)^{2(t+1)}.$$

Therefore, we obtain

$$4p_t(\alpha_1(\eta), \beta(\eta)) + 16\eta^2 K \sum_{i=1}^{t-1} p_i(\alpha_1(\eta), \beta(\eta)) \cdot q_{t-i-1}(\alpha_1(\eta), \beta(\eta)) \left[1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right]^{t-i-1}$$

$$\leq 4 \left[1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)} \sum_{i=1}^{t-1} \left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{t-i-1} \right] \left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right]^{2t}$$

$$= 4 \left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{t-1} \left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right]^{2t},$$

which results in

$$E[\|Px_t\|^2] \leq 4 \left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{t-1} \left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right]^{2t} \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2].$$

Finally, from (4.45), we have

$$\begin{aligned}\sum_{k=2}^{d} E[x_t^T P M_k P x_t] \le 4K \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot \left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{t-1} \\ \cdot \left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2}\right]^{2t}.\end{aligned} \tag{4.59}$$

This completes the proof of the first statement.

Next, from $\alpha_2(\eta) = 4\beta(\eta) \ge \alpha_k(\eta)$ for $k \ge 2$ and (A.54) in Lemma A.2.4,

$$\sum_{k=2}^{d} p_m(\alpha_k(\eta), \beta(\eta)) E[(u_k^T x_0)^2] \le p_m(\alpha_2(\eta), \beta(\eta)) \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2]. \tag{4.60}$$

Also, using (A.53) and (A.54) in Lemma A.2.4 and (4.59), we have

$$\begin{aligned}&4\eta^2 \sum_{k=2}^{d} \sum_{r=1}^{m-1} q_{m-r-1}(\alpha_k(\eta), \beta(\eta)) E[x_r^T P M_k P x_r] \\ &\le \frac{16\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)} \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot \left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2}\right]^{2m} \\ &\quad \cdot \sum_{r=1}^{m-1} \left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{r-1} \\ &\le 4 \sum_{k=2}^{d} E[(u_k^T x_0)^2] \left[\left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{m-1} - 1\right] \left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2}\right]^{2m}.\end{aligned}$$

Since $0 < \dfrac{4\eta^2 K m}{\alpha_1(\eta) - \beta(\eta)} < 1$, using that $\exp(x) \le 1 + 2x$ for $x \in [0, 1]$ we have

$$\left(1 + \frac{4\eta^2 K}{\alpha_1(\eta) - 4\beta(\eta)}\right)^{m-1} - 1 \le \exp\left(\frac{4\eta^2 K m}{\alpha_1(\eta) - 4\beta(\eta)}\right) - 1 \le \frac{8\eta^2 K m}{\alpha_1(\eta) - 4\beta(\eta)},$$

leading to

$$
\begin{aligned}
&4\eta^2 \sum_{k=2}^{d} \sum_{r=1}^{m-1} q_{m-r-1}(\alpha_k(\eta), \beta(\eta)) E[x_r^T P M_k P x_r] \\
&\leq \frac{32\eta^2 K m}{\alpha_1(\eta) - 4\beta(\eta)} \cdot \left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right]^{2m} \cdot \sum_{k=2}^{d} E[(u_k^T x_0)^2].
\end{aligned} \tag{4.61}
$$

Using (4.60), (4.61) for Lemma 4.3.5, we finally have

$$
\begin{aligned}
\sum_{k=2}^{d} E[(u_k^T x_m)^2] \leq \sum_{k=2}^{d} E[(u_k^T x_0)^2] \cdot \Bigg[& p_m(\alpha_2(\eta), \beta(\eta)) \\
& + \frac{32\eta^2 K m}{\alpha_1(\eta) - 4\beta(\eta)} \cdot \left(\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right)^{2m} \Bigg].
\end{aligned} \tag{4.62}
$$

Lastly, using Lemma 4.3.5 for $k = 1$, we have

$$
E[(u_1^T x_m)^2] = p_m(\alpha_1(\eta), \beta(\eta)) E[(u_1^T x_0)^2] + 4\eta^2 \sum_{r=1}^{m-1} q_{m-r-1}(\alpha_1(\eta), \beta(\eta)) E[x_r^T P M_1 P x_r].
$$

Since PM_kP is positive semi-definite and $q_t(\alpha_1(\eta), \beta(\eta)) \geq 0$ for $1 \leq t < m$ by (A.53) in Lemma A.2.4, we have

$$
E[(u_1^T x_m)^2] \geq p_m(\alpha_1(\eta), \beta(\eta)) E[(u_1^T x_0)^2]. \tag{4.63}
$$

Also, from $\alpha_1(\eta) > \alpha_2(\eta) = 4\beta(\eta)$ and (A.52) in Lemma A.2.4, we have

$$
p_m(\alpha_1(\eta), \beta(\eta)) \geq \frac{1}{4} \left(\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta) - 4\beta(\eta)}}{2} \right)^{2m}. \tag{4.64}
$$

Using (4.62), (4.63) and (4.64), we eventually obtain

$$\frac{\sum_{k=2}^{d} E[(u_k^T x_m)^2]}{E[(u_1^T x_m)^2} \leq \left[\frac{p_m(\alpha_2(\eta), \beta(\eta))}{p_m(\alpha_1(\eta), \beta(\eta))} + \frac{128\eta^2 Km}{\alpha_1(\eta) - 4\beta(\eta)}\right] \cdot \frac{\sum_{k=2}^{d} E[(u_k^T x_0)^2]}{E[(u_1^T x_0)^2},$$

which completes the proof. □

Lemma 4.3.6 provides a bound for $\sum_{k=2}^{d} E[x_t^T P M_k P x_t]$. Note that it depends on Δ and blows up as Δ goes to zero due to the term involving $1/(\alpha_1(\eta) - 4\beta(\eta))$. Due to this dependency, VR HB Power tends to require a larger batch size than VR Power given the same values of η and m. Lemma 4.3.6 also establishes a bound for θ_m as a function of θ_0, η, m and K under some assumption.

Lemma 4.3.7. *For some $\mu \geq 0$, let $\eta = \Delta^\mu$ and*

$$m = \left\lceil \left(\frac{1 - \eta + \eta\lambda_1}{\eta\lambda_1\Delta + \sqrt{\eta\lambda_1\Delta(2(1-\eta) + \eta(\lambda_1 + \lambda_2))}} + \frac{\sqrt{\eta\lambda_1\Delta(2(1-\eta) + \eta(\lambda_1 + \lambda_2))}}{\eta\lambda_1\Delta + \sqrt{\eta\lambda_1\Delta(2(1-\eta) + \eta(\lambda_1 + \lambda_2))}} \right) \frac{\log 8}{2} \right\rceil \tag{4.65}$$

and

$$|S| \geq \frac{128\eta\sigma^2 m}{\lambda_1\Delta\left[2(1-\eta) + \eta(\lambda_1 + \lambda_2)\right]}. \tag{4.66}$$

Then, we have $\theta_m \leq \frac{3}{4} \cdot \theta_0$.

Proof. Using the conditions on m and $|S|$, we have

$$0 \leq \frac{4\eta^2 Km}{\alpha_1(\eta) - 4\beta(\eta)} \leq \frac{1}{128}. \tag{4.67}$$

Also, from

$$p_m(\alpha_2(\eta),\beta(\eta)) = (\beta(\eta))^m, \quad p_m(\alpha_1(\eta),\beta(\eta)) \geq \frac{1}{4}\left[\frac{\sqrt{\alpha_1(\eta)}}{2} + \frac{\sqrt{\alpha_1(\eta)-4\beta(\eta)}}{2}\right]^{2m}$$

and the choice of and m, we have

$$\begin{aligned}
\frac{p_m(\alpha_2(\eta),\beta(\eta))}{p_m(\alpha_1(\eta),\beta(\eta))} &\leq 4\left[\frac{\sqrt{4\beta(\eta)}}{\sqrt{\alpha_1(\eta)}+\sqrt{\alpha_1(\eta)-4\beta(\eta)}}\right]^{2m} \\
&= 4\left[1-\frac{\sqrt{\alpha_1(\eta)}-\sqrt{4\beta(\eta)}+\sqrt{\alpha_1(\eta)-4\beta(\eta)}}{\sqrt{\alpha_1(\eta)}+\sqrt{\alpha_1(\eta)-4\beta(\eta)}}\right]^{2m} \\
&= 4\left(1-\frac{\eta\lambda_1\Delta+\sqrt{\eta\lambda_1\Delta(2(1-\eta)+\eta(\lambda_1+\lambda_2))}}{1-\eta+\eta\lambda_1+\sqrt{\eta\lambda_1\Delta(2(1-\eta)+\eta(\lambda_1+\lambda_2))}}\right)^{2m} \\
&\leq 4\exp\left[-2\frac{\eta\lambda_1\Delta+\sqrt{\eta\lambda_1\Delta(2(1-\eta)+\eta(\lambda_1+\lambda_2))}}{1-\eta+\eta\lambda_1+\sqrt{\eta\lambda_1\Delta(2(1-\eta)+\eta(\lambda_1+\lambda_2))}}m\right] \\
&\leq \frac{1}{2}.
\end{aligned} \tag{4.68}$$

Therefore, using (4.67) and (4.68) in Lemma 4.3.6, we finally have

$$\begin{aligned}
\frac{\sum_{k=2}^d E[(u_k^T x_m)^2]}{E[(u_1^T x_m)^2]} &\leq \left[\frac{p_m(\alpha_2(\eta),\beta(\eta))}{p_m(\alpha_1(\eta),\beta(\eta))} + \frac{128\eta^2 Km}{\alpha_1(\eta)-4\beta(\eta)}\right]\cdot\frac{\sum_{k=2}^d E[(u_k^T x_0)^2]}{E[(u_1^T x_0)^2]} \\
&\leq \frac{3}{4}\cdot\frac{\sum_{k=2}^d E[(u_k^T x_0)^2]}{E[(u_1^T x_0)^2]},
\end{aligned}$$

which completes the proof. □

Lemma 4.3.7 provides explicit conditions for m and $|S|$ to ensure a sufficient decrease of θ_m. Note that when $\mu = 0$, we have $|S| \geq \mathcal{O}(\frac{1}{\Delta^{3/2}})$, which improves the analysis of VR Power+M in [**79**] by removing the dependency on $\sqrt{d}$. Also, for any $|S| \geq 1$, there exists some η and m satisfying the conditions in Lemma 4.3.7. This implies that VR HB

Power works with any batch size while VR Power+M does not. The overall convergence is established next.

Theorem 4.3.8. *Suppose that an initial vector $\tilde{x}_0$ satisfies $u_1^T \tilde{x}_0 \neq 0$ and let $\tilde{\theta}_0 = (1 - (u_1^T \tilde{x}_0)^2)/(u_1^T \tilde{x}_0)^2 \geq \epsilon$ for some $\epsilon > 0$. If $\eta = \Delta^\mu$ and m and $|S|$ satisfy (4.65) and (4.66), after $\tau = \lceil \log(\tilde{\theta}_0/\epsilon)/\log(4/3) \rceil$ epochs of VR HB Power, we have $\tilde{\theta}_\tau \leq \epsilon$.*

Proof of Theorem 4.3.8. By repeatedly applying Lemma 4.3.7, we have

$$\frac{\sum_{k=2}^d E[(u_k^T \tilde{x}_\tau)^2]}{E[(u_1^T \tilde{x}_\tau)^2]} \leq \left(\frac{3}{4}\right)^\tau \frac{\sum_{k=2}^d E[(u_k^T \tilde{x}_0)^2]}{E[(u_1^T \tilde{x}_0)^2]} = \left(\frac{3}{4}\right)^\tau \tilde{\theta}_0.$$

Since $\tau = \lceil \log(\tilde{\theta}_0/\epsilon)/\log(4/3) \rceil$, we have

$$\tau \log\left(\frac{3}{4}\right) \leq \log\left(\frac{\epsilon}{\tilde{\theta}_0}\right),$$

resulting in

$$\frac{\sum_{k=2}^d E[(u_k^T \tilde{x}_\tau)^2]}{E[(u_1^T \tilde{x}_\tau)^2]} \leq \epsilon.$$

□

The global convergence result in Theorem 4.3.8 is based on the single epoch result in Lemma 4.3.7. Since $\tau = \mathcal{O}\big(\log(\frac{1}{\epsilon})\big)$, the iteration complexity is $\tau m = \mathcal{O}\big(\frac{1}{\Delta^{1/2+\mu/2}} \log(\frac{1}{\epsilon})\big)$. On the other hand, from $|S| = \mathcal{O}\big(\frac{1}{\Delta^{3/2-\mu/2}}\big)$, the sample complexity amounts to $\mathcal{O}\big((n + \frac{1}{\Delta^2}) \log(\frac{1}{\epsilon})\big)$. Note that VR HB Power has the same sample complexity as VR Power but may have small iteration complexity. Therefore, if per sample cost is cheaper than per iteration cost, VR HB Power can be more efficient than VR Power.

4.4. Practical Considerations

In this section, we discuss some practical aspects implementing the proposed algorithms. First, to ensure that the algorithms are numerically stable, we consider normalizations as introduced in [**72**] and [**79**]. After updating x_{t+1}, we normalize x_{t+1} as $x_{t+1} \leftarrow x_{t+1}/\|x_{t+1}\|_2$ in VR Power and update x_t and x_{t+1} as $x_t \leftarrow x_t/\|x_{t+1}\|_2$ and $x_{t+1} \leftarrow x_{t+1}/\|x_{t+1}\|_2$ in VR HB Power. Since these scaling schemes do not impact the sample paths of $x_t/\|x_t\|$, we can obtain the same results with numerical stability.

Another practical issue with the implementations of VR Power and VR HB Power is to estimate λ_1 and λ_2. As appearing in Lemma 4.3.3 and Lemma 4.3.7, accurate values of λ_1 and λ_2 are essential to determine the values of η, m, and β (for VR HB Power). In the experiments, the mini-batch size $|S|$ is given as some percentage of n, so no estimation is required for $|S|$. In order to estimate λ_1 and λ_2 at a regular interval (at the start of each inner-loop), we use the exact gradients of two consecutive outer-loop iterates $\tilde{x}_{s-1}$ and $\tilde{x}_s$. Since we expect that $\tilde{x}_s$ approaches u_1 as the iterations advance, using the Rayleigh quotient, we estimate λ_1 as

$$\hat{\lambda}_1 = \frac{(\tilde{x}_s)^T C(\tilde{x}_s)}{(\tilde{x}_s)^T \tilde{x}_s}. \tag{4.69}$$

To estimate λ_2 in the same way, we need an estimate of u_2. In Power iteration, an iterate first approaches the subspace spanned by u_1 and u_2 before converging to u_1. That being said, after a number of iterations, we can approximate it by a linear combination of u_1 and u_2. Based on this observation, we estimate u_2 as

$$\hat{u}_2 = \tilde{x}_{s-1} - (\tilde{x}_{s-1}^T \tilde{x}_s)\tilde{x}_s. \tag{4.70}$$

The idea of the above estimation is to project $\tilde{x}_{s-1}$ to the space orthogonal to $\tilde{x}_s$. If $\tilde{x}_s \approx u_1$ and $\tilde{x}_{s-1} \approx \alpha_1 u_1 + \alpha_2 u_2$ for some $\alpha_1, \alpha_2 (\neq 0)$, we have $\hat{u}_2 \approx u_2$. Using the Rayleigh quotient of $\hat{u}_2$, we estimate λ_2 as

$$\hat{\lambda}_2 = \frac{\tilde{x}_{s-1}^T C \tilde{x}_{s-1} - 2\theta_s \tilde{x}_s^T C \tilde{x}_{s-1} + \theta_s^2 \tilde{x}_s^T C \tilde{x}_s}{1 - \theta_s^2} \tag{4.71}$$

where $\theta_s = \tilde{x}_{s-1}^T \tilde{x}_s$. While two matrix-vector multiplications, $C\tilde{x}_{s-1}$ and $C\tilde{x}_s$, are involved in computing (4.69) and (4.71), they incur no extra computation since they are the exact gradients of $\tilde{x}_{s-1}$ and $\tilde{x}_s$, which are computed regardless of the estimation. As a result, we can obtain $\hat{\lambda}_1$ and $\hat{\lambda}_2$ by only computing some inner products. For initial estimation of $\hat{\lambda}_1$ and $\hat{\lambda}_2$, we run Power iteration five times and use the last two iterates. Note that the exact gradient of the last iterate is computed at the start of the very first outer-loop iteration.

Given $|S|$ and estimates of λ_1 and λ_2, we use bisection search to find $\eta \in (0, 1]$ such that the terms on the right-hand sides of (4.33) and (4.66) are almost equal to $|S|$. After η is found, we use (4.32) and (4.65) to determine m.

4.5. Numerical Experiments

In this section, we test the performance of VR Power and VR HB Power with that of (i) VR-PCA [**72**], (ii) VR Power+M [**79**] and (iii) Fast PCA [**22**] for finding the first eigenvector u_1 of the covariance matrix C constructed by data vectors $a_i, i = 1, \ldots, n$ from real world datasets. Note that all present stochastic variance-reduced PCA algorithms are compared in this experiment.

4.5.1. Datasets

The datasets include ijcnn [**68**], covertype [**8**], YearPredictionMSD [**7**] and MNIST [**44**] as summarized in Tabel 4.2. All of them are obtained either from the UCI repository [**18**] or the LIBSVM library [**17**]. They are carefully chosen to incorporate a variety of datasets in terms of size and eigen-gap. The first three datasets are standardized with a mean of

Table 4.2. Summary of datasets for PCA

DATASET	n	d	Δ
ICJNN(TEST)	91,701	22	0.0079
COV	581,012	54	0.2106
MSD	463,715	90	0.3224
MNIST	70,000	764	0.8851

zero and standard deviation of one while the last one is scaled to the range between 0 and 1 to preserve its sparsity.

4.5.2. Settings

In order to report a comprehensive comparison of the algorithms, we consider two settings for selecting hyper-parameters. In the first setting, we use hyper-parameter tuning. Specifically, we use a grid search to find the best values of η, m and $|S| = \rho\%$ of each algorithm and dataset where $\eta \in \{0.01, 0.05, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0\}$, $m \in \{25, 50, 100, 200\}$ and $\rho \in \{1, 2, 5, 10\}$.

In the second setting, we use the following theoretically derived or recommended hyper-parameter values.

- VR-PCA:

$$\eta = \frac{\sqrt{n}}{\sum_{i=1}^{n} \|a_i\|^2}, \quad m = n, \quad |S| = 1.$$

- VR Power+M:

$$\beta = \frac{\lambda_2^2}{4},\ \sigma^2 = \frac{\sum_{i=1}^{n} \|a_i\|^2}{n},\ |S| = \frac{\lambda_2 \log 16}{\sqrt{\lambda_1^2 - \lambda_2^2}},\ T = \frac{512 \log 16 \lambda_2 \sigma^2 \sqrt{d}}{\sqrt{\lambda_1^2 - \lambda_2^2}}.$$

- Fast PCA: $\delta = \lambda_1 - \lambda_2$. We only consider the accurate regime. In order to solve each problem, we use SVRG [**33**] with $\tilde{\epsilon} = 10^{-3}$,

$$\eta = \frac{\lambda_1 - \lambda_2}{7(2\lambda_1 + \lambda_2)^2}, \quad m = \left\lceil \frac{1}{2\eta^2(2\lambda_1 + \lambda_2)^2} \right\rceil.$$

- VR Power, VR HB Power: $|S| = \rho\% \cdot n$ for $\rho \in \{1, 2\}$ and $\sigma^2 = \sum_{i=1}^{n} \|a_i\|^2/n$. For η and m, we use bisection search explained in Section 4.4. Also, the scaling schemes in Section 4.4 are used to ensure numerical stability. The exact values of λ_1 and λ_2 are used to find η and m.
- PF VR Power, PF VR HB Power: As opposed to VR Power and VR HB Power, adaptive estimates of $\hat{\lambda}_1$ and $\hat{\lambda}_2$ obtained by the procedure in Section 4.4 are used to find η and m.

4.5.3. Results

Figure 4.1 displays the experimental result with hyper-parameter tuning. In the figure, the x-axis represents time in seconds and the y-axis represents the optimality gap, $1 - (\tilde{x}_s^T u_1)^2$, in the log-scale. Since VR-PCA and VR Power are related algorithms, their performances are similar except for cov where the step size of VR-PCA is tuned to the largest possible

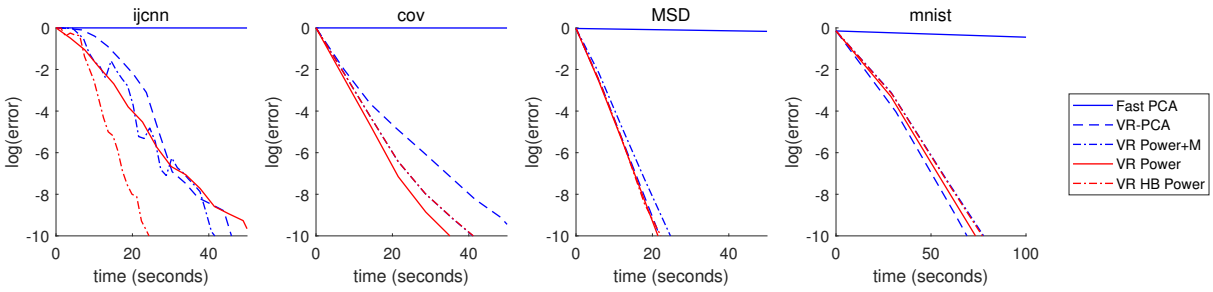

Figure 4.1. Convergence plots of stochastic variance-reduced PCA algorithms with hyper-parameters tuned

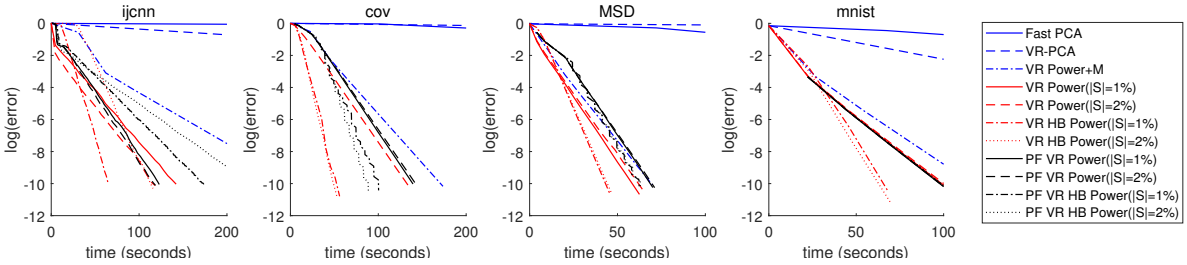

Figure 4.2. Convergence plots of stochastic variance-reduced PCA algorithms with recommended hyper-parameters and parameter-free algorithms

value of 1.0. If some larger values are included in the grid, VR-PCA would have a similar performance to VR Power even for cov. On the other hand, VR HB Power always performs better than VR Power+M due to its additional control through the step size. VR HB Power works particularly well for ijcnn which has the smallest eigen-gap. If the eigen-gap is large, the performance of VR HB Power is not much different from the performances of VR Power+M, VR-PCA and VR Power. We were not able to find good hyperparameters for Fast PCA.

Figure 4.2 shows the experimental result without parameter tuning. In the figure, regardless of the batch size, VR Power and VR HB Power outperform VR-PCA, VR Power+M and Fast PCA. Although VR Power and VR-PCA are similar algorithms, the performance of VR Power is much better than that of VR-PCA due to the choice of η

and m. While VR Power precisely choose the values of η and m depending on the values of λ_1, λ_2 and $|S|$, VR-PCA does not utilize such information and let them depend only on n. As a result, the step size is too small and the epoch length is too large, leading to slow convergence. On the other hand, due to the extra dependency on $\sqrt{d}$, VR Power+M requires too large samples and thus it is slower than VR Power even for ijcnn which has the smallest eigen-gap. The epoch length m of SVRG in Fast PCA is of the order of $1/\Delta^2$. Therefore, Fast PCA takes a significant amount of time to solve each convex problem, which makes its optimality gap not decrease as sharply as other algorithms. On the other hand, PF VR HB Power takes longer than VR HB Power while the performance of PF VR Power looks very similar to that of VR Power. This is because VR HB Power has the additional momentum parameter β, which makes its performance more affected by estimation errors. Nevertheless, both parameter-free algorithms work very well compared to VR-PCA, VR Power+M and Fast PCA.

4.6. Final Remarks

In this chapter, we present two mini-batch stochastic variance-reduced algorithms for PCA and derive exact forms of their parameters to attain the optimal runtime. For any batch size, the result shows that the optimal runtime can be achieved by appropriately choosing the step size and epoch length. We also introduce practical implementations which automatically find such values depending on batch sizes. The framework used in our analysis is not specific to the proposed algorithms but can be applied to analyze other stochastic variance-reduced PCA algorithms and improve their results. In our framework, the optimality gap is measured as the ratio of two expectation terms and this enables us

to develop global convergence statements. Experimental results show that the proposed algorithms work well for arbitrary batch sizes.

CHAPTER 5

Stochastic Scale Invariant Power Iteration

5.1. Introduction

We consider scale invariant problems with finite-sum objective functions of the form

$$\max_{x} \quad f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) \quad \text{subject to} \quad x \in \partial\mathcal{B}_d \tag{5.1}$$

where f_i are scale invariant functions of the same type, i.e. f_i are either multiplicatively scale invariant with the multiplicative factor $u(c) = |c|^p$ such that $f_i(cx) = u(c)f_i(x)$ or additively scale invariant satisfying $f_i(cx) = f_i(x) + v(c)$ with the additive factor $v(c) = \log_a |c|$. It covers interesting problems in machine learning and statistics such as L_p-norm kernel PCA [**39**] and the estimation of mixture proportions [**41**], and three extended settings cover more interesting problems such as independent component analysis (ICA) [**29**,**30**], Gaussian mixture models (GMM), Kullback-Leibler divergence non-negative matrix factorization (KL-NMF) [**21**, **45**, **76**] and the Burer-Monteiro factorization of semi-definite programs [**20**].

Assuming that f is twice differentiable on an open set containing $\partial\mathcal{B}_d$, the scale invariant problem (5.1) can be locally viewed as a leading eigenvector problem in the sense that a stationary point x^* is an eigenvector of $\nabla^2 f(x^*)$. If the Lagrange multiplier λ^* satisfying $\lambda^* x^* = \nabla f(x^*)$ is greater than the absolute values of eigenvalues of $\nabla^2 f(x^*)(I - x^*(x^*)^T)$, a stationary point x^* is a local maximum to (5.1). Due to this eigenvector property, the

scale invariant problem can be efficiently solved by a general form of power iteration called *scale invariant power iteration (SCI-PI)* [**38**] specified by

$$x_{k+1} \leftarrow \frac{\nabla f(x_k)}{\|\nabla f(x_k)\|_2}. \tag{5.2}$$

The convergence behavior of (5.2) generalizes that of power iteration. If x_0 is initialized close to a local optimum x^*, the optimality gap $1-(x_k^T x^*)^2$ linearly converges at an asymptotic rate of $(\bar{\lambda}/\lambda^*)^2$ where $\bar{\lambda}$ is the largest absolute value of eigenvalues of $\nabla^2 f(x^*)(I-x^*(x^*)^T)$. The convergence rate specializes to $(\lambda_1/\lambda_2)^2$ in the case of the PCA problem [**34**] where λ_1 and λ_2 are the first and the second eigenvalues of the covariance matrix $\frac{1}{n}\sum_{i=1}^n a_i a_i^T$ constructed by data vectors a_i. This result is consistent with the analysis of power iteration in [**26**].

Due to the analogy between power iteration for the leading eigenvector problem and gradient descent for convex optimization, many advanced algorithms have been developed for power iteration, including noisy [**27**], coordinate-wise [**46**], momentum [**79**], online [**9**, **24**] and stochastic [**40**, **63**, **72**, **73**, **79**] algorithms. In particular, based on the stochastic variance-reduced gradient technique [**33**], stochastic variance-reduced power iteration [**40**] reduces the total runtime to obtain an ϵ-optimal solution from $\mathcal{O}\big(\frac{dn}{\Delta}\log\frac{1}{\epsilon}\big)$ to $\mathcal{O}\big(d\big(n+\frac{1}{\Delta^2}\big)\log\frac{1}{\epsilon}\big)$ where $\Delta = 1-\lambda_2/\lambda_1$ represents the eigen-gap. This decoupling of the data size n from the eigen-gap Δ is significant in a large scale setting where n is large. More importantly, the developement of stochastic algorithms for power iteration opens up the possibility to develop stochastic algorithms for constrained machine learning models.

In this work, we develop a stochastic variance-reduced algorithm to solve (5.1). To solve this general constrained problem in a stochastic manner, we introduce a stochastic

variance-reduced algorithm of SCI-PI [**38**] (S-SCI-PI) and provide its convergence analysis. The update formula of S-SCI-PI generalizes that of VR Power [**40**] but the scaling factor for a full-gradient is not simply the dot product of an inner iterate and an outer iterate but a homogeneous function of it. In the convergence analysis of S-SCI-PI, we derive a bound on the expectation of the optimality gap. Developing this bound is not trivial since two types of errors are involved. The first one is attributed to the difference of the Hessians between the iterate and the optimal solution. To control this error, we derive a condition on initial iterate, step size and batch size, so that the error is not increasing in the course of the algorithm. On the other hand, the second error occurs from stochastic sampling of gradient. Using recursion, we compute a bound of the variance part of the expected optimality gap and develop a compact decomposition of the expectation of the optimality gap. We show that the expectation of the optimality gap converges at a linear rate under some conditions on the initial iterate, the step size, the epoch length and the batch size.

We apply S-SCI-PI to the KL-NMF problem. The KL-NMF subproblem is equivalent to the estimation of mixture proportions problem, which can be reformulated to a scale invariant problem. Using the separable structure of the KL-NMF subproblems, we reformulate the KL-NMF problem as a two-block scale invariant problem [**38**] and alternatively apply S-SCI-PI to optimize two non-negative matrices. Experiments on synthetic and real datasets reveal that the proposed stochastic approach not only converges faster than state-of-the-art deterministic algorithms but also produces robust solutions under random initialization.

This work has the following contributions.

(1) We propose a stochastic algorithm (S-SCI-PI) for solving the finite-sum scale invariant problem. The algorithm adapts the stochastic variance-reduced gradient technique and adjusts the scaling factor of full-gradients depending on the order of scale invariance.

(2) We provide a convergence analysis for S-SCI-PI. Deriving compact representations of error terms, we prove linear convergence of S-SCI-PI where we show that the expected optimality gap decreases at a linear rate under some conditions on the initial iterate, epoch length, batch size and an additional condition.

(3) We provide experiments to show that SCI-PI converges faster than the state-of-the-art deterministic algorithms for KL-NMF and its subproblem.

The paper is organized as follows. We present the algorithm in Section 5.2 and provide the convergence analysis in Section 5.3. We introduce the KL-NMF problem and its reformulation to two-block scale invariant problem in Section 5.4 and discuss some implementation issues in Section 5.4.2. The experimental results on real and synthetic datasets are followed in Section 5.5.

5.2. Algorithm

Before presenting the algorithm, we introduce some notation used throughout the paper. For the scale invariant objective function f in (5.1), we let p be the degree of scale invariance. Let $\nabla_k f(x)$ be the k-th coordinate of the gradient ∇f. For a mini-batch sample $S \subset [n] := \{1, 2, \cdots, n\}$, we define a stochastic function $f_S = \sum_{l \in S} f_l / |S|$. We present the scale invariant multiplicative case and point out the changes needed for the additive case later.

The algorithm has the two-loop structure similar to the stochastic variance-reduced gradient (SVRG) method [**33**]. Before the start of each inner-loop, we compute the full gradient $\tilde{g}_s$ at the outer iterate $\tilde{x}_s$, and utilize this gradient information to construct a stochastic variance-reduced gradient g_t at the inner iterate x_t. In order to derive a stochastic variance-reduced gradient at x_t utilizing the full gradient at $\tilde{x}_s$, we decompose x_t as

$$x_t = \frac{x_t^T \tilde{x}_s}{\|\tilde{x}_s\|^2} \tilde{x}_s + x_t - \frac{x_t^T \tilde{x}_s}{\|\tilde{x}_s\|^2} \tilde{x}_s.$$

In the above equation, the first component is the projection of x_t onto $\tilde{x}_s$ while the second part represents the orthogonal component of x_t with respect to $\tilde{x}_s$. Since ∇f is scale invariant with degree $p-1$ by Proposition 3.2.3, using $\tilde{g}_s$, we can compute the exact gradient at the first component as

$$\nabla f \left(\frac{x_t^T \tilde{x}_s}{\|\tilde{x}_s\|^2} \, \tilde{x}_s \right) = \frac{|x_t^T \tilde{x}_s|^{p-1}}{\|\tilde{x}_s\|^{2(p-1)}} \nabla f(\tilde{x}_s) = \alpha_t \tilde{g}_s \tag{5.3}$$

where $\alpha_t = |x_t^T \tilde{x}_s|^{p-1} / \|\tilde{x}_s\|^{2(p-1)}$. To approximate the difference of gradients at x_t and $(x_t^T \tilde{x}_s)\tilde{x}_s / \|\tilde{x}_s\|^2$, we use stochastic sample $S_t \subset [n]$ as

$$\frac{1}{|S_t|} \sum_{l \in S_t} \left(\nabla f_l(x_t) - \alpha_t \nabla f_l(x_0) \right). \tag{5.4}$$

Using (5.3) and (5.4), we obtain a stochastic variance-reduced gradient g_t at x_t as

$$g_t = \alpha_t \tilde{g}_s + \frac{1}{|S_t|} \sum_{l \in S_t} \left(\nabla f_l(x_t) - \alpha_t \nabla f_l(x_0) \right).$$

To control the progress of the algorithm depending on the variance of g_t, we introduce a step size $\eta \in (0, 1]$. Using the step size η, we derive the following update rule

$$x_{t+1} \leftarrow (1 - \eta)x_t + \eta \frac{g_t}{\|x_t\|^{p-2}}.$$

Note that we further scale g_t to match its scale with x_t. We divide g_t by $\|x_t\|^{p-2}$ since $\nabla f(x) = \nabla^2 f(x)x$ and $\nabla^2 f(x)$ is scale invariant with degree $p - 2$ by Proposition 3.2.3.

Summarizing all the above, we obtain Algorithm 5.

Algorithm 5 Stochastic SCI-PI (S-SCI-PI)

Parameter: step size $\eta \in (0, 1]$, batch size $|S|$
Randomly initialize outer iterate $\tilde{x}_0 \in \partial\mathcal{B}_d$
for $s = 0, 1, \ldots$ **do**
  $x_0 \leftarrow \tilde{x}_s$
  $\tilde{g}_s \leftarrow \nabla f(x_0)$
  for $t = 0, 1, \ldots, m - 1$ **do**
    $\alpha_t \leftarrow \frac{|x_t^T x_0|^{p-1}}{\|x_0\|^{2(p-1)}}$
    Sample $S_t \subset [n]$ of size $|S|$ uniformly at random
    $g_t \leftarrow \alpha_t \tilde{g}_s + \frac{1}{|S_t|} \sum_{l \in S_t} (\nabla f_l(x_t) - \alpha_t \nabla f_l(x_0))$
    $x_{t+1} \leftarrow (1 - \eta)x_t + \eta \frac{g_t}{\|x_t\|^{p-2}}$
  end for
  $\tilde{x}_{s+1} \leftarrow x_m$
end for

In the additive scale invariant case, the algorithm remains the same except that we set $p = 0$. The analyses remains the same since we only use the property $\nabla^2 f(x)x = (p - 1)\nabla f(x)$ for multiplicity while this expression for additive reads $\nabla^2 f(x)x = -\nabla f(x)$. These expressions are provided in Proposition 3.2.3. Thus $p = 0$ in the multiplicative case yields the additive case.

5.3. Convergence Analysis

For the analysis of the algorithm, we assume that every f_i is twice continuously differentiable on an open set containing $\mathcal{B}_{d,\infty}$. Let x^* be a local optimal solution satisfying $\nabla f(x^*) = \lambda^* x^*$, (λ_i, v_i) be an eigen-pair of $\nabla^2 f(x^*)$ and $\sigma = \|\nabla^2 f(x^*)\|$. Due to the eigenvector property of the scale invariant problem, x^* is an eigenvector of $\nabla^2 f(x^*)$. Without loss of generality, we let $x^* = v_1$. Since x^* is a local maximum, by Proposition 3.2.4, we have $\lambda^* > \overline{\lambda} = \max_{2\leq i\leq d}|\lambda_i|$.

Let H_i be the Hessian of $\nabla_i f$ and $F_i(y^1, \cdots, y^d) = (\lambda^* - \lambda_1)\mathbb{1}_{i=1} I + \sum_{j=1}^d v_{ij} H_j(y^j)$. Also, we let $G_S(y^1, \cdots, y^d)$ be the matrix such that $\nabla\nabla_j g_S(y^j)^T$ is the j^{th} row of $G_S(y^1, \cdots, y^d)$ where $g_S = f_S - f$.

Next, we introduce some constants that are used to derive bounds in the analysis. First, let M be a constant such that

$$M = \max_{x\in\mathcal{B}_d,\, y^1,\cdots,y^d\in\mathcal{B}_{d,\infty}} \sqrt{\textstyle\sum_{i=1}^d (x^T F_i(y^1, \cdots, y^d)x)^2}. \tag{5.5}$$

This constant measures local smoothness of the objective function f near the local optimal solution x^*. We define quantities K and L as

$$K = \max_{y^1,\cdots,y^d\in\mathcal{B}_\infty} E_S\left[\|G_S(y^1, \cdots, y^d)\|^2\right], \quad L = \max_{y^1,\cdots,y^d\in\mathcal{B}_\infty} \|G_S(y^1, \cdots, y^d)\|^2 \tag{5.6}$$

and let L_0 be an upper bound of L which we obtain by setting $|S| = 1$. K and L measure deviation of f_S from its mean f with respect to stochastic sample S of size $|S|$. K measures the mean squared deviation (variance) of f_S and L is concerned with the maximum squared deviation of f_S from f. As the batch size $|S|$ is increasing, both K and L are decreasing,

and both of them become zero when $|S| = n$. While K decreases as a factor of $1/|S|$, L is a non-trivial function of $|S|$. Therefore, if some f_i is extremely irregular (i.e. $|f_i - f|$ has an extremely large value around the solution), we would have to use a batch size close to n to ensure that L is smaller than some level.

Next we present the convergence analysis for S-SCI-PI. We first analyze a single inner iteration which computes x_{t+1} from x_t. Let

$$\alpha(\eta) = 1 - \eta + \eta\lambda^*, \quad \beta(\eta) = 1 - \eta + \eta\bar{\lambda}, \quad y_k = \frac{x_k}{\|x_k\|}, \quad \Delta_t = 1 - y_t^T x^*.$$

Since the optimality gap is expressed as $\sum_{i=2}^{d}(x_t^T v_k)^2/(x_t^T v_1)^2$, it is important to analyze how $x_t^T v_k$ changes after each iteration. The following lemma provides an expression of $x_{t+1}^T v_k$ as a sum of three components.

Lemma 5.3.1. *For $1 \le k \le d$ and any t, if $x_t^T x_0 \ge 0$, then we have*

$$\begin{aligned} x_{t+1}^T v_k = &\left(1 - \eta + \eta(\lambda_k + (\lambda^* - \lambda_1)\mathbb{1}_{k=1})\right) x_t^T v_k \\ &+ \frac{1}{2}\eta\|x_t\|(y_t - x^*)^T F_k(\hat{y}_t^1, \cdots, \hat{y}_t^d)(y_t - x^*) \\ &+ \eta\left(G_{S_t}(\bar{y}_t^1, \cdots, \bar{y}_t^d)\big(x_t - (x_t^T y_0)y_0\big)\right)^T v_k. \end{aligned}$$

Proof. From the update rule in Algorithm 5, we have

$$\begin{aligned} x_{t+1} &= (1-\eta)x_t + \frac{\eta}{\|x_t\|^{p-2}}\left(\nabla f_{S_t}(x_t) - \alpha_t \nabla f_{S_t}(y_0) + \alpha_t \tilde{g}\right) \\ &= (1-\eta)x_t + \frac{\eta}{\|x_t\|^{p-2}}\nabla f(x_t) \\ &\quad + \frac{\eta}{\|x_t\|^{p-2}}\left[\nabla f_{S_t}(x_t) - \nabla f(x_t) - \alpha_t\left(\nabla f_{S_t}(y_0) - \nabla f(y_0)\right)\right]. \end{aligned} \tag{5.7}$$

Since $\nabla_i f$ is twice continuously differentiable on an open set containing $\partial\mathcal{B}_d$, using the Taylor theorem, we obtain

$$\nabla_i f(y_t) = \nabla_i f(x^*) + \nabla\nabla_i f(x^*)(y_t - x^*) + \frac{1}{2}(y_t - x^*)^T H_i(\hat{y}_t^i)(y_t - x^*) \tag{5.8}$$

where $\hat{y}_t^i \in \mathcal{N}(y_t, x^*) \triangleq \{z : z_j = \mu_j x_j^* + (1-\mu_j)y_{tj},\ 0 \le \mu_j \le 1,\ j \in [d]\}$. We let the j^{th} coordinates of z, x^*, y_t, v_k as z_j, x_j^*, y_{tj}, v_{kj}, respectively. Since f is scale invariant with the degree of p, by Proposition 3.2.3, ∇f is scale invariant with the degree of $p-1$, leading to

(5.9)
$$\frac{\nabla f(x_t)^T v_k}{\|x_t\|^{p-1}} = \nabla f(x^*)^T v_k + (y_t - x^*)^T \nabla^2 f(x^*) v_k + \frac{1}{2}(y_t - x^*)^T \sum_{i=1}^{d} v_{ki} H_i(\hat{y}_t^i)(y_t - x^*).$$

For $k = 1$, using $v_1 = x^*$, we have

$$\nabla f(x^*)^T v_1 = \lambda^*, \quad (y_t - x^*)^T \nabla^2 f(x^*) v_1 = (y_t - x^*)^T \nabla^2 f(x^*) x^* = \lambda_1 (y_t^T x^* - 1),$$

which results in

$$\begin{aligned}
\frac{\nabla f(x_t)^T v_1}{\|x_t\|^{p-1}} &= \lambda^* - \lambda_1(1 - y_t^T x^*) + \frac{1}{2}(y_t - x^*)^T \sum_{i=1}^{d} v_{1i} H_i(\hat{y}_t^i)(y_t - x^*) \\
&= \lambda^* y_t^T x^* + (\lambda^* - \lambda_1)(1 - y_t^T x^*) + \frac{1}{2}(y_t - x^*)^T \sum_{i=1}^{d} v_{1i} H_i(\hat{y}_t^i)(y_t - x^*) \\
&= \lambda^* y_t^T x^* + \frac{1}{2}(y_t - x^*)^T \big[(\lambda^* - \lambda_1) I + \sum_{i=1}^{d} v_{1i} H_i(\hat{y}_t^i)\big](y_t - x^*) \\
&= \lambda^* y_t^T x^* + \frac{1}{2}(y_t - x^*)^T F_1(\hat{y}_t^1, \cdots, \hat{y}_t^d)(y_t - x^*)).
\end{aligned} \tag{5.10}$$

For $2 \leq k \leq d$, from (5.9), $(x^*)^T v_k = v_1^T v_k = 0$ and $\nabla f(x^*)^T v_k = \lambda^* v_1^T v_k = 0$, we have

$$\frac{\nabla f(x_t)^T v_k}{\|x_t\|^{p-1}} = \lambda_k y_t^T x^* + \frac{1}{2}(y_t - x^*)^T F_k(\hat{y}_t^1, \cdots, \hat{y}_t^d)(y_t - x^*). \tag{5.11}$$

Since ∇f_l is scale invariant with the degree of $p-1$ for each $l \in [n]$, we have

$$\nabla f_l(x_t) = \|x_t\|^{p-1} \nabla f_l(y_t), \quad \alpha_t \nabla f_l(y_0) = \|x_t\|^{p-1} (y_t^T y_0)^{p-1} \nabla f_l(y_0),$$

which leads to

$$\frac{1}{\|x_t\|^{p-1}} \left(\nabla f_{S_t}(x_t) - \nabla f(x_t) - \alpha_t \left(\nabla f_{S_t}(y_0) - \nabla f(y_0)\right)\right) = \nabla g_{S_t}(y_t) - \nabla g_{S_t}\big((y_t^T y_0) y_0\big).$$

Using the Taylor approximation of $\nabla_k g_{S_t}$ around $(y_t^T y_0) y_0$, we have

$$\nabla_k g_{S_t}(y_t) - \nabla_k g_{S_t}((y_t^T y_0) y_0) = \nabla \nabla_k g_{S_t}\big(\bar{y}_t^k\big)^T \big(y_t - (y_t^T y_0) y_0\big)$$

where $\bar{y}_t^k \in \mathcal{N}(y_t, (y_t^T y_0) y_0)$. This leads to

$$\frac{1}{\|x_t\|^{p-2}} \left(\nabla f_{S_t}(x_t) - \nabla f(x_t) - \alpha_t \left(\nabla f_{S_t}(y_0) - \nabla f(y_0)\right)\right) = G_{S_t}(\bar{y}_t^1, \cdots, \bar{y}_t^d)\big(x_t - (x_t^T y_0) y_0\big). \tag{5.12}$$

Using (5.7), (5.10), (5.11) and (5.12), we have

$$\begin{aligned} x_{t+1}^T v_k = {} & \left(1 - \eta + \eta(\lambda_k + (\lambda^* - \lambda_1)\mathbb{1}_{k=1})\right) x_t^T v_k + \frac{1}{2}\eta \|x_t\| (y_t - x^*)^T F_k(\hat{y}_t^1, \cdots, \hat{y}_t^d)(y_t - x^*) \\ & + \eta \left(G_{S_t}(\bar{y}_t^1, \cdots, \bar{y}_t^d)\big(x_t - (x_t^T y_0) y_0\big)\right)^T v_k. \end{aligned} \tag{5.13}$$

□

In Lemma 5.3.1, the first term represents the growth of $x_t^T v_k$. The multiplicative factor is $1-\eta+\eta\lambda^*$ if $k=1$ and $1-\eta+\eta\lambda_k$ otherwise. The second component is attributed to the difference of the Hessians at x_t and x^*. As x_t closes on x^*, this term goes to zero. The last term is the stochastic error. The stochastic error is affected by the batch size $|S|$ and how closely x_t is aligned with x_0 where we compute the full gradient. The following lemma provides a condition on η, L, M and x_0 to ensure that $y_t^T x^*$ is not smaller than $y_0^T x^*$ for every stochastic realization.

In the below, we frequently use the fact that for $0<\eta\leq 1$, $\eta\leq \max(1,\nu)^{-1}$ implies

$$n\nu \leq 1. \tag{5.14}$$

This can be easily proved by $\eta\nu \leq \max(1,\nu)^{-1}\nu \leq 1$ for $\nu\geq 0$ and $\eta\nu<0$ for $\nu<0$. Also, we often use that (5.16) implies

$$\frac{\sqrt{\Delta_0}}{1-\Delta_0}\leq 1, \quad \frac{1}{1-\Delta_0}\leq\sqrt{2}. \tag{5.15}$$

Lemma 5.3.2. *For any positive integer m, if the step size η, $|S|$ and x_0 are chosen to satisfy*

$$\Delta_0 \leq \min\left\{1-\frac{1}{\sqrt{2}}, \frac{(\lambda^*-\bar{\lambda})^2}{4(M+2\sqrt{L_0})^2}\right\} \tag{5.16}$$

and either one of the following conditions:

$$L\leq \frac{(\lambda^*-\bar{\lambda}-2M\sqrt{\Delta_0})^2}{32}, \tag{5.17}$$

$$\eta\leq \max(1,\nu_1,\nu_2,\nu_3)^{-1} \tag{5.18}$$

where

(5.19a)

$$\nu_1 = 1 - \lambda^* + \theta_1 m\sqrt{2\Delta_0}$$

(5.19b)

$$\nu_2 = m\lambda^* + 1 - (m+1)(\bar{\lambda} + M\sqrt{\Delta_0})$$

(5.19c)

$$\nu_3 = \frac{128L\theta_1\lambda^* m^2}{\theta_2^2\bar{\lambda}\Delta_0\sqrt{\Delta_0}} + 1 - \big(\bar{\lambda} + M\sqrt{\Delta_0}\big)$$

(5.20a) $\theta_1 = \lambda^* + \sigma + M\sqrt{\dfrac{\Delta_0}{2}} + 2\sqrt{L}$

(5.20b) $\theta_2 = \lambda^* - \bar{\lambda} - 2\sqrt{\Delta_0}\big(M + 2\sqrt{L}\big)$,

then we have $x_t^T x_0 \geq 0$ *and* $\Delta_t \leq \Delta_0$ *for all* $0 \leq t \leq m$.

Proof. We prove by induction. Suppose that we have $\Delta_s \leq \Delta_0$ for $s \leq t < m$. Since $\Delta_0 \leq 1 - 1/\sqrt{2}$, this implies that $y_t^T x^* \geq 1/\sqrt{2}$ and $y_0^T x^* \geq 1/\sqrt{2}$. Therefore, we have

$$\begin{aligned}
y_t^T y_0 &= \left[(y_t^T x^*)x^* + y_t - (y_t^T x^*)x^*\right]^T \left[(y_0^T x^*)x^* + y_0 - (y_0^T x^*)x^*\right] \\
&= (y_t^T x^*)(y_0^T x^*) + (y_t - (y_t^T x^*)x^*)^T (y_0 - (y_0^T x^*)x^*) \\
&\geq (y_t^T x^*)(y_0^T x^*) - \|y_t - (y_t^T x^*)x^*\|\|y_0 - (y_0^T x^*)x^*\| \\
&\geq (y_t^T x^*)(y_0^T x^*) - \sqrt{1 - (y_t^T x^*)^2}\sqrt{1 - (y_0^T x^*)^2} \\
&\geq 0,
\end{aligned}$$

which leads to

$$\|x_t - (x_t^T y_0)y_0\|^2 = \|x_t\|^2(1 - (y_k^T y_0)^2) \leq 2\|x_t\|^2(1 - y_t^T y_0) = \|x_t\|^2\|y_t - y_0\|^2.$$

By the triangular inequality, $(a+b)^2 \le 2(a^2+b^2)$ and $\Delta_t \le \Delta_0$, we have

$$\|y_t - y_0\|^2 \le 2(\|y_t - x^*\|^2 + \|y_0 - x^*\|^2) \le 4\|y_0 - x^*\|^2.$$

From $y_0^T x^* \ge 0$, we further obtain

$$\|x_t - (x_t^T y_0) y_0\|^2 \le 4\|x_t\|^2 \|y_0 - x^*\|^2 = 8\|x_t\|^2(1 - y_0^T x^*) \tag{5.21}$$

$$\le 8\|x_t\|^2(1 - (y_0^T x^*)^2) = 8\|x_t\|^2 \sum_{k=2}^{d} (y_0^T v_k)^2. \tag{5.22}$$

Using Lemma 5.3.1, the definitions of M and L, (5.21) and that $\Delta_t \le \Delta_0$, we have

$$\begin{aligned} x_{t+1}^T v_1 &\ge (1 - \eta + \eta\lambda^*)\, x_t^T v_1 - \frac{1}{2}\eta M \|x_t\| \|y_t - x^*\|^2 - \eta\sqrt{L}\|x_t - (x_t^T y_0) y_0\| \\ &\ge (1 - \eta + \eta\lambda^*)\, x_t^T v_1 - \eta M (1 - y_t^T x^*)\|x_t\| - \eta\sqrt{8L(1 - y_0^T x^*)}\|x_t\| \qquad (5.23) \\ &\ge \left[1 - \eta + \eta\left(\lambda^* - \frac{M\Delta_0}{1-\Delta_0} - \frac{\sqrt{8L\Delta_0}}{1-\Delta_0}\right)\right] y_0^T x^* \|x_t\|. \end{aligned}$$

By (5.15), (5.16) and that $L \le L_0$, we have

$$\lambda^* - \frac{M\Delta_0}{1-\Delta_0} - \frac{\sqrt{8L\Delta_0}}{1-\Delta_0} \ge \lambda^* - \left(M + 4\sqrt{L}\right)\sqrt{\Delta_0} = \lambda^* - \frac{(\lambda^* - \bar{\lambda})\left(M + 4\sqrt{L}\right)}{2M + 4\sqrt{L_0}} \ge 0.$$

This leads to $x_{t+1}^T v_1 \ge 0$.

Now, we prove that $\Delta_{t+1} \le \Delta_0$. Since

$$\sum_{k=2}^{d} (1 - \eta + \eta\lambda_k)^2 (x_t^T v_k)^2 \le (1 - \eta + \eta\bar{\lambda})^2 \sum_{k=2}^{d} (x_t^T v_k)^2 \tag{5.24}$$

$$\sum_{k=1}^{d} \left(1 - \eta + \eta(\lambda_k + (\lambda^* - \lambda_1)\mathbb{1}_{k=1})\right)^2 (x_t^T v_k)^2 \le (1 - \eta + \eta\lambda^*)^2 \|x_t\|^2 \tag{5.25}$$

(5.26)
$$\sum_{k=2}^{d}\left[(y_t-x^*)^T F_k(\hat{y}_t^1,\cdots,\hat{y}_t^d)(y_t-x^*)\right]^2 \le \sum_{k=1}^{d}\left[(y_t-x^*)^T F_k(\hat{y}_t^1,\cdots,\hat{y}_t^d)(y_t-x^*)\right]^2 \le M^2\|y_t-x^*\|^4$$

(5.27)
$$\sum_{k=2}^{d}\left[v_k^T G_{S_t}(\bar{y}_t^1,\cdots,\bar{y}_t^d)\big(x_t-(x_t^T y_0)y_0\big)\right]^2 \le \sum_{k=1}^{d}\left[v_k^T G_{S_t}(\bar{y}_t^1,\cdots,\bar{y}_t^d)\big(x_t-(x_t^T y_0)y_0\big)\right]^2 \le L\|x_t-(x_t^T y_0)y_0\|^2$$

where (5.27) follows from $\|\sum_{k=1}^{d} v_k v_k^T\| = 1$, using Lemma 5.3.1 and the Cauchy-Schwarz inequality, we have

(5.28)
$$\sum_{k=2}^{d}(x_{t+1}^T v_k)^2 \le \Big[(1-\eta+\eta\bar{\lambda})\sqrt{\sum_{k=2}^{d}(x_t^T v_k)^2}+\frac{1}{2}\eta M\|x_t\|\|y_t-x^*\|^2+\eta\sqrt{L}\|x_t-(x_t^T y_0)y_0\|\Big]^2$$

(5.29)
$$\|x_{t+1}\|^2 \le \Big[1-\eta+\eta\lambda^*+\frac{1}{2}\eta M\|y_0-x^*\|^2+\eta\sqrt{L}\|y_t-(y_t^T y_0)y_0\|\Big]^2\|x_t\|^2.$$

First, we consider the case (5.17). Since $\Delta_t \le \Delta_0 \le 1$, we have $0 \le y_t^T x^* \le 1$ and $\sum_{k=2}^{d}(y_t^T v_k)^2 = 1-(y_t^T x^*)^2 \le 1-(y_0^T x^*)^2 = \sum_{k=2}^{d}(y_0^T v_k)^2$, resulting in

(5.30)
$$\|y_t-x^*\|^2 = 2\sqrt{1-y_t^T x^*}\sqrt{1-(y_t^T x^*)^2} \le 2\sqrt{\Delta_t}\sqrt{\sum_{k=2}^{d}(y_t^T v_k)^2} \le 2\sqrt{\Delta_0}\sqrt{\sum_{k=2}^{d}(y_0^T v_k)^2}.$$

Plugging (5.22) and (5.30) into (5.28), we have

$$\sum_{k=2}^{d}(x_{t+1}^T v_k)^2 \leq \left[1-\eta+\eta\left(\bar{\lambda}+M\sqrt{\Delta_0}+2\sqrt{2L}\right)\right]^2 \|x_t\|^2 \sum_{k=2}^{d}(y_0^T v_k)^2. \tag{5.31}$$

Combining (5.23) and (5.31), we have

(5.32)
$$\frac{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2}{(x_{t+1}^T v_1)^2} \leq \left[\frac{1-\eta+\eta\left(\bar{\lambda}+M\sqrt{\Delta_0}+2\sqrt{2L}\right)}{1-\eta+\eta\left(\lambda^*-M\Delta_0/(1-\Delta_0)-2\sqrt{2L\Delta_0}/(1-\Delta_0)\right)}\right]^2 \frac{\sum_{k=2}^{d}(y_0^T v_k)^2}{(y_0^T v_1)^2}.$$

Using (5.15) and (5.17), we have

$$\lambda^* - \frac{M\Delta_0}{1-\Delta_0} - \frac{2\sqrt{2L\Delta_0}}{1-\Delta_0} - \left(\bar{\lambda}+M\sqrt{\Delta_0}+2\sqrt{2L}\right) \geq (\lambda^*-\bar{\lambda}) - 2M\sqrt{\Delta_0} - 4\sqrt{2L} \geq 0.$$

Therefore, from (5.32), we finally have

$$\frac{1-(y_{t+1}^T x^*)^2}{(y_{t+1}^T x^*)^2} = \frac{\sum_{k=2}^{d}(y_{t+1}^T v_k)^2}{(y_{t+1}^T v_1)^2} = \frac{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2}{(x_{t+1}^T v_1)^2} \leq \frac{\sum_{k=2}^{d}(y_0^T v_k)^2}{(y_0^T v_1)^2} = \frac{1-(y_0^T x^*)^2}{(y_0^T x^*)^2}.$$

Since $(1-x^2)/x^2$ is decreasing for $x \geq 0$, this leads to $\Delta_{t+1} = 1 - y_{t+1}^T x^* \leq 1 - y_0^T x^* = \Delta_0$.

Next, we derive $\Delta_{t+1} \leq \Delta_0$ from (5.18). From (5.21) and (5.29), we have

$$\|x_{t+1}\|^2 \leq \left[1-\eta+\eta\left(\lambda^* + \frac{1}{2}M\|y_0-x^*\|^2 + 2\sqrt{L}\|y_0-x^*\|\right)\right]^2 \|x_t\|^2.$$

Using induction, this leads to

$$\|x_{t+1}\|^2 \leq \left[1-\eta+\eta\left(\lambda^* + \frac{1}{2}M\|y_0-x^*\|^2 + 2\sqrt{L}\|y_0-x^*\|\right)\right]^{2(t+1)} \|x_0\|^2. \tag{5.33}$$

On the other hand, from (5.13), (5.21) and the definition of L, we have

$$\begin{aligned} x_{t+1}^T y_0 &= (1-\eta)x_t^T y_0 + \frac{\eta \nabla f(x_t)^T y_0}{\|x_t\|^{p-2}} + \eta y_0^T G_{S_t}(\bar{y}_t^1, \cdots, \bar{y}_t^d)(x_t - (x_t^T y_0)y_0) \\ &\geq (1-\eta)x_t^T y_0 + \frac{\eta \nabla f(x_t)^T y_0}{\|x_t\|^{p-2}} - 2\eta\sqrt{L}\|y_0 - x^*\|\|x_t\|. \end{aligned}$$

Replacing v_k with y_0 in (5.9) and using $\nabla f(x^*) = \lambda^* x^*$ and the definition of M, we have

$$\begin{aligned} \frac{\nabla f(x_t)^T y_0}{\|x_t\|^{p-1}} &= \nabla f(x^*)^T y_0 + (y_t - x^*)^T \nabla^2 f(x^*) y_0 + \frac{1}{2}(y_t - x^*)^T \sum_{i=1}^d y_{0i} H_i(\hat{y}_t^i)(y_t - x^*) \\ &= \lambda^* y_t^T y_0 + (y_t - x^*)^T \left(\nabla^2 f(x^*) - \lambda^* I\right) y_0 - \frac{1}{2} M \|y_t - x^*\|^2 \\ &\geq \lambda^* y_t^T y_0 - (\lambda^* + \sigma)\|y_t - x^*\| - \frac{1}{2} M \|y_t - x^*\|^2. \end{aligned}$$

This results in

$$\begin{aligned} x_{t+1}^T y_0 &\geq (1 - \eta + \eta\lambda^*) x_t^T y_0 - \eta\big(\lambda^* + \sigma + \frac{1}{2} M \|y_0 - x^*\| + 2\sqrt{L}\big)\|y_0 - x^*\|\|x_t\| \\ &= (1 - \eta + \eta\lambda^*) x_t^T y_0 - \eta\theta_1 \sqrt{2\Delta_0}\|x_t\|. \end{aligned}$$

Using (5.33), we obtain

$$x_{t+1}^T y_0 \geq (1 - \eta + \eta\lambda^*) x_t^T y_0 - \eta\theta_1 \sqrt{2\Delta_0}\big[1 - \eta + \eta\lambda^* + \eta\theta_1\sqrt{2\Delta_0}\big]^t \|x_0\|.$$

By mathematical recursion, we further have

$$x_{t+1}^T y_0 \geq \left(2(1 - \eta + \eta\lambda^*)^{t+1} - \left[1 - \eta + \eta\lambda^* + \eta\theta_1\sqrt{2\Delta_0}\right]^{t+1}\right)\|x_0\|. \tag{5.34}$$

Since $\|x_t - (x_t^T y_0)y_0\|^2 = \|x_t\|^2 - (x_t^T y_0)^2$, using (5.33) and (5.34), we have

$$\|x_t - (x_t^T y_0)y_0\|^2 \leq 4(1-\eta+\eta\lambda^*)^{2t}\Big[\Big(1+\frac{\eta\theta_1\sqrt{2\Delta_0}}{1-\eta+\eta\lambda^*}\Big)^t - 1\Big]\|x_0\|^2.$$

By (5.18), (5.19a) and (5.14), $\eta(1-\lambda^*+\theta_1 m\sqrt{2\Delta_0}) \leq 1$ or $\eta\theta_1 m\sqrt{2\Delta_0}/(1-\eta+\eta\lambda^*) \leq 1$. Since $\eta\theta_1 t\sqrt{2\Delta_0}/(1-\eta+\eta\lambda^*) \leq \eta\theta_1 m\sqrt{2\Delta_0}/(1-\eta+\eta\lambda^*) \leq 1$ and $(1+x)^t \leq \exp(xt) \leq 2xt+1$ for $xt \leq 1$, we obtain

$$\|x_t - (x_t^T y_0)y_0\|^2 \leq 8\eta\theta_1(1-\eta+\eta\lambda^*)^{2t-1}\sqrt{2\Delta_0}t\|x_0\|^2. \tag{5.35}$$

Plugging (5.30) and (5.35) into the square root of (5.28), we have

(5.36)

$$\begin{aligned}
\sqrt{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2} &\leq \big[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\big]\sqrt{\sum_{k=2}^{d}(x_t^T v_k)^2} \\
&\quad + \eta\sqrt{\frac{8\eta L\theta_1\sqrt{2\Delta_0}}{1-\eta+\eta\lambda^*}}(1-\eta+\eta\lambda^*)^t t\|x_0\| \\
&\leq \big[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\big]^{t+1}\sqrt{\sum_{k=2}^{d}(x_0^T v_k)^2} \\
&\quad + \eta\sqrt{\frac{8\eta L\theta_1\sqrt{2\Delta_0}}{1-\eta+\eta\lambda^*}}\sum_{i=1}^{t} i\,(1-\eta+\eta\lambda^*)^i\big[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\big]^{t-i}\|x_0\|.
\end{aligned}$$

For a positive integer t and a non-negative real number $r \geq 0$ such that $t/t \leq 1$, we have

$$(1+r)^t - 1 = r\left((1+r)^{t-1} + (1+r)^{t-2} + \cdots + 1\right) \geq rt$$

$$(1+r)^t - 1 \leq \exp(rt) - 1 \leq 2rt,$$

which leads to

$$\begin{aligned}\sum_{i=1}^{t}(1+r)^i i &= \frac{1+r}{r^2}\left(t(1+r)^{t+1}-(t+1)(1+r)^t+1\right)\\ &\le \frac{1+r}{r^2}\left(t(1+r)^{t+1}-t(1+r)^t-rt\right)\\ &= \frac{(1+r)t}{r}\left((1+r)^t-1\right)\\ &\le 2(1+r)t^2.\end{aligned} \tag{5.37}$$

By (5.18), (5.19b) and (5.14), we have $\eta\left(m(\lambda^*-\bar{\lambda}-M\sqrt{\Delta_0})+1-\bar{\lambda}-M\sqrt{\Delta_0}\right)\le 1$, which implies

$$\frac{1-\eta+\eta\lambda^*}{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})}-1\le\frac{1}{m}.$$

Also, by (5.16), we have $\lambda^*-\bar{\lambda}-M\sqrt{\Delta_0}$, leading to

$$\frac{1-\eta+\eta\lambda^*}{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})}-1=\frac{\eta(\lambda^*-\bar{\lambda}-M\sqrt{\Delta_0})}{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})}\ge 0.$$

Therefore, using (5.37), we have

$$\begin{aligned}&\sum_{i=1}^{t} i\,(1-\eta+\eta\lambda^*)^i\left[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\right]^{t-i}\\ &=\left[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\right]^t\sum_{i=1}^{t} i\,\left[\frac{1-\eta+\eta\lambda^*}{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})}\right]^i\\ &\le 2(1-\eta+\eta\lambda^*)t^2\left[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\right]^{t-1}.\end{aligned} \tag{5.38}$$

Plugging (5.38) into (5.36), we obtain

(5.39)
$$\sqrt{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2} \le \left[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\right]^{t+1}\sqrt{\sum_{k=2}^{d}(x_0^T v_k)^2} + 2\eta\sqrt{8(1-\eta+\eta\lambda^*)\eta L\theta_1\sqrt{2\Delta_0}}\,t^2\left[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\right]^{t-1}\|x_0\|.$$

On the other hand, from (5.23), we have

$$x_{t+1}^T v_1 \ge \left[1-\eta+\eta\left(\lambda^*-\frac{M\Delta_0}{1-\Delta_0}-\frac{2\sqrt{2L\Delta_0}}{1-\Delta_0}\right)\right]^{t+1} x_0^T v_1. \tag{5.40}$$

Combining (5.39) and (5.40), we have

(5.41)
$$\frac{\sqrt{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2}}{x_{t+1}^T v_1} \le \left[\frac{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})}{1-\eta+\eta\left[\lambda^*-(M\Delta_0+2\sqrt{2L\Delta_0})/(1-\Delta_0)\right]}\right]^{t+1}\frac{\sqrt{\sum_{k=2}^{d}(x_0^T v_k)^2}}{x_0^T v_1} + \frac{2\eta t^2\sqrt{8(1-\eta+\eta\lambda^*)\eta L\theta_1\sqrt{2\Delta_0}}\left[1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})\right]^{t-1}}{\left(1-\eta+\eta\left[\lambda^*-(M\Delta_0+2\sqrt{2L\Delta_0})/(1-\Delta_0)\right]\right)^{t+1} y_0^T v_1}.$$

Since $0<\eta\le 1$, we have

$$\frac{\bar{\lambda}}{\lambda^*} \le \frac{1-\eta+\eta\bar{\lambda}}{1-\eta+\eta\lambda^*} \le \frac{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})}{1-\eta+\eta\lambda^*}. \tag{5.42}$$

Let

$$\gamma = \frac{\lambda^*-\bar{\lambda}-M\sqrt{\Delta_0}-(M\Delta_0+2\sqrt{2L\Delta_0})/(1-\Delta_0)}{1-\eta+\eta\left[\lambda^*-(M\Delta_0+2\sqrt{2L\Delta_0})/(1-\Delta_0)\right]}. \tag{5.43}$$

By (5.15) and $\theta_2 \geq 0$ due to (5.16), we have

(5.44)
$$\frac{1}{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})} = \frac{\gamma}{1-\eta\gamma}\left[\frac{1}{\lambda^*-\bar{\lambda}-M\sqrt{\Delta_0}-(M\Delta_0+2\sqrt{2L\Delta_0})/(1-\Delta_0)}\right] \leq \frac{\gamma}{\theta_2(1-\eta\gamma)}.$$

Using (5.42), (5.44) and that $y_0^T v_1 \geq 1/\sqrt{2}$, we have

$$\frac{2\eta t^2\sqrt{8(1-\eta+\eta\lambda^*)\eta L\theta_1\sqrt{2\Delta_0}}}{y_0^T v_1(1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0}))^2} \leq 8\sqrt{2}\sqrt{\frac{\lambda^*}{\bar{\lambda}}}\sqrt{\frac{\eta L\theta_1\sqrt{\Delta_0}}{1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0})}}\frac{\eta\gamma t^2}{\theta_2(1-\eta\gamma)}.$$

By (5.18), (5.19c) and (5.14), we have

$$\eta\left(\frac{128L\theta_1\lambda^* m^2}{\theta_2^2\bar{\lambda}\Delta_0\sqrt{\Delta_0}} + 1 - \left(\bar{\lambda}+M\sqrt{\Delta_0}\right)\right) \leq 1$$

or

$$\frac{\eta L\theta_1\sqrt{\Delta_0}}{1-\eta+\eta\left(\bar{\lambda}+M\sqrt{\Delta_0}\right)} \leq \frac{\theta_2^2\bar{\lambda}\Delta_0^2}{128\lambda^* m^2},$$

which results in

$$\frac{2\eta t^2\sqrt{8(1-\eta+\eta\lambda^*)\eta L\theta_1\sqrt{2\Delta_0}}}{y_0^T v_1(1-\eta+\eta(\bar{\lambda}+M\sqrt{\Delta_0}))^2} \leq \frac{\eta\gamma t^2\Delta_0}{(1-\eta\gamma)m} \leq \frac{\eta\gamma t^2}{(1-\eta\gamma)m}\frac{\sum_{k=2}^d (x_0^T v_k)^2}{(x_0^T v_1)^2}. \tag{5.45}$$

The last inequality follows from

$$\Delta_0 = 1 - y_0^T x^* \leq 1 - (y_0^T x^*)^2 \leq \frac{\sum_{k=2}^d (y_0^T v_k)^2}{(y_0^T v_1)^2} = \frac{\sum_{k=2}^d (x_0^T v_k)^2}{(x_0^T v_1)^2}.$$

Plugging (5.43) and (5.45) into (5.41), we have

$$\frac{\sqrt{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2}}{x_{t+1}^T v_1} \leq (1-\eta\gamma)^{t+1}\left[1+\frac{\eta\gamma t^2}{(1-\eta\gamma)m}\right]\frac{\sqrt{\sum_{k=2}^{d}(x_0^T v_k)^2}}{x_0^T v_1}.$$

Using $1+nx \leq (1+x)^n$ for $x \geq 0$ and the fact that $\gamma \geq 0$ by (5.16), we have

$$\begin{aligned}(1-\eta\gamma)^{t+1}\left[1+\frac{\eta\gamma t^2}{(1-\eta\gamma)m}\right] &= \left[1-\left[\left(1+\frac{\eta\gamma}{1-\eta\gamma}\right)^{t+1}-1-\frac{\eta\gamma t^2}{(1-\eta\gamma)m}\right](1-\eta\gamma)^{t+1}\right] \\ &\leq \left[1-\Big(t+1-\frac{t^2}{m}\Big)\eta\gamma(1-\eta\gamma)^t\right],\end{aligned}$$

which yields

$$\frac{\sqrt{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2}}{x_{t+1}^T v_1} \leq \frac{\sqrt{\sum_{k=2}^{d}(x_0^T v_k)^2}}{x_0^T v_1}$$

due to $t < m$. From that

$$\frac{1-(y_{t+1}^T x^*)^2}{(y_{t+1}^T x^*)^2} = \frac{\sum_{k=2}^{d}(x_{t+1}^T v_k)^2}{(x_{t+1}^T v_1)^2} \leq \frac{\sum_{k=2}^{d}(x_0^T v_k)^2}{(x_0^T v_1)^2} = \frac{1-(y_0^T x^*)^2}{(y_0^T x^*)^2}$$

and $(1-x^2)/x^2$ is decreasing for $x \geq 0$, we finally have $\Delta_{t+1} = 1-y_{t+1}^T x^* \leq 1-y_0^T x^* = \Delta_0$.

□

Note that $\lambda^* - \bar{\lambda}_2$ is an eigen-gap at the solution and L and Δ_0 are decreasing functions of the batch size $|S|$ and the dot product $y_0^T x^*$. Given that Δ_0 is moderately small, we can satisfy conditions (5.17) or (5.18) by increasing the batch size $|S|$ or decreasing the step size η, respectively. Conditioning on x_t, the next lemma derives expectation bounds for several quantities involving $(x_{t+1}^T v_k)^2$ and norms.

Lemma 5.3.3. *For any positive integer m, if η, $|S|$ and x_0 satisfy* (5.16), (5.17) *(or* (5.18)*) and*

$$\eta \leq \max(1, 1 - \lambda^* + \sqrt{2}M\Delta_0)^{-1}, \tag{5.46}$$

then for any $0 \leq t \leq m$, we have

$$\begin{aligned}
E[\|x_{t+1}\|^2|x_t] &\leq \big[\big(\alpha(\eta) + \eta M\Delta_t\big)^2 + \eta^2 K\big]\|x_t\|^2, \\
E\big[\sum_{k=2}^{d}(x_{t+1}^T v_k)^2|x_t\big] &\leq \big(\beta(\eta) + \eta M\sqrt{\Delta_t}\big)^2 \sum_{k=2}^{d}(x_t^T v_k)^2 + 8\eta^2 K\|x_t\|^2 \sum_{k=2}^{d}(y_0^T v_k)^2, \\
E[(x_{t+1}^T v_1)^2|x_t] &\geq \left[\alpha(\eta) - \frac{\eta M\Delta_t}{1-\Delta_t}\right]^2 (x_t^T v_1)^2.
\end{aligned}$$

Proof. By Lemma 5.3.1, we have

$$\begin{aligned}
x_{t+1}^T v_k = &\left(1 - \eta + \eta(\lambda_k + (\lambda^* - \lambda_1)\mathbb{1}_{k=1})\right) x_t^T v_k \\
&+ \frac{1}{2}\eta\|x_t\|(y_t - x^*)^T F_k(\hat{y}_t^1, \cdots, \hat{y}_t^d)(y_t - x^*) \\
&+ \eta\left(G_{S_t}(\bar{y}_t^1, \cdots, \bar{y}_t^d)\big(x_t - (x_t^T y_0)y_0\big)\right)^T v_k.
\end{aligned}$$

Since S_t is sampled uniformly at random, $E[f_{S_t}(y)] = f(y)$ for all $y \in \mathbb{R}^d$, which leads to

$$\begin{aligned}
E[(x_{t+1}^T v_1)^2 \,|\, x_t] = &\big[(1 - \eta + \eta\lambda^*)x_t^T v_1 + \frac{1}{2}\eta\|x_t\|(y_t - x^*)^T F_1(\hat{y}_t^1, \cdots, \hat{y}_t^d)(y_t - x^*)\big]^2 \\
&+ \eta^2\big(x_t - (x_t^T y_0)y_0\big)^T E[G_{S_t}(\bar{y}_t^1, \cdots, \bar{y}_t^d)^T v_1 v_1^T G_{S_t}(\bar{y}_t^1, \cdots, \bar{y}_t^d)]\big(x_t - (x_t^T y_0)y_0\big).
\end{aligned} \tag{5.47}$$

In the same way, for $2 \le k \le d$, we have

(5.48)
$$E[(x_{t+1}^T v_k)^2 \,|\, x_t] = \big[(1-\eta+\eta\lambda_k)x_t^T v_k + \frac{1}{2}\eta\|x_t\|(y_t-x^*)^T F_k(\hat{y}_t^1,\cdots,\hat{y}_t^d)(y_t-x^*)\big]^2 + \eta^2\big(x_t-(x_t^T y_0)y_0\big)^T E[G_{S_t}(\bar{y}_t^1,\cdots,\bar{y}_t^d)^T v_k v_k^T G_{S_t}(\bar{y}_t^1,\cdots,\bar{y}_t^d)]\big(x_t-(x_t^T y_0)y_0\big).$$

Using the definition of M and that $\|\sum_{k=1}^d v_k v_k^T\| = 1$, we have

(5.49)
$$\eta^2\big(x_t-(x_t^T y_0)y_0\big)^T \sum_{k=1}^d E[\|G_{S_t}(\bar{y}_t^1,\cdots,\bar{y}_t^d)^T v_k\|^2]\big(x_t-(x_t^T y_0)y_0\big) \le \eta^2 K\|x_t-(x_t^T y_0)y_0\|^2.$$

Using (5.47), (5.48), (5.25), (5.26), (5.49) and the Cauchy-Schwarz inequality, we have

$$\begin{aligned} E[\|x_{t+1}\|^2|x_t] &\le (1-\eta+\eta\lambda^*)^2\|x_t\|^2 + \frac{1}{2}\eta M(1-\eta+\eta\lambda^*)\|x_t\|\|y_k-x^*\|^2 \\ &\quad + \frac{1}{4}\eta^2 M^2\|x_t\|^2\|y_k-x^*\|^4 + \eta^2 K\|x_t-(x_t^T y_0)y_0\|^2. \end{aligned} \tag{5.50}$$

Using $\|x_t-(x_t^T y_0)y_0\|^2 \le \|x_t\|^2$ in (5.50), we obtain

$$\begin{aligned} E[\|x_{t+1}\|^2|x_t] &\le \Big[\big(1-\eta+\eta\lambda^*+\frac{1}{2}\eta M\|y_t-x^*\|^2\big)^2 + \eta^2 K\Big]\|x_t\|^2 \\ &= \Big[\big(1-\eta+\eta\lambda^*+\eta M(1-y_t^T x^*)\big)^2 + \eta^2 K\Big]\|x_t\|^2, \end{aligned} \tag{5.51}$$

which establishes the first statement.

In the same way, using (5.48), (5.24), (5.26), (5.49) and the Cauchy-Schwarz inequality, we have

$$\begin{aligned} E\Big[\sum_{k=2}^d (x_{t+1}^T v_k)^2|x_t\Big] &\le \Big[(1-\eta+\eta\bar{\lambda})\sqrt{\sum_{k=2}^d (x_t^T v_k)^2} + \frac{1}{2}\eta M\|x_t\|\|y_t-x^*\|^2\Big]^2 \\ &\quad + \eta^2 K\|x_t-(x_t^T y_0)y_0\|^2. \end{aligned} \tag{5.52}$$

By Lemma 5.3.2, we have $\Delta_t \le \Delta_0 \le 1 - 1/\sqrt{2}$ and thus $y_t^T x^* \ge 1/\sqrt{2}$ and $y_0^T x^* \ge 1/\sqrt{2}$. Since $y_t^T x^* \ge 0$, using (5.30), we have

$$\frac{1}{2}\eta M \|x_t\| \|y_t - x^*\|^2 \le \eta M \sqrt{\Delta_t} \sqrt{\sum_{k=2}^{d} (x_t^T v_k)^2}.$$

As a result of (5.22) which we can use since $\Delta_t \le \Delta_0$, we obtain

(5.53)
$$E\Big[\sum_{k=2}^{d} (x_{t+1}^T v_k)^2 | x_t\Big] \le \Big[1 - \eta + \eta\bar{\lambda} + \eta M\sqrt{\Delta_t}\Big]^2 \sum_{k=2}^{d} (x_t^T v_k)^2 + 8\eta^2 K \|x_t\|^2 \sum_{k=2}^{d} (y_0^T v_k)^2,$$

which shows the second statement in the lemma.

Lastly, from (5.47), we have

$$E[(x_{t+1}^T v_1)^2 | x_t] \ge \left[(1 - \eta + \eta\lambda^*) x_t^T v_1 + \frac{1}{2}\eta \|x_t\| (y_t - x^*)^T F_1(\hat{y}_t^1, \cdots, \hat{y}_t^d)(y_t - x^*)\right]^2$$

By (5.46) and (5.14), we have $\eta(1 - \lambda^* + M\Delta_0\sqrt{2}) \le 1$. Since $1/(1 - \Delta_0) \le \sqrt{2}$ by (5.16), we further have

$$\eta\left(\frac{M\Delta_0}{1 - \Delta_0} + 1 - \lambda^*\right) \le 1.$$

Due to $\Delta_t \le \Delta_0$, this implies that

$$\begin{aligned}
(1 - \eta + \eta\lambda^*) x_t^T v_1 - \frac{1}{2}\eta M \|x_t\| \|y_t - x^*\|^2 &= \Big[(1 - \eta + \eta\lambda^*)(1 - \Delta_t) - \eta M \Delta_t\Big] \|x_t\| \\
&= \Big[1 - \eta\Big(\frac{M\Delta_t}{1 - \Delta_t} + 1 - \lambda^*\Big)\Big](1 - \Delta_t)\|x_t\| \\
&\ge \Big[1 - \eta\Big(\frac{M\Delta_0}{1 - \Delta_0} + 1 - \lambda^*\Big)\Big](1 - \Delta_t)\|x_t\| \\
&\ge 0.
\end{aligned}$$

Since $(a+b)^2 \geq (a-c)^2$ holds if $a \geq c$ and $|b| \leq c$, we finally have

$$\begin{aligned} E[(x_{t+1}^T v_1)^2 | x_t] &\geq \left[(1-\eta+\eta\lambda^*) x_t^T v_1 - \frac{1}{2}\eta M \|x_t\| \|y_t - x^*\|^2 \right]^2 \\ &= \left[\alpha(\eta) - \frac{\eta M \Delta_t}{1-\Delta_t} \right]^2 (x_t^T v_1)^2. \end{aligned}$$

□

Using induction on the single iteration bound in Lemma 5.3.3, we derive an upper bound for $E[\sum_{k=2}^d (x_t^T v_k)^2]$ and a lower bound $E[(x_t^T v_1)^2]$ as functions of $E[\sum_{k=2}^d (x_0^T v_k)^2]$ and $E[(x_0^T v_1)^2]$ for a single outer iteration.

Lemma 5.3.4. *For any positive integer m, if η, $|S|$ and x_0 satisfy* (5.16), (5.17) *(or* (5.18)*),* (5.46) *and*

$$\eta \leq \max\left(1, 1-\lambda^* - M\sqrt{\Delta_0} + \sqrt{Km}\right)^{-1}, \tag{5.54}$$

then we have

$$\begin{aligned} E\Big[\sum_{k=2}^d (x_t^T v_k)^2\Big] &\leq E\Big[\sum_{k=2}^d (x_0^T v_k)^2\Big] \Big[\big(\beta(\eta) + \eta M\sqrt{\Delta_0}\big)^{2t} + 16\eta^2 K t \big(\alpha(\eta) + \eta M\sqrt{\Delta_0}\big)^{2(t-1)}\Big], \\ E[(x_t^T v_1)^2] &\geq \left[\alpha(\eta) - \frac{\eta M \Delta_0}{1-\Delta_0}\right]^{2t} E[(x_0^T v_1)^2]. \end{aligned}$$

Proof. By Lemma 5.3.2, we have $\Delta_t \leq \Delta_0$. Repeatedly applying Lemma 5.3.3, we have

$$\begin{aligned} E[\|x_t\|^2 | x_0] = E[E[\|x_t\|^2 | x_{t-1}] | x_0] &\leq \big[\big(\alpha(\eta) + \eta M \Delta_0\big)^2 + \eta^2 K\big] E[\|x_{t-1}\|^2 | x_0] \\ &\leq \big[\big(\alpha(\eta) + \eta M \Delta_0\big)^2 + \eta^2 K\big]^t \|x_0\|^2. \end{aligned} \tag{5.55}$$

Using (5.55), we have

$$E\Big[\|x_t\|^2\sum_{k=2}^{d}(y_0^Tv_k)^2\Big] = E\Big[E\Big[\|x_t\|^2\sum_{k=2}^{d}(y_0^Tv_k)^2|x_0\Big]\Big] = E\Big[E[\|x_t\|^2|x_0]\sum_{k=2}^{d}(y_0^Tv_k)^2\Big]$$

$$= E\Big[\big[\big(\alpha(\eta)+\eta M\Delta_0\big)^2+\eta^2K\big]^t\|x_0\|^2\sum_{k=2}^{d}(y_0^Tv_k)^2\Big] \tag{5.56}$$

$$= \big[\big(\alpha(\eta)+\eta M\Delta_0\big)^2+\eta^2K\big]^tE\Big[\sum_{k=2}^{d}(x_0^Tv_k)^2\Big].$$

Using Lemma 5.3.3 and that $\Delta_t \le \Delta_0$, we have

$$E\Big[\sum_{k=2}^{d}(x_t^Tv_k)^2\Big] \le \big(\beta(\eta)+\eta M\sqrt{\Delta_0}\big)^2E\Big[\sum_{k=2}^{d}(x_{t-1}^Tv_k)^2\Big] + 8\eta^2KE\Big[\|x_{t-1}\|^2\sum_{k=2}^{d}(y_0^Tv_k)^2\Big]. \tag{5.57}$$

By induction on (5.57) using (5.56), we have

$$\begin{aligned}
E\Big[\sum_{k=2}^{d}(x_t^Tv_k)^2\Big] &\le \big(\beta(\eta)+\eta M\sqrt{\Delta_0}\big)^2E\Big[\sum_{k=2}^{d}(x_{t-1}^Tv_k)^2\Big] \\
&\quad + 8\eta^2K\big[\big(\alpha(\eta)+\eta M\Delta_0\big)^2+\eta^2K\big]^{t-1}E\Big[\sum_{k=2}^{d}(x_0^Tv_k)^2\Big] \\
&\le E\Big[\sum_{k=2}^{d}(x_0^Tv_k)^2\Big]\Big[\big(\beta(\eta)+\eta M\sqrt{\Delta_0}\big)^{2t} \\
&\quad + 8\eta^2K\sum_{s=1}^{t}\big[\alpha(\eta)+\eta M\sqrt{\Delta_0}\big]^{2(t-s)}\big[\big(\alpha(\eta)+\eta M\sqrt{\Delta_0}\big)^2+\eta^2K\big]^{s-1}\Big] \\
&\le E\Big[\sum_{k=2}^{d}(x_0^Tv_k)^2\Big]\Big[\big(\beta(\eta)+\eta M\sqrt{\Delta_0}\big)^{2t} \\
&\quad + 8\big(\alpha(\eta)+\eta M\sqrt{\Delta_0}\big)^{2t}\Big[\Big(1+\frac{\eta^2K}{\big(\alpha(\eta)+\eta M\sqrt{\Delta_0}\big)^2}\Big)^t-1\Big]\Big].
\end{aligned}$$

By (5.54) and (5.14), we have $\eta\big(1-\lambda^*-M\sqrt{\Delta_0}+\sqrt{Km}\big)\leq 1$, which leads to

$$0\leq \frac{\eta^2 Kt}{\big(\alpha(\eta)+\eta M\sqrt{\Delta_0}\big)^2}\leq 1.$$

Using $(1+x)^t-1\leq \exp(xt)-1\leq 2xt$ for $xt\in[0,1]$, we have

$$E\Big[\sum_{k=2}^{d}(x_t^T v_k)^2\Big]\leq E\Big[\sum_{k=2}^{d}(x_0^T v_k)^2\Big]\left[\big(\beta(\eta)+\eta M\sqrt{\Delta_0}\big)^{2t}+16\eta^2 Kt\big(\alpha(\eta)+\eta M\sqrt{\Delta_0}\big)^{2(t-1)}\right].$$

On the other hand, using $\Delta_t\leq\Delta_0$ and Lemma 5.3.3, we have

$$E[(x_t^T v_1)^2]=E[E[(x_t^T v_1)^2|x_{t-1}]]\geq\left[\alpha(\eta)-\frac{\eta M\Delta_0}{1-\Delta_0}\right]^2 E[(x_{t-1}^T v_1)^2]. \tag{5.58}$$

By induction on (5.58) using $\Delta_t\leq\Delta_0$, we finally have

$$E[(x_t^T v_1)^2]\geq\left[\alpha(\eta)-\frac{\eta M\Delta_0}{1-\Delta_0}\right]^{2t} E[(x_0^T v_1)^2].$$

□

The inequalities in Lemma 5.3.4 are important since we can combined them to yield a bound on the optimality gap which is expressed as $E\big[\sum_{k=2}^{d}(x_t^T v_k)^2\big]/E[(x_t^T v_1)^2]$. In the next lemma, we show that under some conditions on $\eta, m, |S|$ and x_0, the optimality gap decreases at least by $1-\rho$ after each outer iteration.

Lemma 5.3.5. *For any positive integer m, if η, $|S|$ and x_0 satisfy* (5.16), (5.17) *(or* (5.18)*) and*

$$\eta\leq \max(1,\nu_4,\nu_5)^{-1} \tag{5.59}$$

where

$$\nu_4 = 1 - \lambda^* - M\sqrt{\Delta_0} + \max\left(\sqrt{Km},\ \frac{64K}{\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0}}\right) \tag{5.60}$$

$$\nu_5 = 1 - \lambda^* + M\sqrt{\Delta_0} + \max\left(2m\big(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0}\big),\ \frac{4mM\sqrt{\Delta_0}}{\log 2}\right), \tag{5.61}$$

then we have

$$\frac{E[\sum_{k=2}^{d}(x_m^T v_k)^2]}{E[(x_m^T v_1)^2]} \le (1-\rho)\cdot\frac{E[\sum_{k=2}^{d}(x_0^T v_k)^2]}{E[(x_0^T v_1)^2]}$$

where

$$0 < \rho = \frac{\eta m\big(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0}\big)}{2\big(1 - \eta + \eta(\lambda^* - M\sqrt{\Delta_0})\big)} < 1. \tag{5.62}$$

Proof. By (5.59) and (5.60), we have (5.54). Also, (5.59), (5.61) and the fact that $\sqrt{2\Delta_0} \le 1$ due to (5.16) imply (5.46). Therefore, by Lemma 5.3.4, we have

$$\delta_m \le \left[\left(\frac{\beta(\eta) + \eta M\sqrt{\Delta_0}}{\alpha(\eta) - \eta M\Delta_0/(1-\Delta_0)}\right)^{2m} + \frac{16\eta^2 Km\big[\alpha(\eta) + \eta M\sqrt{\Delta_0}\big]^{2(m-1)}}{\big[\alpha(\eta) - \eta M\Delta_0/(1-\Delta_0)\big]^{2m}}\right]\delta_0 \tag{5.63}$$

where $\delta_t = E[\sum_{k=2}^{d}(x_t^T v_k)^2]/E[(x_t^T v_1)^2]$. By (5.15) due to (5.16) and the fact that $1 + x \le \exp(x)$ for all $x \in \mathbb{R}$, we have

$$\begin{aligned}
\left(\frac{\beta(\eta) + \eta M\sqrt{\Delta_0}}{\alpha(\eta) - \eta M\Delta_0/(1-\Delta_0)}\right)^{2m} &\le \left(1 - \frac{\eta(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0})}{1 - \eta + \eta(\lambda^* - M\sqrt{\Delta_0})}\right)^{2m} \\
&\le \exp\left(-\frac{2\eta m(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0})}{1 - \eta + \eta(\lambda^* - M\sqrt{\Delta_0})}\right) \\
&\le 1 - \frac{\eta m(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0})}{1 - \eta + \eta(\lambda^* - M\sqrt{\Delta_0})} \\
&= 1 - 2\rho
\end{aligned} \tag{5.64}$$

where the last inequality follows from that $\exp(-x) \leq 1 - x/2$ for $0 \leq x \leq 1$ since (5.59), (5.61) and (5.14) imply $2\eta m(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0})/(1 - \eta + \eta(\lambda^* - M\sqrt{\Delta_0})) \leq 1$.

On the other hand, by (5.15) due to (5.16) and the fact that $(1+x)^n \leq \exp(nx)$, we have

(5.65)
$$\begin{aligned}\frac{16\eta^2 Km\big[\alpha(\eta) + \eta M\sqrt{\Delta_0}\big]^{2(m-1)}}{\big[\alpha(\eta) - \eta M\Delta_0/(1-\Delta_0)\big]^{2m}} &\leq \frac{16\eta^2 Km}{\big(\alpha(\eta) + \eta M\sqrt{\Delta_0}\big)^2}\left(1 + \frac{2\eta M\sqrt{\Delta_0}}{\alpha(\eta) - \eta M\sqrt{\Delta_0}}\right)^{2m} \\ &\leq \frac{16\eta^2 Km}{\big(\alpha(\eta) + \eta M\sqrt{\Delta_0}\big)^2}\exp\left(\frac{4\eta m M\sqrt{\Delta_0}}{\alpha(\eta) - \eta M\sqrt{\Delta_0}}\right).\end{aligned}$$

By (5.59), (5.61) and (5.14), we have $\eta\big(1 - \lambda^* - M\sqrt{\Delta_0} + 64K/(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0})\big) \leq 1$, which leads to

(5.66)
$$\begin{aligned}\frac{\rho}{2} - \frac{16\eta^2 Km}{\big(\alpha(\eta) + \eta M\sqrt{\Delta_0}\big)^2} &\geq \frac{\eta m(\lambda^* - \bar{\lambda} - 2M\sqrt{\Delta_0})}{4\big(1 - \eta + \eta(\lambda^* + M\sqrt{\Delta_0})\big)} - \frac{16\eta^2 Km}{\big(1 - \eta + \eta(\lambda^* + M\sqrt{\Delta_0})\big)^2} \\ &\geq 0.\end{aligned}$$

By (5.59), (5.61) and (5.14), we have $\eta(1 - \lambda^* + M\sqrt{\Delta_0} + 4mM\sqrt{\Delta_0}/\log 2) \leq 1$, resulting in

$$\exp\left(\frac{4\eta m M\sqrt{\Delta_0}}{\alpha(\eta) - \eta M\sqrt{\Delta_0}}\right) \leq 2. \tag{5.67}$$

Using (5.64), (5.65), (5.66) and (5.67) in (5.63), we finally have

$$\frac{E[\sum_{k=2}^d (x_m^T v_k)^2]}{E[(x_m^T v_1)^2]} \leq (1-\rho) \cdot \frac{E[\sum_{k=2}^d (x_0^T v_k)^2]}{E[(x_0^T v_1)^2]}.$$

□

Finally, we analyze the entire algorithm. Let

$$\tilde{\Delta}_0 = 1 - \tilde{x}_0^T x^*, \quad \tilde{\delta}_s = \frac{E[\sum_{k=2}^d (\tilde{x}_s^T v_k)^2]}{E[(\tilde{x}_s^T v_1)^2]}.$$

By repeatedly applying Lemma 5.3.5, the following theorem states that $\tilde{\delta}_s$ decreases at a liner rate under some conditions on $\eta, |S|$ and $\tilde{x}_0$.

Theorem 5.3.6. *For any positive integer m, if η, $|S|$ and $\tilde{x}_0$ satisfy* (5.16), (5.17) *(or* (5.18)*) and* (5.59) *with $\Delta_0 = \tilde{\Delta}_0$, then for any $\epsilon > 0$, after $\tau = \lceil (1/\rho)\log(\tilde{\delta}_0/\epsilon) \rceil$ epochs of S-SCI-PI (Algorithm 5), we have $\tilde{\delta}_\tau \leq \epsilon$.*

Proof. Since η, $|S|$ and $x_0 = \tilde{x}_0$ satisfy (5.16), (5.17) (or (5.18)) and (5.59), by Lemmas 5.3.2 and 5.3.5, we have $\tilde{\Delta}_1 = \Delta_m \leq \Delta_0 = \tilde{\Delta}_0$ and $\tilde{\delta}_1 = \delta_m \leq (1-\rho)\delta_0 = (1-\rho)\tilde{\delta}_0$. By repeatedly applying the same argument, we have $\tilde{\delta}_\tau \leq (1-\rho)^\tau \tilde{\delta}_0$. Since $\tau \geq (1/\rho)\log(\tilde{\delta}_0/\epsilon)$, we finally obtain

$$\tilde{\delta}_\tau \leq (1-\rho)^\tau \tilde{\delta}_0 \leq \exp(-\tau\rho)\tilde{\delta}_0 \leq \epsilon.$$

This completes the proof. □

Theorem 5.3.6 states that for epoch length m, if $\widetilde{x}_0$ is moderately close to x^* and the step size η and the batch size $|S|$ satisfies certain conditions, the optimality gap vanishes at an exponential rate. If there are few irregular f_i and sampling costs are cheap, we can satisfy (5.17) by making L small. In this case, η can take a large value and we are able to obtain rapid convergence. On the other hand, if there are many irregular data samples and sampling costs are not cheap, we may not satisfy (5.17). Nevertheless, we

can always ensure linear convergence of Algorithm 5 by choosing a small enough step size η as in [**72**].

5.4. KL-divergence NMF

Let $V \in \mathbb{R}_+^{N\times M}$ be a given non-negative matrix, which we want to compress into the product of $W \in \mathbb{R}_+^{N\times K}$ and $H \in \mathbb{R}_+^{K\times M}$.

Consider the KL-NMF problem defined in (3.4.4). Let H_j be the j-th column of H. Note that the objective function $D_{KL}(V\|WH)$ is separable in $H_1, \cdots, H_M$ and thus

$$H_j^{\text{new}} = \underset{H_j\geq 0}{\arg\max} \sum_i [V_{ij}\log(WH_j)_i - (WH_j)_i] \tag{5.68}$$

serves as the j-th subproblem. By Lemma 3.4.5, the j-th KL-NMF subproblem (5.68) is equivalent to the following mixture proportion problem

$$X_j^{\text{new}} = \underset{X_j\in\mathcal{S}^d}{\arg\max} \sum_{i=1}^N V_{ij}\log(LX_j)_i \tag{5.69}$$

with $L_{ik} = W_{ik}/(\sum_{i'} W_{i'k})$ and the original solution can be recovered via

$$H_{kj}^{\text{new}} = \frac{\sum_i V_{ij}}{\sum_i W_{ik}} X_{jk}^{\text{new}}, \quad k = 1, \cdots, d. \tag{5.70}$$

This result implies that the KL divergence NMF subproblem for H, namely

$$H^{\text{new}} = \underset{H\geq 0}{\arg\max} [V_{ij}\log(WH)_{ij} - (WH)_{ij}],$$

can be solved by S-SCI-PI after we reformulate problem (5.68) into (5.69) for $j = 1, \cdots, M$.

Using a stochastic sample $S \subset \{1, \cdots, N\}$ of size $|S|$, our S-SCI-PI updates H by

$$
\begin{aligned}
H_{kj}^{\text{new}} &\leftarrow H_{kj} \left[(1-\eta) + \eta \sum_{i \in S} \frac{L_{ik} V_{ij}}{(LH)_{ij}} \right]^2 \quad \forall k, j, \\
H^{\text{new}} &\leftarrow \text{column-rescale}(H^{\text{new}}),
\end{aligned} \tag{5.71}
$$

where column-rescale(X) is rescaling the columns of X to have sum 1. The update for W is similar due to

$$D_{KL}(V \| WH) = D_{KL}(V^T \| H^T W^T).$$

Rather than dealing with M sub-problems (5.68), we tackle a single optimization problem. Let $X = [X_1, \cdots, X_M]$ be the concatenation of the M column vectors $X_1, \cdots, X_M \in \mathbb{R}^K$ defined on the unit simplex. Lemma 3.4.5 states that in the exact alternating minimization algorithm, the update of H amounts to solving

$$\min_X \sum_{i=1}^{NM} \text{vec}(V)_i \log[(I_M \otimes L)\text{vec}(X)]_i \quad \text{subject to} \quad \text{vec}(X_j) \in \mathcal{S}^K, \quad j = 1, \cdots, M \tag{5.72}$$

where $\text{vec}(X) = (X_{11}, \cdots, X_{K1}, \cdots, X_{1M}, \cdots, X_{KM})$ is a vectorization of $X \in \mathbb{R}^{K \times M}$, $\text{vec}(V) \in \mathbb{R}^{KM}$ is defined similarly and $I_M \otimes L = \text{kron}(I_M, L) \in \mathbb{R}^{NM \times KM}$ is the Kronecker product of I_M and L.

This allows us to exploit fast matrix multiplication routines (i.e. efficient matrix computation library such as OpenBLAS or intel MKL) in solving the aggregated problem (5.72) instead of solving the j-th subproblem sequentially for $j = 1, \cdots, M$.

5.4.1. Related Algorithms

Let $Z = WH$ henceforth. We omit the update of W since it can be derived similarly.

Multiplicative Update (MU/EM) [**45**]: MU updates all H_{kj}'s simultaneously by

$$H_{kj}^{\text{new}} = H_{kj} \frac{\sum_i W_{ik} V_{ij}/Z_{ij}}{\sum_i W_{ik}}$$

for all k and j.

Let us emphasize that the MU update is identical to the standard EM algorithm for the estimation of mixture proportions.

Cyclic Coordinate Descent (CCD/SCD) [**28**, **60**]: For all j and k, CCD/CSD runs coordinate-wise updates of H

$$H_{kj}^{\text{new}} = \max\left\{0, H_{kj} - \frac{\sum_i W_{ik}(1 - V_{ij}/Z_{ij})}{\sum_i V_{ij} W_{ik}^2/Z_{ij}^2}\right\}$$

sequentially in a pre-fixed cyclic order.

Projected Gradient Descent (PGD) [**48**]: PGD given element-wise step sizes (denoted by α_{kj}'s) updates all H_{kj}'s simulataneously via

$$H_{kj}^{\text{new}} = \max\left\{0, H_{kj} - \alpha_{kj}\left(\textstyle\sum_i W_{ik}(1 - V_{ij}/Z_{ij})\right)\right\}.$$

Note that Multiplicative Update (MU) is a special case of PGD when $\alpha_{jk} = H_{jk}/(\sum_i W_{ik})$, which does not require projection onto the non-negative orthant. Also, CCD updates H_{jk} one at a time with a coordinate-wise optimal step size $\alpha_{jk} = 1/\sum_i (V_i W_{ik}^2/Z_{ij}^2)$. By contrast, PGD uses a single step size $\alpha_j = \alpha_{j1} = \cdots = \alpha_{jK}$ for each column j for fast line searches.

The proposed S-SCI-PI algorithm (5.71) is most similar to MU since W and H are updated multiplicatively as well. As a special case of S-SCI-PI, the full-batch S-SCI-PI is denoted by F-SCI-PI.

Let us highlight that S-SCI-PI and all the comparison methods belong to the family of alternating minimization algorithms, which update H given W and then update W given H iteratively.

5.4.2. Practical Considerations

Various Sampling Scheme. In this part, we compare several sampling schemes for the update of H. The sampling scheme for the update of W can be similarly discussed, but omitted.

Vector-wise Sampling: We construct $V_S \in \mathbb{R}_+^{|S| \times M}$ and $W_S \in \mathbb{R}_+^{|S| \times D}$ by sampling rows of $V \in \mathbb{R}_+^{N \times M}$ and $W \in \mathbb{R}_+^{N \times D}$ uniformly at random, respectively. The stochastic gradient reads

$$\nabla_S^{\text{row}} f(H) = \frac{n}{|S|} W_S^T [V_S \oslash (W_S H)].$$

For a dense data matrix V, we prefer to use this vector-wise (or row-wise) sampling scheme for the update of H, since it allows us to exploit fast matrix multiplication libraries.

Element-wise Sampling: We vectorize the problem by introducing the element-wise iterator $i = (i_1, i_2) \in [N] \times [M] = [NM]$. This yields

$$f(H) = \sum_{i \in \mathcal{I}} V_{i_1, i_2} \log \sum_{k=1}^{D} W_{i_1, k} H_{k, i_2}$$

where $\mathcal{I}$ is the subset of $[NM]$ such that $V_{i_1,i_2} = 0$ if and only if $i = (i_1, i_2) \in \mathcal{I}$. In other words, $\mathcal{I}$ is the index set of the nonzero elements in V.

We construct S by sampling $|S|$ elements of $\mathcal{I}$ uniformly at random, and consider the stochastic gradient as

$$\nabla_S^{\text{elem}} f(H) = \frac{|\mathcal{I}|}{|S|} \sum_{i \in S} \sum_k \frac{W_{i_1,k} V_{i_1,i_2}}{\sum_{k'=1}^D W_{i_1,k'} H_{k',i_2}} E_{k,i_2}$$

where E_{k,i_2} is the standard basis matrix having 1 at (k, i_2)-th entry and 0 otherwise.

For a sparse data matrix V, we prefer this element-wise sampling scheme for H over the row-wise sampling scheme, since each column has a different sparsity pattern.

Numerical Issues. Since the KL-NMF objective function (3.51) and its gradient are unstable when entries of V and WH are close to 0. As reported in [**38**] and based on the experiments in Section 5.5, MU and the full-batch version of F-SCI-PI are numerically stable. On the other hand, S-SCI-PI has certain numerical issues since randomness of the stochastic gradient may lead entries of W and H arbitrary close to 0. Thus we have added safeguard to avoid this by rejecting the stochastic gradient when it produces a numeric error (i.e. when any of the elements of stochastic gradient is negative). In such a case we proceed to the next iteration.

5.5. Numerical Experiments

We test the proposed algorithm S-SCI-PI on synthetic and real-world data sets. All experiments are implemented on a standard laptop (2.6 GHz Intel Core i7 processor and 16GB of RAM) using the C++ programming language.

We use 4 real data sets publicly available online (See Table 3.1) and three synthetic data sets generated from Poisson distributions. We preprocess the real data sets by removing few rows and columns having sums less than 20 for NIPS and KOS data sets. For synthetic data, $V \in \mathbb{R}^{N\times M}$ generated from i.i.d. Poisson random variables, i.e. $V_{ij} \sim \text{Poisson}(-\log(1-\rho))$. Here ρ denotes sparsity or proportion of nonzero entries of V. This corresponds to the null signal case since in this case KL-NMF is the maximum likelihood estimation problem when $WH = 0$.

Table 5.1. Summary of synthetic datasets for KL-NMF

Name	# of samples	# of features	# of nonzeros	Sparsity
Pois1	1,000	1,000	900,000	0.90
Pois2	3,000	3,000	900,000	0.10
Pois3	9,000	9,000	900,000	0.01

We set $K = 20$ features. All the reported values are averaged over 10 independent replicates started at different initial points, each of which is obtained by running 5 MU/EM steps on a Uniform(0,1) random matrix. In certain runs due to numerical errors the outcomes were peculiar and thus they were disregarded (but each observation has 5 normal runs). The benchmark algorithms are MU/EM, CCD/SCD, PGD, and F-SCI-PI.

For S-SCI-PI, we perform grid search on the parameters by selecting the best parameters among different batch proportions $|S|/n \in \{0.0001, 0.001, 0.01, 0.1\}$, epoch lengths $m \in \{10, 100, 1000\}$ and step sizes $\eta \in \{0.01, 0.1, 1\}$.

5.5.1. KL-NMF Subproblem

First, we compare the performance of S-SCI-PI against the four benchmark algorithms (F-SCI-PI, MU/EM, CCD/SCD, PGD) on the KL-NMF subproblem (5.72). We run the

algorithms until they attain an optimal solution W^* and compare relative objective values over time.

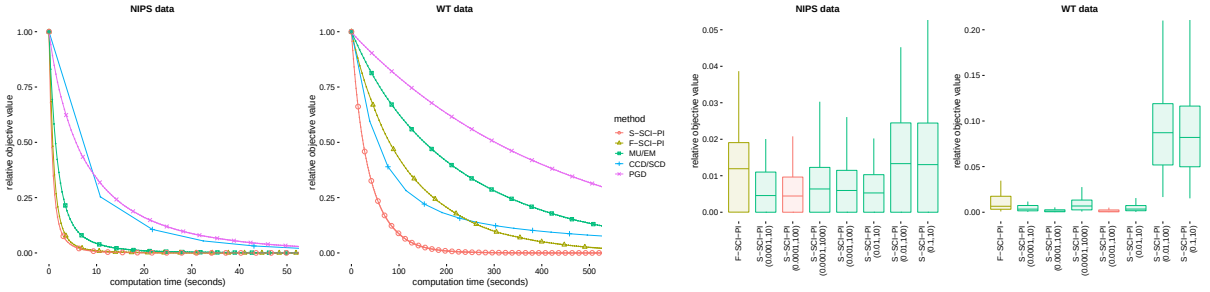

Figure 5.1. *(Left 2 figures)* Convergence plots for the KL-NMF subproblem. *(Right 2 figures)* Boxplots showing the relative errors after 30 seconds from 10 independent replicates. The red colored boxes indicate the selected batch sizes and epoch lengths, respectively.

The left two figures in Figure 5.1 display the results for the two larger size real world data sets (NIPS, WT). It shows that S-SCI-PI is an overall winner solving the KL divergence subproblems and hence an efficient method for exact alternating minimization. However, it does not outperform F-SCI-PI significantly on the sparse NIPS data set. As reported in [**28**], CCD/SCD is faster than MU/EM for the dense WT data set. However, our result on NIPS shows that CCD/SCD is very slow mainly due to the expensive coordinate updates.

Next, we compare convergence of S-SCI-PI with different batch sizes, epoch lengths and step sizes. We select two data sets (NIPS and WT) and report the relative objective values of S-SCI-PI with few selected parameters choices in the right figure in Figure 5.1. They show that a careful choice of the batch size $|S|$ and epoch length m yields non-negligible improvements on the convergence of S-SCI-PI. Again, we confirm that the

stochastic approach (S-SCI-PI) has a remarkable improvement over the full gradient approach (F-SCI-PI) for the dense WT data set.

5.5.2. KL-NMF Problem

To solve the entire KL-NMF problem (3.51), we update H via a single iteration of each algorithm (a single epoch for S-SCI-PI) and then update W in a similar way. We compare S-SCI-PI with F-SCI-PI and MU/EM. We leave out CCD/SCD and PGD since they are much slower.

For dense data sets (WT, MITF), we apply vector-wise sampling only on the columns (of dimension 19,200 and 2,429, respectively) since the other dimension is small (287 and 361, respectively). For sparse data sets (NIPS, KOS), the element-wise sampling scheme is applied to both dimensions, which turns out to be more effective.

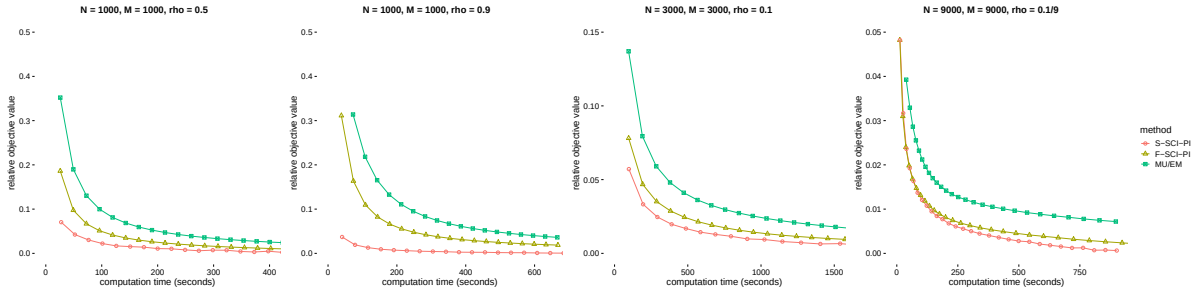

Figure 5.2. Convergence plots (relative error vs. computation time) of one-step alternating minimization on synthetic data sets.

Figure 5.2 displays the results for the four synthetic data sets. By comparing the left two figures in Figure 5.2, and the right figures, we conclude that S-SCI-PI performs much better than F-SCI-PI for dense matrices, but is the winner also for sparse matrices.

Figure 5.3 displays the relative errors with respect to the computation time for the 4 real data sets. Overall, S-SCI-PI with the chosen batch and epoch size improves the

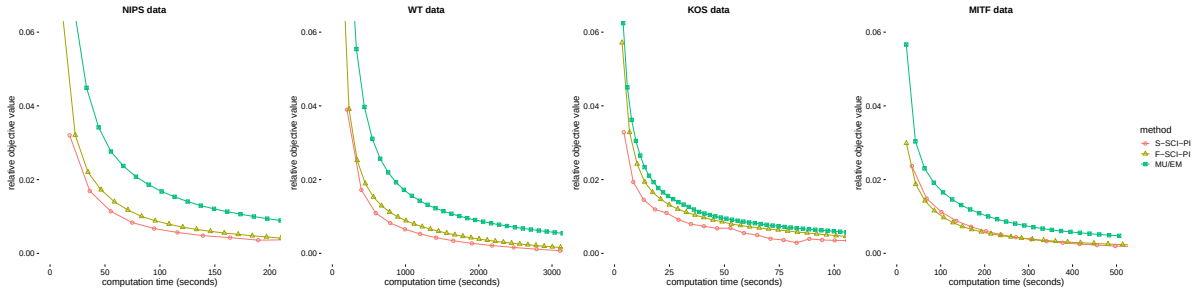

Figure 5.3. Convergence plots (relative error vs. computation time) of one-step alternating minimization on real data sets.

convergence over F-SCI-PI. However, S-SCI-PI does not outperform F-SCI-PI for the MITF data set, which has a relatively small number of columns (2,429). Also, both S-SCI-PI and F-SCI-PI exhibit much faster convergence than MU/EM. This clearly attests that S-SCI-PI is an excellent practical option for the KL-NMF problem.

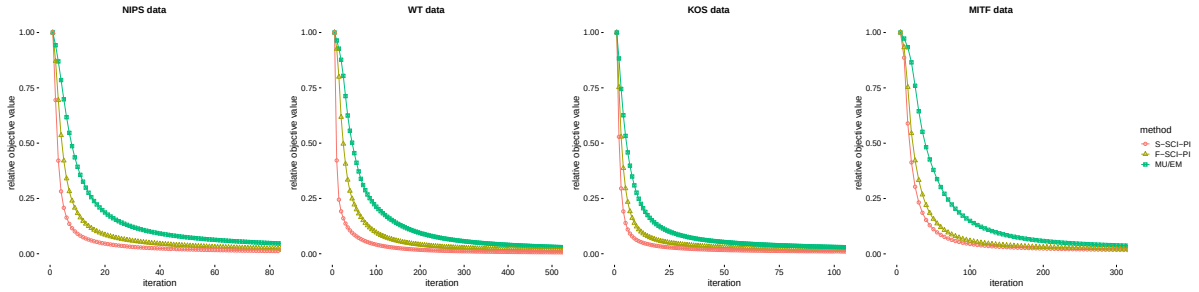

Figure 5.4. Convergence plots (relative error vs. iteration) of one-step alternating minimization on real data sets.

Figure 5.4 displays the relative errors with respect to the outer loop iterations. S-SCI-PI takes longer steps per outer loop iteration than F-SCI-PI and MU/EM, at the expense of larger computational complexity. The batch size and the epoch length balance the trade-off between them. We also notice that F-SCI-PI and MU/EM have almost the same computation time but F-SCI-PI takes longer step than MU/EM.

5.6. Final Remarks

In this work, we introduce a stochastic variance-reduced algorithm (S-SCI-PI) to solve finite-sum scale invariant problems and provide its convergence analysis. Our analysis shows that under some conditions on initial iterate, epoch length, batch size, and an additional condition the algorithm attains linear convergence in expectation. This algorithm is applied to solve the KL-NMF problem. The experimental results demonstrate its superior performance over state-of-the-art methods.

CHAPTER 6

Conclusion

This thesis introduces scale invariant problems and studies deterministic and stochastic solution methods. Starting with the L1-norm kernel PCA problem, we develop a novel dual reformulation of scale invariant problems and derive an iterative algorithm based on geometrical understandings of the dual formulation. Our algorithm, SCI-PI, not only has a general form of power iteration but also extends the attractive linear convergence property of power iteration. The second half of this thesis studies scale invariant problems with finite-sum objective functions. In order to exploit the finite-sum structure, we develop stochastic generalizations of power iteration and SCI-PI to solve PCA and finite-sum scale invariant problems. Built upon the recent stochastic variance-reduced gradient technique, our stochastic algorithms attain linear convergence in expectation. Our numerical experiments on various unsupervised machine learning models reveal that the proposed deterministic and stochastic algorithms are computationally competitive to state-of-the-art algorithms as well as often yield better solutions.

References

[1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[2] Zeyuan Allen-Zhu and Yuanzhi Li. LazySVD: Even faster SVD decomposition yet without agonizing pain. In *Advances in Neural Information Processing Systems*, pages 974–982, 2016.

[3] Zeyuan Allen-Zhu and Yuanzhi Li. First efficient convergence for streaming k-PCA: a global, gap-free, and near-optimal rate. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 487–492, 2017.

[4] Raman Arora, Andrew Cotter, Karen Livescu, and Nathan Srebro. Stochastic optimization for PCA and PLS. In *Annual Allerton Conference on Communication, Control, and Computing*, pages 861–868, 2012.

[5] Maria-Florina Balcan, Simon Shaolei Du, Yining Wang, and Adams Wei Yu. An improved gap-dependency analysis of the noisy power method. In *Conference on Learning Theory*, pages 284–309, 2016.

[6] Akshay Balsubramani, Sanjoy Dasgupta, and Yoav Freund. The fast convergence of incremental PCA. In *Advances in Neural Information Processing Systems*, pages 3174–3182, 2013.

[7] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *International Conference on Music Information Retrieval*, 2011.

[8] Jock A Blackard and Denis J Dean. Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables. *Computers and Electronics in Agriculture*, 24(3):131–151, 1999.

[9] Christos Boutsidis, Dan Garber, Zohar Karnin, and Edo Liberty. Online principal components analysis. In *Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 887–901, 2015.

[10] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[11] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. LOF: identifying density-based local outliers. In *ACM SIGMOD International Conference on Management of Data*, pages 93–104, 2000.

[12] J Paul Brooks, José H Dulá, and Edward L Boone. A pure L1-norm principal component analysis. *Computational Statistics and Data Analysis*, 61:83–98, 2013.

[13] Sébastien Bubeck. Convex optimization: algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.

[14] Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

[15] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):11, 2011.

[16] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717, 2009.

[17] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011.

[18] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.

[19] Chris Ding, Ding Zhou, Xiaofeng He, and Hongyuan Zha. R_1-PCA: rotational invariant L_1-norm principal component analysis for robust subspace factorization. In *International Conference on Machine Learning*, pages 281–288, 2006.

[20] Murat A Erdogdu, Asuman Ozdaglar, Pablo A Parrilo, and Nuri Denizcan Vanli. Convergence rate of block-coordinate maximization Burer-Monteiro method for solving large SDPs. *arXiv preprint arXiv:1807.04428*, 2018.

[21] Cédric Févotte and Jérôme Idier. Algorithms for nonnegative matrix factorization with the β-divergence. *Neural Computation*, 23(9):2421–2456, 2011.

[22] Dan Garber and Elad Hazan. Fast and simple PCA via convex optimization. *arXiv preprint arXiv:1509.05647*, 2015.

[23] Dan Garber, Elad Hazan, Chi Jin, Sham M Kakade, Cameron Musco, Praneeth Netrapalli, and Aaron Sidford. Faster eigenvector computation via shift-and-invert preconditioning. In *International Conference on Machine Learning*, pages 2626–2634, 2016.

[24] Dan Garber, Elad Hazan, and Tengyu Ma. Online learning of eigenvectors. In *International Conference on Machine Learning*, pages 560–568, 2015.

[25] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[26] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[27] Moritz Hardt and Eric Price. The noisy power method: A meta algorithm with applications. In *Advances in Neural Information Processing Systems*, pages 2861–2869, 2014.

[28] Cho-Jui Hsieh and Inderjit S Dhillon. Fast coordinate descent methods with variable selection for non-negative matrix factorization. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1064–1072, 2011.

[29] Aapo Hyvarinen. Fast ICA for noisy data using Gaussian moments. In *IEEE International Symposium on Circuits and Systems*, volume 5, pages 57–61, 1999.

[30] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*. John Wiley & Sons, 2004.

[31] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural Networks*, 13(4-5):411–430, 2000.

[32] Prateek Jain, Chi Jin, Sham M Kakade, Praneeth Netrapalli, and Aaron Sidford. Streaming PCA: matching matrix Bernstein and near-optimal finite sample guarantees for Oja's algorithm. In *Conference on Learning Theory*, pages 1147–1164, 2016.

[33] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323, 2013.

[34] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[35] Michel Journée, Yurii Nesterov, Peter Richtárik, and Rodolphe Sepulchre. Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research*, 11(Feb):517–553, 2010.

[36] George N Karystinos and Athanasios P Liavas. Efficient computation of the binary vector that maximizes a rank-deficient quadratic form. *IEEE Transactions on Information Theory*, 56(7):3581–3593, 2010.

[37] Qifa Ke and Takeo Kanade. Robust L_1 norm factorization in the presence of outliers and missing data by alternative convex programming. In *Conference on Computer Vision and Pattern Recognition*, pages 739–746, 2005.

[38] Cheolmin Kim, Youngseok Kim, and Diego Klabjan. Scale invariant power iteration. *arXiv preprint arXiv:1905.09882*, 2019.

[39] Cheolmin Kim and Diego Klabjan. A simple and fast algorithm for L1-norm kernel PCA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[40] Cheolmin Kim and Diego Klabjan. Stochastic variance-reduced algorithms for PCA with arbitrary mini-batch sizes. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[41] Youngseok Kim, Peter Carbonetto, Matthew Stephens, and Mihai Anitescu. A fast algorithm for maximum likelihood estimation of mixture proportions using sequential quadratic programming. *arXiv preprint arXiv:1806.01412*, 2018.

[42] Nojun Kwak. Principal component analysis based on L_1-norm maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(9):1672–1680, 2008.

[43] Nojun Kwak. Nonlinear projection trick in kernel methods: an alternative to the kernel trick. *IEEE Transactions on Neural Networks and Learning Systems*, 24(12):2113–2119, 2013.

[44] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[45] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, pages 556–562, 2001.

[46] Qi Lei, Kai Zhong, and Inderjit S Dhillon. Coordinate-wise power method. In *Advances in Neural Information Processing Systems*, pages 2064–2072, 2016.

[47] Chris Junchi Li, Mengdi Wang, Han Liu, and Tong Zhang. Near-optimal stochastic approximation for online principal component estimation. *Mathematical Programming*, 167(1):75–97, 2018.

[48] Chih-Jen Lin. Projected gradient methods for non-negative matrix factorization. *Neural Computation*, 19(10):2756–2779, 2007.

[49] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *IEEE International Conference on Data Mining*, pages 413–422, 2008.

[50] Guangcan Liu and Ping Li. Recovery of coherent data via low-rank dictionary pursuit. In *Advances in Neural Information Processing Systems*, pages 1206–1214, 2014.

[51] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2013.

[52] Guangcan Liu, Qingshan Liu, and Ping Li. Blessing of dimensionality: recovering mixture data via dictionary pursuit. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(1):47–60, 2017.

[53] Guangcan Liu, Huan Xu, Jinhui Tang, Qingshan Liu, and Shuicheng Yan. A deterministic analysis for LRR. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):417–430, 2016.

[54] Huikang Liu, Man-Chung Yue, and Anthony Man-Cho So. On the estimation performance and convergence rate of the generalized power method for phase synchronization. *SIAM Journal on Optimization*, 27(4):2426–2446, 2017.

[55] Ronny Luss and Marc Teboulle. Conditional gradient algorithms for rank-one matrix approximations with a sparsity constraint. *SIAM Review*, 55(1):65–98, 2013.

[56] Panos P Markopoulos, George N Karystinos, and Dimitris A Pados. Optimal algorithms for L_1-subspace signal processing. *IEEE Transactions on Signal Processing*, 62(19):5046–5058, 2014.

[57] John C Mason and David C Handscomb. *Chebyshev polynomials*. Chapman and Hall/CRC, 2002.

[58] Michael McCoy and Joel A Tropp. Two proposals for robust PCA using semidefinite programming. *Electronic Journal of Statistics*, 5:1123–1160, 2011.

[59] Ioannis Mitliagkas, Constantine Caramanis, and Prateek Jain. Memory limited, streaming PCA. In *Advances in Neural Information Processing Systems*, pages 2886–2894, 2013.

[60] Laura Muzzarelli, Susanne Weis, Simon B Eickhoff, and Kaustubh R Patil. Rank selection in non-negative matrix factorization: systematic comparison and a new MAD metric. In *International Joint Conference on Neural Networks*, pages 1–8, 2019.

[61] Yu Nesterov. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization Methods and Software*, 9(1-3):141–160, 1998.

[62] Feiping Nie, Heng Huang, Chris Ding, Dijun Luo, and Hua Wang. Robust principal component analysis with non-greedy L1-norm maximization. In *International Joint Conference on Artificial Intelligence*, volume 22, pages 1433–1438, 2011.

[63] Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273, 1982.

[64] Young Woong Park. *Optimization for regression, PCA, and SVM: optimality and scalability*. PhD thesis, Northwestern University, 2015.

[65] Young Woong Park and Diego Klabjan. Iteratively reweighted least squares algorithms for L1-norm principal component analysis. In *IEEE International Conference on Data Mining*, pages 430–438, 2016.

[66] Young Woong Park and Diego Klabjan. Three iteratively reweighted least squares algorithms for L_1-norm principal component analysis. *Knowledge and Information Systems*, 54(3):541–565, 2018.

[67] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[68] Danil Prokhorov. IJCNN 2001 neural network competition. In *International Joint Conference on Neural Networks*, volume 1, page 97, 2001.

[69] Shebuti Rayana. ODDS library, 2016.

[70] Prasanna K Sahoo and Palaniappan Kannappan. *Introduction to functional equations*. Chapman and Hall/CRC, 2011.

[71] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International Conference on Artificial Neural Networks*, pages 583–588, 1997.

[72] Ohad Shamir. A stochastic PCA and SVD Algorithm with an exponential convergence rate. In *International Conference on Machine Learning*, pages 144–152, 2015.

[73] Ohad Shamir. Fast stochastic algorithms for SVD and PCA: convergence properties and convexity. In *International Conference on Machine Learning*, pages 248–256, 2016.

[74] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. In *IEEE International Conference on Data Mining*, pages 172–179, 2003.

[75] Jialei Wang, Weiran Wang, Dan Garber, and Nathan Srebro. Efficient coordinate-wise leading eigenvector computation. In *Algorithmic Learning Theory*, pages 806–820, 2018.

[76] Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: a comprehensive review. *IEEE Transactions on Knowledge and Data Engineering*, 25(6):1336–1353, 2013.

[77] Yingchao Xiao, Huangang Wang, Wenli Xu, and Junwu Zhou. L1 norm based KPCA for novelty detection. *Pattern Recognition*, 46(1):389–396, 2013.

[78] Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust PCA via outlier pursuit. In *Advances in Neural Information Processing Systems*, pages 2496–2504, 2010.

[79] Peng Xu, Bryan He, Christopher De Sa, Ioannis Mitliagkas, and Chris Re. Accelerated stochastic power iteration. In *International Conference on Artificial Intelligence and Statistics*, pages 58–67, 2018.

[80] Zhiqiang Xu. Gradient descent meets shift-and-invert preconditioning for eigenvector computation. In *Advances in Neural Information Processing Systems*, pages 2830–2839, 2018.

APPENDIX A

Additional Lemmas

A.1. Chapter 3

On several occasions, we use if $x \in \partial B_d$, $y \in \partial B_d$, then

$$\|x - y\|^2 = \|x\|^2 + \|y\|^2 - 2x^T y = 2(1 - x^T y).$$

Note that if $x^T y \geq 0$, then

$$\sqrt{1 - (x^T y)^2} = \sqrt{(1 - x^T y)(1 + x^T y)} \geq \sqrt{1 - x^T y} = \frac{\|x - y\|}{\sqrt{2}}.$$

By Cauchy-Schwarz, we also have

$$\sqrt{1 - (x^T y)^2} = \sqrt{(1 - x^T y)(1 + x^T y)} \leq \sqrt{2}\sqrt{1 - x^T y} = \|x - y\|.$$

A.1.1. For the Proofs of Theorem 3.3.2 and Theorem 3.4.2

Lemma A.1.1. *Let $\{v_1, \ldots, v_d\}$ be an orthogonal basis in $\mathbb{R}^d$ with $x^* = v_1$ and $\{x_k\}_{k=0,1,\cdots}$ be the sequence of iterates generated by SCI-PI. If for every $x \in \partial\mathcal{B}_d$ we have*

$$\nabla f(x)^T v_1 = \lambda^* + \alpha(x), \quad \sum_{i=2}^{d} (\nabla f(x)^T v_i)^2 \leq \left(\bar{\lambda}_2 \|x - x^*\| + \beta(x)\right)^2 \tag{A.1}$$

where

$$\alpha(x) = o(\sqrt{\|x - x^*\|}), \quad \beta(x) = o(\|x - x^*\|),$$

then there exists some $\delta > 0$ such that under the initial condition $1 - x_0^T x^* < \delta$, we have

$$1 - (x_k^T x^*)^2 \leq \prod_{t=0}^{k-1} \left(\frac{\bar{\lambda}_2}{\lambda^*} + \gamma_t \right)^2 \left(1 - (x_0^T x^*)^2 \right), \ \frac{\bar{\lambda}_2}{\lambda^*} + \gamma_t < 1, \text{ and } \lim_{k \to \infty} \gamma_k = 0.$$

Proof. By (A.1) for every $x \in \partial \mathcal{B}_d$, we have

$$\frac{\sum_{i=2}^d (\nabla f(x)^T v_i)^2}{(\nabla f(x)^T v_1)^2} \leq \left(\frac{\bar{\lambda}_2 \|x - x^*\| + \beta(x)}{\lambda^* + \alpha(x)} \right)^2.$$

Let

$$\frac{\bar{\lambda}_2 \|x - x^*\| + \beta(x)}{\lambda^* + \alpha(x)} = \frac{\bar{\lambda}_2}{\lambda^*} \|x - x^*\| + \theta(x).$$

Then, we have $\theta(x) = o(\|x - x^*\|)$ and

$$\frac{\sum_{i=2}^d (\nabla f(x)^T v_i)^2}{(\nabla f(x)^T v_1)^2} \leq \left(\frac{\bar{\lambda}_2}{\lambda^*} + \frac{\theta(x)}{\|x - x^*\|} \right)^2 \|x - x^*\|^2. \tag{A.2}$$

Letting

$$\epsilon(x) = \frac{\theta(x)}{\|x - x^*\|}, \tag{A.3}$$

we can further represent (A.2) as

$$\begin{aligned} \frac{\sum_{i=2}^d (\nabla f(x)^T v_i)^2}{(\nabla f(x)^T v_1)^2} &\leq \left(\frac{\bar{\lambda}_2}{\lambda^*} + \epsilon(x) \right)^2 \left(1 + \frac{1 - x^T x^*}{1 + x^T x^*} \right) \left(1 - (x^T x^*)^2 \right) \\ &= \left(\frac{\bar{\lambda}_2}{\lambda^*} + \gamma(x) \right)^2 \left(1 - (x^T x^*)^2 \right) \end{aligned} \tag{A.4}$$

where

$$\gamma(x) = \frac{\bar{\lambda}_2}{\lambda^*}\left(\frac{1 - x^T x^*}{1 + x^T x^* + \sqrt{2(1 + x^T x^*)}}\right) + \epsilon(x)\sqrt{1 + \frac{1 - x^T x^*}{1 + x^T x^*}}. \tag{A.5}$$

From (A.1), there exists some $\delta_1 > 0$ such that if $1 - x^T x^* < \delta_1$, then

$$\nabla f(x)^T v_1 > 0. \tag{A.6}$$

Also, by (A.3), for any $\bar{\gamma} > 0$ satisfying

$$\frac{\bar{\lambda}_2}{\lambda^*} + \bar{\gamma} < 1, \tag{A.7}$$

there exists some constant $\delta_2 > 0$ such that if $1 - x^T x^* < \delta_2$, then

$$|\epsilon(x)| \leq \frac{\bar{\gamma}}{4}. \tag{A.8}$$

Let $\delta = \min\{\delta_1, \delta_2, \frac{\lambda^*}{\lambda_2}\bar{\gamma}, 1\}$. Before proving the main statement, we first prove the following two statements:

1. If $1 - x_k^T x^* < \delta$, then we have

$$x_{k+1}^T x^* > 0,\ 1 - (x_{k+1}^T x^*)^2 \leq \left(\frac{\bar{\lambda}_2}{\lambda^*} + \gamma_k\right)^2 \left(1 - (x_k^T x^*)^2\right),\ \text{and } \gamma_k \leq \bar{\gamma}. \tag{A.9}$$

Since $\delta < 1$, we have $x_k^T x^* > 0$. Also, from $1 - x_k^T x^* < \delta_1$ and $x^* = v_1$, using the update rule of SCI-PI and (A.6), we obtain $x_{k+1}^T x^* = \dfrac{\nabla f(x_k)^T v_1}{\|\nabla f(x_k)\|} > 0$. On other the hand, since $|x_{k+1}^T v_1| \leq \|x_{k+1}\|\|v_1\| = 1$, we have

$$1 - (x_{k+1}^T x^*)^2 \leq \frac{1 - (x_{k+1}^T v_1)^2}{(x_{k+1}^T v_1)^2}.$$

Also, from the fact that $\{v_1, \ldots, v_d\}$ forms an orthogonal basis in $\mathbb{R}^d$, we have $\nabla f(x_k) = \sum_{i=1}^d (\nabla f(x_k)^T v_i) v_i$ and $\|\nabla f(x_k)\|^2 = \sum_{i=1}^d (\nabla f(x_k)^T v_i)^2$. Using the update rule of SCI-PI, we have

$$\frac{1 - (x_{k+1}^T v_1)^2}{(x_{k+1}^T v_1)^2} = \frac{\|\nabla f(x_k)\|^2 - (\nabla f(x_k)^T v_1)^2}{(\nabla f(x_k)^T v_1)^2} = \frac{\sum_{i=2}^d (\nabla f(x_k)^T v_i)^2}{(\nabla f(x_k)^T v_1)^2},$$

resulting in

$$1 - (x_{k+1}^T x^*)^2 \leq \frac{\sum_{i=2}^d (\nabla f(x_k)^T v_i)^2}{(\nabla f(x_k)^T v_1)^2}.$$

Let $\gamma_k = \gamma(x_k)$ and $\epsilon_k = \epsilon(x_k)$. Since $x_k^T x^* > 0$ and $1 - x_k^T x^* < \min\{\delta_2, \frac{\lambda^*}{\bar{\lambda}_2}\bar{\gamma}\}$, from (A.5), we have

$$\gamma_k = \frac{\bar{\lambda}_2}{\lambda^*}\left(\frac{1 - x_k^T x^*}{1 + x_k^T x^* + \sqrt{2(1 + x_k^T x^*)}}\right) + \epsilon_k \sqrt{1 + \frac{1 - x_k^T x^*}{1 + x_k^T x^*}} \leq \frac{\bar{\gamma}}{2} + \frac{\bar{\gamma}}{2} = \bar{\gamma},$$

2. Using mathematical induction, we show that if

$$1 - x_0^T x^* < \delta, \tag{A.10}$$

then, for all $k \geq 0$. we have

$$1 - x_k^T x^* < \delta. \tag{A.11}$$

By (A.10), we have $1 - x_0^T x^* < \delta$, which shows the base case. Next, suppose that $1 - x_k^T x^* < \delta$ holds. Then, we have (A.9). Also, from $\delta < 1$, we have $x_k^T x^* > 0$. Since

$$x_{k+1}^T x^* > 0,\ x_k^T x^* > 0,\ 1 - (x_{k+1}^T x^*)^2 \leq \left(\frac{\bar{\lambda}_2}{\lambda^*} + \bar{\gamma}\right)^2 \left(1 - (x_k^T x^*)^2\right) < 1 - (x_k^T x^*)^2$$

we have

$$1 - x_{k+1}^T x^* < 1 - x_k^T x^* < \delta,$$

which completes the induction proof.

Now, we prove the main statement. Since (A.11) holds for all $k \geq 0$, we can repeatedly apply (A.9) to obtain

$$1 - (x_k^T x^*)^2 \leq \prod_{t=0}^{k-1} \left(\frac{\bar{\lambda}_2}{\lambda^*} + \gamma_t \right)^2 \left(1 - (x_0^T x^*)^2\right), \text{ and } \quad \frac{\bar{\lambda}_2}{\lambda^*} + \gamma_k \leq \frac{\bar{\lambda}_2}{\lambda^*} + \bar{\gamma} \leq 1.$$

Since

$$1 - (x_k^T x^*)^2 < \left(\frac{\bar{\lambda}_2}{\lambda^*} + \bar{\gamma} \right)^{2k} \left(1 - (x_0^T x^*)^2\right), \tag{A.12}$$

we have $(x_k^T x^*)^2 \to 1$. Moreover, from that $x_k^T x^* > 0$ for all $k \geq 0$ by (A.11), we have $x_k \to x^*$, and thus $\lim_{k\to\infty} \gamma_k = 0$ by (A.5). With (A.12), this gives the desired result. □

Lemma A.1.2. *Let $\{v_1, \ldots, v_d\}$ be an orthogonal basis in $\mathbb{R}^d$. If $x^* = v_1$ and a sequence of iterates $\{x_k\}_{k=0,1,\cdots}$ generated by SCI-PI satisfies*

$$\nabla f(x_k)^T v_1 \geq A - B(1 - x_k^T x^*) - C\sqrt{1 - x_k^T x^*} \tag{A.13}$$

and

$$\sum_{i=2}^{d} (\nabla f(x_k)^T v_i)^2 \leq \left(D\sqrt{1 - (x_k^T x^*)^2} + E\sqrt{2(1 - x_k^T x^*)} + \frac{F}{2}\|x_k - x^*\|^2 \right)^2 \tag{A.14}$$

where $A > 0$ and B, C, D, E, F are non-negative real numbers such that

$$B + C > 0, \quad \frac{D+E}{A} < 1.$$

Then, under the initial condition that $1 - x_0^T x^* < \delta$ where

$$\delta = \min\left\{ \left(\frac{A}{B+C}\right)^2, \left(\frac{A-D-E}{B+C+E+F}\right)^2, 1 \right\}, \tag{A.15}$$

we have

$$1 - (x_k^T x^*)^2 \leq \prod_{t=0}^{k-1} \left(\frac{D+E}{A} + \gamma_t\right)^2 \left(1 - (x_0^T x^*)^2\right), \frac{D+E}{A} + \gamma_t < 1, \text{ and } \lim_{k\to\infty} \gamma_k = 0.$$

Proof. Before proving the main result, we first show the following two statements:

(1) If $1 - x_k^T x^* < \delta$, then we have

$$x_{k+1}^T x^* > 0, 1 - (x_{k+1}^T x^*)^2 < \left(\frac{D+E}{A} + \gamma_k\right)^2 \left(1 - (x_k^T x^*)^2\right), \frac{D+E}{A} + \gamma_k < 1 \tag{A.16}$$

for all $k \geq 0$ where

$$\gamma_k = \frac{\left(A(E+F) + (B+C)(D+E)\right)\sqrt{1 - x_k^T x^*}}{A\left(A - (B+C)\sqrt{1 - x_k^T x^*}\right)}. \tag{A.17}$$

Since $0 < x_k^T x^* \leq 1$, we have $\sqrt{1 - x_k^T x^*} \geq 1 - x_k^T x^*$. Using $x^* = v_1$, the update rule of SCI-PI, (A.13), and the fact that $\delta \leq (A/(B+C))^2$, we have

$$x_{k+1}^T x^* = \frac{\nabla f(x_k)^T v_1}{\|\nabla f(x_k)\|} \geq \frac{A - B(1 - x_k^T x^*) - C\sqrt{1 - x_k^T x^*}}{\|\nabla f(x_k)\|} > 0 \tag{A.18}$$

since

$$\frac{A - B(1 - x_k^T x^*) - C\sqrt{1 - x_k^T x^*}}{\|\nabla f(x_k)\|} \geq \frac{A - (B + C)\sqrt{1 - x_k^T x^*}}{\|\nabla f(x_k)\|} > 0.$$

Using the same arguments in Lemma A.1.1, we have

$$1 - (x_{k+1}^T x^*)^2 \leq \frac{\sum_{i=2}^d (\nabla f(x_k)^T v_i)^2}{(\nabla f(x_k)^T v_1)^2}. \tag{A.19}$$

By (A.18), we have

$$A - B(1 - x_k^T x^*) - C\sqrt{1 - x_k^T x^*} > 0.$$

Therefore, by plugging (A.13) and (A.14) into (A.19) and using that $x_k^T x^* > 0$, we have

$$\begin{aligned}
1 - (x_{k+1}^T x^*)^2 &\leq \left(\frac{D\sqrt{1 - (x_k^T x^*)^2} + E\sqrt{2(1 - x_k^T x^*)} + \frac{F}{2}\|x_k - x^*\|^2}{A - B(1 - x_k^T x^*) - C\sqrt{1 - x_k^T x^*}} \right)^2 \\
&= \left(\frac{D + E\sqrt{1 + \frac{1 - x_k^T x^*}{1 + x_k^T x^*}} + F\sqrt{\frac{1 - x_k^T x^*}{1 + x_k^T x^*}}}{A - B(1 - x_k^T x^*) - C\sqrt{1 - x_k^T x^*}} \right)^2 \left(1 - (x_k^T x^*)^2\right) \\
&\leq \left(\frac{D + E\left(1 + \sqrt{1 - x_k^T x^*}\right) + F\sqrt{1 - x_k^T x^*}}{A - (B + C)\sqrt{1 - x_k^T x^*}} \right)^2 \left(1 - (x_k^T x^*)^2\right) \\
&= \left(\frac{D + E}{A} + \gamma_k \right)^2 \left(1 - (x_k^T x^*)^2\right) \qquad \text{(A.20)}
\end{aligned}$$

where we use the fact that $\sqrt{1 + x} \leq 1 + \sqrt{x}$ for $x \geq 0$ to derive the second inequality. Lastly, from

$$\sqrt{1 - x_k^T x^*} < \sqrt{\delta} \leq \frac{A - D - E}{B + C + E + F},$$

we have

$$\gamma_k < 1 - \frac{D+E}{A}.$$

(2) Using mathematical induction, we show that if

$$1 - x_0^T x^* < \delta, \tag{A.21}$$

then, for all $k \geq 0$, we have

$$1 - x_k^T x^* < \delta. \tag{A.22}$$

By (A.21), we have $1 - x_0^T x^* < \delta$, which proves the base case. Next, suppose that we have $1 - x_k^T x^* < \delta$. Then, we have (A.16). Also, from $\delta < 1$, we have $x_k^T x^* > 0$. Since

$$x_{k+1}^T x^* > 0,\ x_k^T x^* > 0,\ 1 - (x_{k+1}^T x^*)^2 < 1 - (x_k^T x^*)^2,$$

we have

$$1 - x_{k+1}^T x^* < 1 - x_k^T x^* < \delta.$$

This completes the induction proof.

Now, we prove the main statement. Since (A.22) holds for all $k \geq 0$, by repeatedly applying (A.16), we obtain

$$1 - (x_k^T x^*)^2 \leq \prod_{t=0}^{k-1} \left(\frac{D+E}{A} + \gamma_t \right)^2 \left(1 - (x_0^T x^*)^2\right), \text{ and } \frac{D+E}{A} + \gamma_k < 1. \tag{A.23}$$

Since $(D+E)/A+\gamma_k<1$ for all $k\geq 0$, $1-(x_k^Tx^*)^2$ is monotone decreasing, and so is $1-x_k^Tx^*$ by non-negativity. Moreover, from that γ_k is a monotone increasing function of $1-x_k^Tx^*$, we have $\gamma_{k+1}\leq\gamma_k$ for all $k\geq 0$, resulting in

$$\prod_{t=0}^{k-1}\left(\frac{D+E}{A}+\gamma_t\right)^2\leq\left(\frac{D+E}{A}+\gamma_0\right)^{2k}.$$

Since $(D+E)/A+\gamma_0<1$ by (A.16), we have $(x_k^Tx^*)^2\to 1$. Due to $x_k^Tx^*>0$ for all $k\geq 0$, this implies $x_k\to x^*$, and thus $\lim_{k\to\infty}\gamma_k=0$ due to (A.17). With (A.23), this gives the desired result. □

A.1.2. For the Proofs of Theorem 3.4.6 and Theorem 3.4.8

Lemma A.1.3. *Suppose that $f(w,z)$ is scale invariant in $w\in\mathbb{R}^{d_w}$ for each $z\in\mathbb{R}^{d_z}$ and twice continuously differentiable on an open set containing $\partial\mathcal{B}_{d_w}\times\partial\mathcal{B}_{d_z}$. Let (w^*,z^*) be a point satisfying*

$$\nabla_w f(w^*,z^*)=\lambda_w^*w^*,\quad \lambda_w^*>\bar{\lambda}_2^w=\max_{2\leq i\leq d_w}|\lambda_i^w|,\quad w^*=v_1^w$$

where (λ_i^w,v_i^w) is an eigen-pair of $\nabla_{ww}^2f(w^*,z^*)$. Then, for any $w\in\partial\mathcal{B}_{d_w}$ and $z\in\partial\mathcal{B}_{d_z}$, we have

$$\nabla_w f(w,z)^Tv_1^w=\lambda_w^*+(z-z^*)^T\nabla_{zw}^2f(w^*,z^*)w^*+\alpha^w(w,z)$$

and

$$\sum_{i=2}^{d_w}(\nabla_w f(w,z)^Tv_i^w)^2\leq\left(\bar{\lambda}_2^w\sqrt{1-(w^Tw^*)^2}+\nu^{wz}\|z-z^*\|+\beta^w(w,z)\right)^2$$

where

$$\alpha^w(w,z) = o\left(\left\|\begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix}\right\|\right), \quad \beta^w(w,z) = o\left(\left\|\begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix}\right\|\right).$$

Therefore, we have

$$1 - \frac{(\nabla_w f(w,z)^T w^*)^2}{\|\nabla_w f(w,z)\|^2} \leq \left(\frac{\bar{\lambda}_2^w}{\lambda_w^*}\sqrt{1-(w^T w^*)^2} + \frac{\nu^{wz}}{\lambda_w^*}\|z-z^*\| + \theta^w(w,z)\right)^2$$

where

$$\nu^{wz} = \|\nabla^2_{wz} f(w^*,z^*)\|, \quad \theta^w(w,z) = o\left(\left\|\begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix}\right\|\right).$$

Proof. Since $\nabla^2_{ww} f(w^*,z^*)$ is real and symmetric, without loss of generality, we assume that $\{v_1^w, \ldots, v_{d_w}^w\}$ forms an orthogonal basis in $\mathbb{R}^{d_w}$.

By Taylor expansion of $\nabla_w f(w,z)^T v_i^w$ at (w^*,z^*), we have

$$\nabla_w f(w,z)^T v_i^w = \nabla_x f(w^*,z^*)^T v_i^w + \begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix}^T \begin{bmatrix} \nabla^2_{ww} f(w^*,z^*) \\ \nabla^2_{zw} f(w^*,z^*) \end{bmatrix} v_i^w + R_i^w(w,z)$$

where

$$R_i^w(w,z) = o\left(\left\|\begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix}\right\|\right).$$

Using $\nabla_w f(w^*,z^*) = \lambda_w^* w^*$ and $w^* = v_1^w$, we have

$$\nabla_w f(w^*,z^*)^T v_1^w = \lambda_w^*, \quad (w-w^*)^T \nabla^2_{ww} f(w^*,z^*) v_1^w = -\lambda_1^w (1 - w_k^T w^*).$$

Therefore, we obtain

$$\nabla_w f(w,z)^T v_1^w = \lambda_w^* + (w-w^*)^T \nabla^2_{zw} f(w^*,z^*) w^* + \alpha^w(w,z) \tag{A.24}$$

where

$$\alpha^w(w,z) = R_1^w(w,z) - \lambda_1^w(1 - w^T w^*) = o\left(\left\| \begin{bmatrix} w - w^* \\ z - z^* \end{bmatrix} \right\|\right).$$

In the same way, for $2 \le i \le d_w$, we have

$$\nabla_w f(w^*, z^*)^T v_i^w = \lambda_w^*(w^*)^T v_i^w = 0, \quad (w - w^*)^T \nabla_{ww}^2 f(w^*, z^*) v_i^w = \lambda_i^w w^T v_i^w,$$

resulting in

$$\nabla_w f(w,z)^T v_i^w = \lambda_i^w w^T v_i^w + (z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) v_i^w + R_i^w(w,z). \tag{A.25}$$

From (A.25), we obtain

$$\begin{aligned}
\sum_{i=2}^{d_w} (\nabla_w f(w,z)^T v_i^w)^2 = &\sum_{i=2}^{d_w} (\lambda_i^w)^2 (w^T v_i^w)^2 + \sum_{i=2}^{d_w} \left((z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) v_i^w\right)^2 \\
&+ \sum_{i=2}^{d_w} (R_i^w(w,z))^2 + 2 \sum_{i=2}^{d_w} \lambda_i^w (w^T v_i^w)(z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) v_i^w \\
&+ 2 \sum_{i=2}^{d_w} \lambda_i^w (w^T v_i^w) R_i^w(w,z) \\
&+ 2 \sum_{i=2}^{d_w} (z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) v_i^w R_i^w(w,z).
\end{aligned}$$

Since $\{v_1^w, \ldots, v_{d_w}^w\}$ forms an orthogonal basis in $\mathbb{R}^{d_w}$, with $w^* = v_1^w$ and $\|w\|^2 = 1$, we have

$$\sum_{i=2}^{d_w} (\lambda_i^w)^2 (w^T v_i^w)^2 \le (\bar{\lambda}_2^w)^2 \left(1 - (w^T w^*)^2\right)$$

and

$$\sum_{i=2}^{d_w} \left((z-z^*)^T \nabla^2_{zw} f(w^*,z^*) v_i^w\right)^2 \leq \|(z-z^*)^T \nabla^2_{zw} f(w^*,z^*)\|^2 \leq (\nu^{wz})^2 \|z-z^*\|^2.$$

Let $\bar{R}_2^w(w,z) = \max_{2\leq i\leq d_w} |R_i^w(w,z)|$. Note that

$$\bar{R}_2^w(w,z) = o\left(\left\| \begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix} \right\|\right).$$

Using the Cauchy-Shwartz inequality, we have

$$\sum_{i=2}^{d_w} \lambda_i^w (w^T v_i^w)(z-z^*)^T \nabla^2_{zw} f(w^*,z^*) v_i^w \leq \bar{\lambda}_2^w \nu^{wz} \|z-z^*\| \sqrt{1-(w^T w^*)^2}.$$

Also, we have

$$\sum_{i=2}^{d_w} \lambda_i^w (w^T v_i^w) R_i^w(w,z) \leq \bar{\lambda}_2^w \bar{R}_2^w(w,z) \sqrt{d_w} \sqrt{1-(w^T w^*)^2}$$

and

$$\sum_{i=2}^{d_w} R_i^w(w,z)(z-z^*)^T \nabla^2_{zw} f(w^*,z^*) v_i^w \leq \nu^{wz} \bar{R}_2^w(w,z) \sqrt{d_w} \|z-z^*\|.$$

Therefore, we obtain

$$\sum_{i=2}^{d_w} (\nabla_w f(w,z)^T v_i^w)^2 \leq \left(\bar{\lambda}_2^w \sqrt{1-(w^T w^*)^2} + \nu^{wz}\|z-z^*\| + \beta^w(w,z)\right)^2 \tag{A.26}$$

where

$$\beta^w(w,z) = \bar{R}_2^w(w,z)\sqrt{d_w} = o\left(\left\| \begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix} \right\|\right).$$

Since $\{v_1^w, \ldots, v_{d_w}^w\}$ forms an orthogonal basis in $\mathbb{R}^{d_w}$ and $|w^T w^*| \leq \|w\|\|w^*\| = 1$, we have

$$1 - \frac{(\nabla_w f(w,z)^T w^*)^2}{\|\nabla_w f(w,z)\|^2} \leq \frac{\sum_{i=2}^{d_w} (\nabla_w f(w,z)^T v_i^w)^2}{(\nabla_w f(w,z)^T v_1^w)^2}.$$

Using (A.24) and (A.26), we have

$$\frac{\sum_{i=2}^{d_w} (\nabla_w f(w,z)^T v_i^w)^2}{(\nabla_w f(w,z)^T v_1^w)^2} \leq \left(\frac{\bar{\lambda}_2^w}{\lambda_w^*} \sqrt{1 - (w^T w^*)^2} + \frac{\nu^{wz}}{\lambda_w^*} \|z - z^*\| + \theta^w(w,z) \right)^2$$

where

$$\theta^w(w,z) = \frac{\beta^w(w,z)}{\lambda_w^*} - \left(\frac{\bar{\lambda}_2^w \sqrt{1 - (w^T w^*)^2} + \nu^{wz} \|z - z^*\| + \sqrt{d_w} \beta^w(w,z)}{\lambda_w^*} \right)$$
$$\cdot \left(\frac{(z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) w^* + \beta^w(w,z)}{\lambda_w^* + (z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) w^* + \beta^w(w,z)} \right).$$

Since

$$|(z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) w^*| \leq \nu^{wz} \|z - z^*\|,$$

we have

$$|(z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) w^*| \sqrt{1 - (w^T w^*)^2} \leq \frac{1}{2} \left(1 - (w^T w^*)^2\right) + \frac{1}{2} (\nu^{wz})^2 \|z - z^*\|^2$$

and

$$\nu^{wz} |(z - z^*)^T \nabla_{zw}^2 f(w^*, z^*) w^*| \|z - z^*\| \leq (\nu^{wz})^2 \|z - z^*\|^2.$$

From

$$1 - (w^T w^*)^2 = o\left(\left\| \begin{bmatrix} w - w^* \\ z - z^* \end{bmatrix} \right\| \right), \quad \|z - z^*\|^2 = o\left(\left\| \begin{bmatrix} w - w^* \\ z - z^* \end{bmatrix} \right\| \right),$$

we finally obtain

$$\theta^w(w,z) = o\left(\left\|\begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix}\right\|\right).$$

This completes the proof. □

Lemma A.1.4. *Suppose that $f(w,z)$ is μ-strongly concave in $z \in \mathbb{R}^{d_z}$ with an L-Lipschitz continuous $\nabla_z f(w,z)$ for each $w \in \partial\mathcal{B}_{d_w}$ and three-times continously differentiable with respect to x and y on an open set containing $\partial\mathcal{B}_{d_w}$ and $\mathbb{R}^{d_z}$, respectively. Let (w^*, z^*) be a point such that $\nabla_z f(w^*, z^*) = 0$. Then, for any $w \in \partial\mathcal{B}_{d_w}$ and $z \in \partial\mathcal{B}_{d_z}$, with $\alpha = 2/(L+\mu)$, we have*

$$\|z + \alpha\nabla_z f(w,z) - z^*\| \leq \left(\frac{2\nu^{zw}}{L+\mu}\right)\|w-w^*\| + \left(\frac{L-\mu}{L+\mu}\right)\|z-z^*\| + \theta^z(w,z) \tag{A.27}$$

where

$$\nu^{zw} = \|\nabla^2_{zw} f(w^*, z^*)\|, \quad \theta^z(w,z) = o\left(\left\|\begin{bmatrix} w-w^* \\ z-z^* \end{bmatrix}\right\|\right).$$

Proof. Let $\nabla_{z,i} f$ be the i^{th} coordinate of $\nabla_z f$ and

$$H_{z,i} = \begin{bmatrix} H^{ww}_{z,i} & H^{wz}_{z,i} \\ H^{zw}_{z,i} & H^{zz}_{z,i} \end{bmatrix}$$

be the Hessian of $\nabla_{z,i} f$. By Taylor expansion of $\nabla_{z,i} f(w,z)$ at (w^*, z), we have

$$\nabla_{z,i} f(w,z) = \nabla_{z,i} f(w^*, z) + \nabla^2_{zw,\cdot i} f(w^*, z)^T (w-w^*) + R^z_i(w,z) \tag{A.28}$$

where $\nabla^2_{zw,\cdot i} f(w^*, z) = \nabla_w \nabla_{z,i} f(w^*, z)$ denotes the i^{th} column of $\nabla^2_{zw} f(w^*, z)$ and

$$R^z_i(w,z) = \frac{1}{2}(w-w^*)^T H^{ww}_{z,i}(\hat{w}^i, z)(w-w^*), \quad \hat{w}^i \in \mathcal{N}(w, w^*). \tag{A.29}$$

Also, from f being three-times continuously differentiable, we have

$$\nabla^2_{zw,\cdot i} f(w^*, z) = \nabla^2_{zw,\cdot i} f(w^*, z^*) + H^{wz}_{z,i}(w^*, \hat{z}^i)(z - z^*), \quad \hat{z}^i \in \mathcal{N}(z, z^*). \tag{A.30}$$

Since

$$\begin{aligned} |(z - z^*)^T H^{zw}_{z,i}(w^*, \hat{z}^i)(w - w^*)| &\leq \|H^{zw}_{z,i}(w^*, \hat{z}^i)\| \|w - w^*\| \|z - z^*\| \\ &\leq \frac{1}{2} \|H^{zw}_{z,i}(w^*, \hat{z}^i)\| \left(\|w - w^*\|^2 + \|z - z^*\|^2 \right), \end{aligned}$$

we have

$$(z - z^*)^T H^{wz}_{z,i}(w^*, \hat{z}^i)(w - w^*) = o\left(\left\| \begin{bmatrix} w - w^* \\ z - z^* \end{bmatrix} \right\| \right). \tag{A.31}$$

By (A.28), (A.29), (A.30), and (A.31), we have

$$\nabla_z f(w, z) = \nabla_z f(w^*, z) + \nabla^2_{zw} f(w^*, z^*)(w - w^*) + \bar{R}^z(w, z) \tag{A.32}$$

where

$$\bar{R}^z_i(w, z) = R^z_i(w, z) + (z - z^*)^T H^{zw}_{z,i}(w^*, \hat{z}^i)(w - w^*) = o\left(\left\| \begin{bmatrix} w - w^* \\ z - z^* \end{bmatrix} \right\| \right).$$

Using (A.32), we have

$$z + \alpha \nabla_z f(w, z) - z^* = z - z^* + \alpha \nabla_z f(w^*, z) + \alpha \nabla^2_{zw} f(w^*, z^*)(w - w^*) + \bar{R}^z(w, z),$$

resulting in

$$\begin{aligned}\|z+\alpha\nabla_z f(w,z)-z^*\| \le \|z-z^*+\alpha\nabla_z f(w^*,z)\| \\ +\alpha\|\nabla^2_{zw} f(w^*,z^*)(w-w^*)\|+\|\bar{R}^z(w,z)\|.\end{aligned} \tag{A.33}$$

Since $-f(w^*,z)$ is μ-strongly convex in z with an L-Lipschitz continuous gradient $-\nabla_z f(w^*,z)$, by theory of convex optimization [**13**, p. 270], we have

$$\|z-z^*+\alpha\nabla_z f(w^*,z)\| \le \left(\frac{L-\mu}{L+\mu}\right)\|z-z^*\| \tag{A.34}$$

due to $\alpha = 2/(L+\mu)$. Also, we have

$$\alpha\|\nabla^2_{zw} f(w^*,z^*)(w-w^*)\| \le \left(\frac{2\nu^{zw}}{L+\mu}\right)\|w-w^*\|. \tag{A.35}$$

Plugging (A.34), (A.35) into (A.33), we finally obtain

$$\|z-z^*+\alpha\nabla_z f(w^*,z)\| \le \left(\frac{L-\mu}{L+\mu}\right)\|z-z^*\|+\left(\frac{2\nu^{zw}}{L+\mu}\right)\|w-w^*\|+\theta^z(w,z)$$

where

$$\theta^z(w,z) = \|\bar{R}^z(w,z)\| = o\left(\left\|\begin{bmatrix} w-w^* \\ z-z^*\end{bmatrix}\right\|\right).$$

□

Lemma A.1.5. *Let M be a 2×2 matrix such that*

$$M = \begin{bmatrix} a & e/b \\ e/c & d\end{bmatrix}$$

for some $a > 0, b > 0, c > 0, d \geq 0, e \geq 0$ *and let* ρ *be the largest absolute eigenvalue of* M*. Then, there exists a sequence* ω_t *such that*

$$\|M^k\| = \prod_{t=0}^{k-1}(\rho + \omega_t) \quad \textit{and} \quad lim_{t\to\infty}\,\omega_t = 0.$$

Proof. The characteristic equation reads

$$\det(M - \lambda I) = \lambda^2 - \lambda(a + d) + ad - \frac{e^2}{bc} = 0$$

with the discriminant of

$$(a - d)^2 + \frac{4e^2}{bc} \geq 0.$$

Thus, all eigenvalues are real.

First, we consider the case when $\det(M - \lambda I) = 0$ has a double root. We obtain the condition for a double root as

$$(a - d)^2 + \frac{4e^2}{bc} = 0.$$

Since $b > 0$ and $c > 0$, this implies

$$a = d, \quad e = 0.$$

Therefore, $M = aI$ and $\rho = a$. From $M^k = a^k I$, we have

$$\|M^k\| = \sqrt{a^{2k}} = \rho^k,$$

resulting in

$$\omega_k = \frac{\|M^{k+1}\|}{\|M^k\|} - \rho = \rho - \rho = 0$$

for all $k \geq 0$.

Next, we consider the case when M has two distinct eigenvalues λ_1 and λ_2. Since $a + d > 0$, we have $\lambda_1 + \lambda_2 > 0$. Without loss of generality, assume $\lambda_1 > \lambda_2$. Then, $\rho = \lambda_1$. Let v_1 and v_2 be corresponding eigenvectors of λ_1 and λ_2, respectively. Since v_1 and v_2 are linearly independent we can represent each column of M as a linear combination of v_1 and v_2 as

$$M = [\alpha_1 v_1 + \beta_1 v_2 \quad \alpha_2 v_1 + \beta_2 v_2].$$

By repeatedly multiplying M, we obtain

$$M^k = [\alpha_1 \lambda_1^{k-1} v_1 + \beta_1 \lambda_2^{k-1} v_2 \quad \alpha_2 \lambda_1^{k-1} v_1 + \beta_2 \lambda_2^{k-1} v_2].$$

Let $C^k = (M^k)^T M^k$. Then, we have

$$C_{11}^k = \alpha_1^2 \lambda_1^{2(k-1)} + \beta_1^2 \lambda_2^{2(k-1)} + 2\alpha_1 \beta_1 (\lambda_1 \lambda_2)^{k-1} v_1^T v_2$$

$$C_{22}^k = \alpha_2^2 \lambda_1^{2(k-1)} + \beta_2^2 \lambda_2^{2(k-1)} + 2\alpha_2 \beta_2 (\lambda_1 \lambda_2)^{k-1} v_1^T v_2$$

and

$$C_{12}^k = \alpha_1 \alpha_2 \lambda_1^{2(k-1)} + \beta_1 \beta_2 \lambda_2^{2(k-1)} + (\alpha_1 \beta_2 + \alpha_2 \beta_1)(\lambda_1 \lambda_2)^{k-1} v_1^T v_2, \quad C_{21}^k = C_{12}^k.$$

Since

$$C_{11}^k \geq \alpha_1^2 \lambda_1^{2(k-1)} + \beta_1^2 \lambda_2^{2(k-1)} - 2\alpha_1 \beta_1 (\lambda_1 \lambda_2)^{k-1} = \left(\alpha_1 \lambda_1^{k-1} - \beta_1 \lambda_2^{k-1}\right)^2 \geq 0$$

and

$$C_{22}^k \geq \alpha_2^2 \lambda_1^{2(k-1)} + \beta_2^2 \lambda_2^{2(k-1)} - 2\alpha_2\beta_2(\lambda_1\lambda_2)^{k-1} = \left(\alpha_2\lambda_1^{k-1} - \beta_2\lambda_2^{k-1}\right)^2 \geq 0,$$

we have

$$\|M^k\| = \sqrt{\frac{1}{2}\left[C_{11}^k + C_{22}^k + \sqrt{\left(C_{11}^k - C_{22}^k\right)^2 + 4(C_{12}^k)^2}\right]},$$

leading to

$$\frac{\|M^{k+1}\|}{\|M^k\|} = \sqrt{\frac{C_{11}^{k+1} + C_{22}^{k+1} + \sqrt{\left(C_{11}^{k+1} - C_{22}^{k+1}\right)^2 + 4(C_{12}^{k+1})^2}}{C_{11}^k + C_{22}^k + \sqrt{\left(C_{11}^k - C_{22}^k\right)^2 + 4(C_{12}^k)^2}}}.$$

From

$$\lim_{k\to\infty} \frac{C_{11}^k}{\lambda_1^{2(k-1)}} = \alpha_1^2, \quad \lim_{k\to\infty} \frac{C_{22}^k}{\lambda_1^{2(k-1)}} = \alpha_2^2, \quad \lim_{k\to\infty} \frac{C_{12}^k}{\lambda_1^{2(k-1)}} = \lim_{k\to\infty} \frac{C_{21}^k}{\lambda_1^{2(k-1)}} = \alpha_1\alpha_2,$$

we obtain

$$\lim_{k\to\infty} \frac{\|M^{k+1}\|}{\|M^k\|} = \sqrt{\lambda_1^2} = \rho.$$

From

$$\lim_{k\to\infty} \omega_k = \lim_{k\to\infty} \frac{\|M^{k+1}\|}{\|M^k\|} - \rho = \rho - \rho = 0,$$

we obtain the desired result. □

A.2. Chapter 4

In the proofs below, for $\alpha, \beta \geq 0$, we let $Y_t(A, \beta)$ and $Z_t(A, \beta)$ be matrix polynomials such that

$$Y_t(A,\beta) = 2AY_{t-1}(A,\beta) - \beta Y_{t-2}(A,\beta),\ t \geq 2,\ Y_1(A,\beta) = A,\ Y_0(A,\beta) = I, \tag{A.36}$$

$$Z_t(A,\beta) = 2AZ_{t-1}(A,\beta) - \beta Z_{t-2}(A,\beta),\ t \geq 2,\ Z_1(A,\beta) = 2A,\ Z_0(A,\beta) = I. \tag{A.37}$$

and let $y_t(\alpha, \beta)$ and $z_t(\alpha, \beta)$ be recurrence polynomials such that

$$y_t(\alpha,\beta) = \sqrt{\alpha}y_{t-1}(\alpha,\beta) - \beta y_{t-2}(\alpha,\beta),\ t \geq 2,\ y_1(\alpha,\beta) = \frac{\sqrt{\alpha}}{2},\ y_0(\alpha,\beta) = 1, \tag{A.38}$$

$$z_t(\alpha,\beta) = \sqrt{\alpha}z_{t-1}(\alpha,\beta) - \beta z_{t-2}(\alpha,\beta),\ t \geq 2,\ z_1(\alpha,\beta) = \sqrt{\alpha},\ z_0(\alpha,\beta) = 1. \tag{A.39}$$

For a sequence of matrices $B_0, B_1, B_2, \cdots$, let

$$\prod_{i=j}^{k} B_i = \begin{cases} B_j B_{j-1} \cdots B_k & \text{if } j \geq k \\ I, & \text{otherwise} \end{cases}.$$

Since the eigenvectors $u_1, u_2, \ldots, u_d$ form an orthogonal basis, we frequently use the fact that for $w \in \mathbb{R}^d$, we have $\|x\|^2 = \sum_{k=1}^{d} (u_k^T x)^2$.

Lemma A.2.1. *Let $x \in \partial\mathcal{B}_d$. For $t \geq 0$, we have*

$$\|P\left[(1-\eta)I + \eta C\right]^t x\|^2 \leq 2(1-\eta+\eta\lambda_1)^{2t}(1-(u_1^T x)^2), \tag{A.40a}$$

$$\|PY_t((1-\eta)I + \eta C, \beta(\eta))x\|^2 \leq 4(1-(u_1^T x)^2)p_t(\alpha_1(\eta), \beta(\eta)), \tag{A.40b}$$

$$\|Z_t((1-\eta)I + \eta C, \beta(\eta))\|^2 \leq q_t(\alpha_1(\eta), \beta(\eta)). \tag{A.40c}$$

Proof. Since $u_1, u_2, \cdots, u_d$ forms an orthogonal basis in $\mathbb{R}^d$, we have $x = \sum_{k=1}^{d}(u_k^T x)u_k$. From that (λ_k, u_k) are eigenpairs of C, we have

$$[(1-\eta)I + \eta C]^t x = \sum_{k=1}^{d}(u_k^T x)(1-\eta+\eta\lambda_k)^t u_k. \tag{A.41}$$

From the definition of x and P in (4.6), we have $P = I - xx^T$. Since

$$\begin{aligned}
\|P\left[(1-\eta)I + \eta C\right]^t x\|^2 &= x^T\left[(1-\eta)I + \eta C\right]^t P^2\left[(1-\eta)I + \eta C\right]^t x \\
&= x^T\left[(1-\eta)I + \eta C\right]^t P\left[(1-\eta)I + \eta C\right]^t x \\
&= x^T\left[(1-\eta)I + \eta C\right]^t (I - xx^T)\left[(1-\eta)I + \eta C\right]^t x \\
&= \|\left[(1-\eta)I + \eta C\right]^t x\|^2 - \left(x^T\left[(1-\eta)I + \eta C\right]^t x\right)^2,
\end{aligned}$$

using (A.41), we have

$$\begin{aligned}
\|P\left[(1-\eta)I + \eta C\right]^t x\|^2 &= \sum_{k=1}^{d}(u_k^T x)^2(1-\eta+\eta\lambda_k)^{2t} - \Big(\sum_{k=1}^{d}(u_k^T x)^2(1-\eta+\eta\lambda_k)^t\Big)^2 \\
&\le (1-\eta+\eta\lambda_1)^{2t} - (u_1^T x)^4(1-\eta+\eta\lambda_1)^{2t} \\
&\le 2(1-(u_1^T x)^2)(1-\eta+\eta\lambda_1)^{2t}
\end{aligned}$$

where the last inequality follows from

$$1 - (u_1^T x)^4 = \big(1 + (u_1^T x)^2\big)\big(1 - (u_1^T x)^2\big) \le 2\big(1 - (u_1^T x)^2\big). \tag{A.42}$$

To prove (A.40b), we first show that

$$Y_t((1-\eta)I + \eta C, \beta(\eta))u_k = y_t(\alpha_k(\eta), \beta(\eta))u_k. \tag{A.43}$$

First, consider the cases when $t = 0$ and 1. For $t = 0$, we have $Y_0((1-\eta)I + \eta C, \beta(\eta))u_k = y_0(\alpha_k(\eta), \beta(\eta))u_k$. For $t = 1$, it follows that

$$Y_1((1-\eta)I + \eta C, \beta(\eta))u_k = (1 - \eta + \eta\lambda_k)u_k = \frac{\sqrt{\alpha_k(\eta)}}{2}u_k = y_1(\alpha_k(\eta), \beta(\eta))u_k.$$

Suppose that (A.43) holds for $t-1$ and $t-2$. Using the definition of Y_t in (A.36), we have

$$\begin{aligned}
&Y_t((1-\eta)I + \eta C, \beta(\eta))u_k \\
&= \left[2((1-\eta)I + \eta C)Y_{t-1}((1-\eta)I + \eta C, \beta(\eta)) - \beta(\eta)Y_{t-2}((1-\eta)I + \eta C, \beta(\eta))\right] u_k \\
&= \left[2(1 - \eta + \eta\lambda_k)y_{t-1}(\alpha_k(\eta), \beta(\eta)) - \beta(\eta)y_{t-2}(\alpha_k(\eta), \beta(\eta))\right] u_k \\
&= \left[\sqrt{\alpha_k(\eta)}y_{t-1}(\alpha_k(\eta), \beta(\eta)) - \beta(\eta)y_{t-2}(\alpha_k(\eta), \beta(\eta))\right] u_k \\
&= y_t(\alpha_k(\eta), \beta(\eta))u_k.
\end{aligned}$$

This completes the proof of (A.43).

Next, we show that

$$(y_t(\alpha_k(\eta), \beta(\eta))^2 = p_t(\alpha_k(\eta), \beta(\eta)). \tag{A.44}$$

For the base cases, we have

$$(y_0(\alpha_k(\eta), \beta(\eta))^2 = 1 = p_0(\alpha_k(\eta), \beta(\eta)), \quad (y_1(\alpha_k(\eta), \beta(\eta))^2 = \frac{\alpha_k}{4} = p_1(\alpha_k(\eta), \beta(\eta))$$

and

$$(y_2(\alpha_k(\eta), \beta(\eta))^2 = \left(\frac{\alpha(\eta)}{2} - \beta(\eta)\right)^2 = p_2(\alpha_k(\eta), \beta(\eta)).$$

Using the definition of y_t in (A.38) for t and $t-1$, we have

$$(y_t(\alpha_k(\eta),\beta(\eta)))^2 = (\sqrt{\alpha_k(\eta)}y_{t-1}(\alpha_k(\eta),\beta(\eta)) - \beta(\eta)y_{t-2}(\alpha_k(\eta),\beta(\eta)))^2$$
$$= \alpha_k(\eta)(y_{t-1}(\alpha_k(\eta),\beta(\eta)))^2 - 2\sqrt{\alpha_k(\eta)}\beta(\eta)y_{t-1}(\alpha_k(\eta),\beta(\eta))y_{t-2}(\alpha_k(\eta),\beta(\eta))$$
$$+ \beta(\eta)^2(y_{t-2}(\alpha_k(\eta),\beta(\eta)))^2$$

and

$$(y_{t-1}(\alpha_k(\eta),\beta(\eta)))^2$$
$$= \alpha_k(\eta)(y_{t-2}(\alpha_k(\eta),\beta(\eta)))^2 - 2\sqrt{\alpha_k(\eta)}\beta(\eta)y_{t-2}(\alpha_k(\eta),\beta(\eta))y_{t-3}(\alpha_k(\eta),\beta(\eta))$$
$$+ \beta(\eta)^2(y_{t-3}(\alpha_k(\eta),\beta(\eta)))^2.$$

Moreover, since

$$y_{t-1}(\alpha_k(\eta),\beta(\eta))y_{t-2}(\alpha_k(\eta),\beta(\eta))$$
$$= \sqrt{\alpha_k(\eta)}(y_{t-2}(\alpha_k(\eta),\beta(\eta)))^2 - \beta(\eta)y_{t-2}(\alpha_k(\eta),\beta(\eta))y_{t-3}(\alpha_k(\eta),\beta(\eta)),$$

we have

$$(y_t(\alpha_k(\eta),\beta(\eta)))^2 = (\alpha_k(\eta)-\beta(\eta))(y_{t-1}(\alpha_k(\eta),\beta(\eta)))^2$$
$$- \beta(\eta)(\alpha_k(\eta)-\beta(\eta))(y_{t-2}(\alpha_k(\eta),\beta(\eta)))^2 + \beta(\eta)^3(y_{t-3}(\alpha_k(\eta),\beta(\eta)))^2.$$

This proves (A.44).

Now, using (A.43), we have

$$Y_t((1-\eta)I+\eta C,\beta(\eta))x=\sum_{k=1}^{d} y_t(\alpha_k(\eta),\beta(\eta))(u_k^T x)u_k. \tag{A.45}$$

Since $u_1, u_2, \cdots, u_d$ form an orthogonal basis in $\mathbb{R}^d$, we have

$$\|Y_t((1-\eta)I+\eta C,\beta(\eta))x\|^2=\sum_{k=1}^{d}(y_t(\alpha_k(\eta),\beta(\eta)))^2(u_k^T x)^2=\sum_{k=1}^{d} p_t(\alpha_k(\eta),\beta(\eta))(u_k^T x)^2.$$

Using (A.52) and (A.54) in Lemma A.2.4, for $k \geq 2$, we have

$$p_t(\alpha_k(\eta),\beta(\eta)) \leq p_t(\alpha_1(\eta),\beta(\eta)) \tag{A.46}$$

Since $\sum_{k=1}^{d}(u_k^T x)^2=1$, we have

$$\|Y_t((1-\eta)I+\eta C,\beta(\eta))x\|^2 \leq p_t(\alpha_1(\eta),\beta(\eta)).$$

Moreover, using $(u_1^T x)^2 \leq 1$ and (A.45), we obtain

$$\begin{aligned}
&\left(w^T Y_t((1-\eta)I+\eta C,\beta(\eta))x\right)^2\\
&=\left(y_t(\alpha_1(\eta),\beta(\eta))(u_1^T x)^2+\sum_{k=2}^{d} y_t(\alpha_k(\eta),\beta(\eta))(u_k^T x)^2\right)^2\\
&\geq (y_t(\alpha_1(\eta),\beta(\eta)))^2(u_1^T x)^4-2y_t(\alpha_1(\eta),\beta(\eta))\sum_{k=2}^{d}|y_t(\alpha_k(\eta),\beta(\eta))|(u_k^T x)^2\\
&\geq (y_t(\alpha_1(\eta),\beta(\eta)))^2(u_1^T x)^4-2(y_t(\alpha_1(\eta),\beta(\eta)))^2(1-(u_k^T x)^2)
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\|PY_t((1-\eta)I+\eta C,\beta(\eta))x\|^2 \\
&= \|Y_t((1-\eta)I+\eta C,\beta(\eta))x\|^2 - \left(x^T Y_t((1-\eta)I+\eta C,\beta(\eta))x\right)^2 \\
&\leq (y_t(\alpha_1(\eta),\beta(\eta)))^2(1-(u_1^T x)^4) + 2(y_t(\alpha_1(\eta),\beta(\eta)))^2(1-(u_k^T x)^2) \\
&\leq 4(y_t(\alpha_1(\eta),\beta(\eta)))^2(1-(u_k^T x)^2)
\end{aligned}$$

where the last inequality follows from (A.42).

Lastly, we prove (A.40c). In the same way we prove (A.43) and (A.44), we can show that

(A.47)

$$Z_t((1-\eta)I+\eta C,\beta(\eta))u_k = z_t(\alpha_k(\eta),\beta(\eta))u_k, \quad (z_t(\alpha_k(\eta),\beta(\eta))^2 = q_t(\alpha_k(\eta),\beta(\eta)).$$

Using (A.53) and (A.54) in Lemma A.2.4, for $k \geq 2$, we have

(A.48)
$$q_t(\alpha_k(\eta),\beta(\eta)) \leq q_t(\alpha_1(\eta),\beta(\eta)).$$

Using (A.47), we have

$$x^T Z_t((1-\eta)I+\eta C,\beta(\eta))x = \sum_{k=1}^{d} z_t(\alpha_k(\eta),\beta(\eta))(u_k^T x)^2 \leq \sum_{k=1}^{d} |z_t(\alpha_k(\eta),\beta(\eta))|(u_k^T x)^2.$$

Moreover, using (A.48) and the fact that $\sum_{k=1}^{d}(u_k^T x)^2 = 1$, we have

$$\sum_{k=1}^{d} |z_t(\alpha_k(\eta),\beta(\eta))|(u_k^T x)^2 \leq |z_t(\alpha_1(\eta),\beta(\eta))| \sum_{k=1}^{d}(u_k^T x)^2 = |z_t(\alpha_1(\eta),\beta(\eta))|.$$

This results in

$$x^T Z_t((1-\eta)I + \eta C, \beta(\eta))x \leq |z_t(\alpha_1(\eta), \beta(\eta))|,$$

leading to

$$\|Z_t((1-\eta)I + \eta C, \beta(\eta))\|^2 \leq |z_t(\alpha_1(\eta), \beta(\eta))|^2 = q_t(\alpha_1(\eta), \beta(\eta)).$$

This complets the proof. □

Lemma A.2.2. *Let x be a vector in $\mathbb{R}^d$ and let M be a $d \times d$ symmetric matrix. Then, we have $x^T M x \leq \|M\|\|x\|^2$.*

Proof. By the cyclic property of the trace, we have

$$x^T M x = \mathrm{Tr}[x^T M x] = \mathrm{Tr}[M x x^T].$$

Since xx^T is positive semi-definite, we have $\mathrm{Tr}[Mxx^T] \leq \|M\|\mathrm{Tr}[xx^T]$. Again, by the cyclic property of the trace, we finally have

$$x^T M x \leq \|M\|\mathrm{Tr}[xx^T] = \|M\|\mathrm{Tr}[x^T x] = \|M\|\|x\|^2.$$

□

Lemma A.2.3. *Let A_i and B_i be $d \times d$ matrices for $i = 0, \cdots, t-1$. Then, we have*

(A.49)
$$\prod_{i=t-1}^{0}(A_i + B_i) = (A_{t-1} + B_{t-1}) \cdots (A_0 + B_0) = \prod_{i=t-1}^{0} A_i + \sum_{i=0}^{t-1}\left[\prod_{j=t-1}^{i+1}(A_j + B_j)B_i \prod_{k=i-1}^{0} A_k\right].$$

Proof. We prove the statement by induction. For $t = 1$, we have

$$\prod_{i=0}^{0} A_i + \sum_{i=0}^{0} \left[\prod_{j=0}^{i+1} (A_j + B_j) B_i \prod_{k=i-1}^{0} A_k \right] = A_0 + \left[\prod_{j=0}^{1} (A_j + B_j) B_0 \prod_{k=-1}^{0} A_k \right] = A_0 + B_0,$$

which proves the base case. Next, suppose that we have (A.49) for $t - 2$. Then, we have

$$\begin{aligned}
\prod_{i=t-1}^{0} (A_i + B_i) &= (A_{t-1} + B_{t-1}) \prod_{i=t-2}^{0} (A_i + B_i) \\
&= (A_{t-1} + B_{t-1}) \left(\prod_{i=t-2}^{0} A_i + \sum_{i=0}^{t-2} \left[\prod_{j=t-2}^{i+1} (A_j + B_j) B_i \prod_{k=i-1}^{0} A_k \right] \right) \\
&= \prod_{i=t-1}^{0} A_i + B_{t-1} \prod_{i=t-2}^{0} A_i + \left(\sum_{i=0}^{t-2} \left[\prod_{j=t-1}^{i+1} (A_j + B_j) B_i \prod_{k=i-1}^{0} A_k \right] \right) \\
&= \prod_{i=t-1}^{0} A_i + \sum_{i=0}^{t-1} \left[\prod_{j=t-1}^{i+1} (A_j + B_j) B_i \prod_{k=i-1}^{0} A_k \right].
\end{aligned}$$

This completes the proof. □

Lemma A.2.4. *Let x_t be a sequence of real numbers such that*

$$x_t = (\alpha - \beta) x_{t-1} - \beta(\alpha - \beta) x_{t-2} + \beta^3 x_{t-3} + L_{t-1} + \beta L_{t-2}$$

for $t \geq 3$ and $x_0 = L_0$, $x_1 = \frac{\alpha}{4} L_0$, $x_2 = \left(\frac{\alpha}{2} - \beta\right)^2 L_0 + L_1$. Then, we have

$$x_t = p_t(\alpha, \beta) L_0 + \sum_{r=1}^{t-1} q_{t-r-1}(\alpha, \beta) L_r. \tag{A.50}$$

Moreover, for $t \geq 0$, we have

- *if* $0 \le \alpha = 4\beta$,

$$p_t(4\beta, \beta) = \beta^t \ge 0, \quad q_t(4\beta, \beta) = (t+1)^2\beta^t \ge 0, \tag{A.51}$$

- *if* $0 \le 4\beta < \alpha$,

$$p_t(\alpha, \beta) = \left[\frac{1}{2}\left(\frac{\sqrt{\alpha}}{2} + \frac{\sqrt{\alpha - 4\beta}}{2}\right)^t + \frac{1}{2}\left(\frac{\sqrt{\alpha}}{2} - \frac{\sqrt{\alpha - 4\beta}}{2}\right)^t\right]^2 > p_t(4\beta, \beta) \ge 0, \tag{A.52}$$

$$q_t(\alpha, \beta) = \frac{1}{\alpha - 4\beta}\left[\left(\frac{\sqrt{\alpha}}{2} + \frac{\sqrt{\alpha - 4\beta}}{2}\right)^{t+1} - \left(\frac{\sqrt{\alpha}}{2} - \frac{\sqrt{\alpha - 4\beta}}{2}\right)^{t+1}\right]^2 > q_t(4\beta, \beta) \ge 0, \tag{A.53}$$

- *if* $0 \le \alpha < 4\beta$,

$$p_t(\alpha, \beta) \le p_t(4\beta, \beta), \quad q_t(\alpha, \beta) \le q_t(4\beta, \beta). \tag{A.54}$$

Proof. It is easy to check that x_0, x_1, and x_2 satisfy (A.50). Suppose that (A.50) holds for $t-1, t-2, t-3$. Then, we have

$$\begin{aligned} x_t &= (\alpha - \beta)x_{t-1} - \beta(\alpha - \beta)x_{t-2} + \beta^3 x_{t-3} + L_{t-1} + \beta L_{t-2} \\ &= p_t(\alpha, \beta)L_0 + L_{t-1} + \alpha L_{t-2} + (\alpha - \beta)^2 L_{t-3} + \sum_{r=1}^{t-4} q_{t-r-1}(\alpha, \beta)L_r \\ &= p_t(\alpha, \beta)L_0 + \sum_{r=1}^{t-1} q_{t-r-1}(\alpha, \beta)L_r. \end{aligned}$$

Therefore, (A.50) holds by induction.

Next, we prove (A.51), (A.52), (A.53) and (A.54). The characteristic equation of (4.9) is

$$r^3 - (\alpha - \beta)r^2 + \beta(\alpha - \beta)r - \beta^3 = 0. \tag{A.55}$$

If $0 \leq \alpha = 4\beta$, (A.55) has a cube root of $r = \beta$. From initial conditions (4.11) and (4.12), we obtain

$$p_t(4\beta, \beta) = \beta^t \geq 0, \quad q_t(4\beta, \beta) = (t+1)^2\beta^t \geq 0. \tag{A.56}$$

If $0 \leq 4\beta < \alpha$, the roots of (A.55) are

$$r = \beta, \frac{\alpha - 2\beta}{2} + \frac{\sqrt{\alpha^2 - 4\alpha\beta}}{2}, \frac{\alpha - 2\beta}{2} - \frac{\sqrt{\alpha^2 - 4\alpha\beta}}{2}.$$

With initial conditions (4.11), we obtain

$$p_t(\alpha, \beta) = \frac{1}{4}\left(\frac{\alpha - 2\beta}{2} + \frac{\sqrt{\alpha^2 - 4\alpha\beta}}{2}\right)^t + \frac{1}{4}\left(\frac{\alpha - 2\beta}{2} - \frac{\sqrt{\alpha^2 - 4\alpha\beta}}{2}\right)^t + \frac{1}{2}\beta^t$$

Using the fact that $\alpha > 4\beta$ and the arithmetic-geometric mean inequality, we have

$$p_t(\alpha, \beta) > \beta^t \geq 0.$$

Moreover, we can further write $p_t(\alpha, \beta)$ as

$$p_t(\alpha, \beta) = \left[\frac{1}{2}\left(\frac{\sqrt{\alpha}}{2} + \frac{\sqrt{\alpha - 4\beta}}{2}\right)^t + \frac{1}{2}\left(\frac{\sqrt{\alpha}}{2} - \frac{\sqrt{\alpha - 4\beta}}{2}\right)^t\right]^2$$

by expanding this expression.

On the other hand, using (4.12), we have

$$\begin{aligned} q_t(\alpha,\beta) &= \frac{1}{\alpha-4\beta}\left[\left(\frac{\alpha-2\beta}{2}+\frac{\sqrt{\alpha^2-4\alpha\beta}}{2}\right)^{t+1}+\left(\frac{\alpha-2\beta}{2}-\frac{\sqrt{\alpha^2-4\alpha\beta}}{2}\right)^{t+1}-2\beta^{t+1}\right] \\ &= \frac{1}{\alpha-4\beta}\left[\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t+1}-\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t+1}\right]^2 \geq 0. \end{aligned}$$

Using the fact that $A^{t+1}-B^{t+1}=(A-B)(A^t+A^{t-1}B+\cdots+B^t)$ for any $A,B\in\mathbb{R}$, we have

$$q_t(\alpha,\beta)=\left[\sum_{i=0}^{t}\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{i}\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t-i}\right]^2.$$

Again, using the arithmetic-geometric mean inequality and the fact that $\alpha>4\beta$, we have

$$q_t(\alpha,\beta)\geq\left[(t+1)\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t/2}\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t/2}\right]^2=(t+1)^2\beta^t=q_t(4\beta,\beta).$$

If $0\leq\alpha<4\beta$, the roots of (A.55) are

$$r=\beta,\frac{\alpha-2\beta}{2}+\frac{\sqrt{4\alpha\beta-\alpha^2}}{2}i,\frac{\alpha-2\beta}{2}-\frac{\sqrt{4\alpha\beta-\alpha^2}}{2}i.$$

Setting

$$\cos\theta_p=\frac{\alpha-2\beta}{2\beta},\quad \sin\theta_p=\frac{\sqrt{4\alpha\beta-\alpha^2}}{2\beta}$$

it is easy to verify that

$$\begin{aligned} p_t(\alpha,\beta) &= \frac{1}{4}\beta^t\Big[\cos\,\theta_p + i\,\sin\,\theta_p\Big]^t + \frac{1}{4}\beta^t\Big[\cos\,\theta_p - i\,\sin\,\theta_p\Big]^t + \frac{1}{2}\beta^t \\ &= \frac{1}{4}(e^{i\theta t} + e^{-i\theta t})\beta^t + \frac{1}{2}\beta^t \\ &= \frac{1}{4}|e^{i\theta t} + e^{-i\theta t}|\beta^t + \frac{1}{2}\beta^t \\ &\le \frac{1}{4}(|e^{i\theta t}| + |e^{-i\theta t}|)\beta^t + \frac{1}{2}\beta^t \\ &= \beta^t. \end{aligned}$$

Moreover, with

$$\cos\,\theta_q = \frac{\alpha-2\beta}{2\beta}, \quad \sin\,\theta_q = \frac{\sqrt{4\alpha\beta-\alpha^2}}{2\beta}, \quad \cos\,\phi_q = 1-\frac{\alpha}{2\beta}, \quad \sin\,\phi_q = -\frac{\sqrt{4\alpha\beta-\alpha^2}}{2\beta},$$

it can be seen by using elementary calculus that

$$q_t(\alpha,\beta) = \left[\frac{2\beta}{4\beta-\alpha} + \frac{2\beta}{4\beta-\alpha}\cos(\phi_q + t\theta_q)\right]\beta^t. \tag{A.57}$$

Let

$$Q(t) = \frac{q_t(4\beta,\beta) - q_t(\alpha,\beta)}{\beta^t}.$$

Then, from (4.9) and (4.11), we have

(A.58)

$$Q(0) = 0, \quad Q(1) = \frac{4\beta-\alpha}{\beta}, \quad Q(2) = \frac{(4\beta-\alpha)(2\beta+\alpha)}{\beta^2}, \quad Q(3) = \frac{(\alpha^2+4\beta^2)(4\beta-\alpha)}{\beta^3}$$

resulting in

(A.59)
$$Q(2) - Q(0) = \frac{(4\beta - \alpha)(2\beta + \alpha)}{\beta^2} \geq 0, \quad Q(3) - Q(1) = \frac{(\alpha^2 + 3\beta^2)(4\beta - \alpha)}{\beta^3} \geq 0.$$

In order to show $Q(t) \geq 0$ for $t \geq 0$, we prove $Q(t+2) - Q(t) \geq 0$ for $t \geq 0$. Using (A.56), (A.57) and standard trigonometric equalities, it follows that

$$Q(t+2) - 2Q(t) + Q(t-2) = 8 + \frac{2\alpha}{\beta}\cos(\phi_q + t\theta_q).$$

In turn, we have

$$\begin{aligned} Q(t+2) - Q(t) &= Q(t) - Q(t-2) + 8 + \frac{2\alpha}{\beta}\cos(\phi_q + t\theta_q) \\ &\geq Q(t) - Q(t-2) + 8 - \frac{2\alpha}{\beta} \\ &= Q(t) - Q(t-2) + \frac{2(4\beta - \alpha)}{\beta} \\ &\geq Q(t) - Q(t-2). \end{aligned} \tag{A.60}$$

From (A.58), (A.59), and (A.60), for $t \geq 0$, we obtain $Q(t) \geq 0$ implying

$$q_t(\alpha, \beta) \leq q_t(4\beta, \beta).$$

□

Lemma A.2.5. *If $\alpha > 4\beta \geq 0$, then for $0 \leq t_1 < t_2$, we have*

$$q_{t_1}(\alpha, \beta) \cdot q_{t_2}(\alpha, \beta) \leq \left(\frac{1}{\alpha - 4\beta}\right) q_{t_1+t_2+1}(\alpha, \beta).$$

Proof. From (A.53) in Lemma A.2.4, we have

$$\begin{aligned}
&q_{t_1}(\alpha,\beta)\cdot q_{t_2}(\alpha,\beta)\\
&=\left(\frac{1}{\alpha-4\beta}\right)^2\left[\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+1}-\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+1}\right]^2\\
&\quad\cdot\left[\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_2+1}-\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_2+1}\right]^2.
\end{aligned}$$

Since

$$0\le\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}<\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2},$$

we have

$$\begin{aligned}
&\left[\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+1}-\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+1}\right]\\
&\quad\cdot\left[\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_2+1}-\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_2+1}\right]\\
&=\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+t_2+2}-\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+1}\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_2+1}\\
&\quad-\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+1}\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_2+1}+\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+t_2+2}\\
&\le\left(\frac{\sqrt{\alpha}}{2}+\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+t_2+2}-\left(\frac{\sqrt{\alpha}}{2}-\frac{\sqrt{\alpha-4\beta}}{2}\right)^{t_1+t_2+2}.
\end{aligned}$$

Therefore, we have

$$q_{t_1}(\alpha,\beta)\cdot q_{t_2}(\alpha,\beta)\le\left(\frac{1}{\alpha-4\beta}\right)q_{t_1+t_2+1}(\alpha,\beta).$$

This completes the proof. □