國立臺灣大學理學院應用數學科學研究所
碩士論文
Institute of Applied Mathematical Sciences
College of Science
National Taiwan University
Master Thesis

整合多重隨機奇異值分解與理論分析
Theoretical and Performance Analysis for Integrated Randomized Singular Value Decomposition

張大衛
Da-Wei Chang

指導教授：王偉仲博士
Advisor: Weichung Wang, Ph.D.

中華民國 106 年 7 月
July, 2017

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

國立臺灣大學碩士學位論文
口試委員會審定書

整合多重隨機奇異值分解與理論分析
Theoretical and Performance Analysis for Integrated Randomized Singular Value Decomposition

本論文係張大衛君 (R04246002) 在國立臺灣大學應用數學科學研究所完成之碩士學位論文，於民國 106 年 7 月 26 日承下列考試委員審查通過及口試及格，特此證明

口試委員：王偉仲

陳定立

陳素雲

所　長：

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

誌謝

在最一開始，我想感謝與這篇論文相關的所有人，沒有你們的協助，我無法將這篇論文完成。

感謝我的指導教授王偉仲老師，教導我該如何進行研究，定期開會討論進度，平常也總是開著辦公室的門，當遭遇問題，或是有新的進展時，都可以一起討論，並給予建設性的回應，除了學術討論以外，也對寫作與報告的模式進行指導，讓我對於論文的寫作有更深入的認識。感謝中研院統計所陳素雲老師與陳定立老師讓我參與整合隨機奇異值分解的這項研究，並且在理論方面提供指導與協助。感謝我的同學楊慕與林建燿，陪伴我一起討論，並且提出了許多新的點子與想法。感謝 R439 研究室的所有同學，營造出歡樂的研究氣氛，推動研究的推展，並且舒緩了研究不順利時的煩躁感。感謝我的家人在這段時間裡在生活與心理上的支持，使我可以無憂無慮，專心一致地完成這篇論文。

最後，再次感謝所有與這篇論文相關的所有人，從一開始接觸問題，到現在完成這份論文，因為有敬愛的師長、同學與家長的支持，才能讓這篇論文順利誕生。

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Acknowledgements

First of all, I would like to thank my advisor Professor Weichung Wang of the Institute of Applied Mathematical Science at National Taiwan University. When I ran into a trouble of the research, Prof. Wang always opened his office door and discussed with me. He also spent a lot of time teaching me the skill for academic writing and presentation. Without his passionate guidance, this thesis would not have been successfully conducted.

I would also like to thank the collaborators of my advisor Dr. Su-Yun Huang and Dr. Ting-Li Chen of the Institute of Statistical Science at Academia Sinica. They allowed me to join this research and guided me in theoretical research. With their aid, the theory in this thesis could be vigorously constructed.

At the end, I would like to acknowledge my classmates, Mu Yang and Chienyao Lin. I am very grateful for their comments about this research. Some analysis and methods in this research are even inspired by their interesting ideas.

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

摘要

維度降低和特徵提取是大數據時代的重要技術，此二技術可以降低數據維數並降低進一步分析數據的計算成本。低秩奇異值分解（low-rank SVD）是這些技術的關鍵部分。為了更快地計算低秩奇異值分解，一些研究提出可以使用隨機抽取子空間的方法來獲得近似結果。在這項研究中，我們提出了一種新的概念，將隨機算法的結果進行整合以獲得更準確的近似值，稱為整合奇異值分解。我們通過理論和數值實驗來分析演算法的性質，以及不同的整合方法。整合方法的架構是有條件的優化問題，其具有唯一的局部極小值。整合子空間將透過線搜索、Kolmogorov-Nagumo 平均、和簡化類型的方法來進行計算，並針對這些方法的理論背景及計算複雜度進行分析，此外，整合奇異值分解與先前隨機奇異值分解的相似與相異處也會進行說明與分析。數值實驗結果顯示，在所提供的例子中，整合奇異值分解相對於同樣數量的隨機奇異值分解，使用線搜索方法時的疊代次數較少。另外，使用簡化類型的方法，來當作線搜索方法的初始值，可以減少收斂所需的疊代次數。

關鍵詞： 數值線性代數、奇異值分解、隨機演算法、數值優化、維度降低

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Abstract

Dimension reduction and feature extraction are the important techniques in the big-data era to reduce the dimension of data and the computational cost for further data analysis. Low-rank singular value decomposition (low-rank SVD) is the key part of these techniques. In order to compute low-rank SVD faster, some researchers propose to use randomized subspace sketching algorithm to get an approximation result (rSVD). In this research, we propose an idea for integrating the results from randomized algorithm to get a more accurate approximation, which is called integrated singular value decomposition (iSVD). We analyze iSVD and the integration methods by theoretical analysis and numerical experiment. The integration scheme is a constraint optimization problem with unique local maximizer up to orthogonal transformation. Line search type method, Kolmogorov-Nagumo type average method and reduction type method are introduced and analyzed for their theoretical background and computational complexity. The similarity and difference between iSVD and rSVD with same sketching number are also explained and analyzed. The numerical experiment shows that the line search method in iSVD converges faster than the one in rSVD for our test examples. Also, using the integrated subspace from reduction as the initial value of line search method can reduce the iteration number to converge.

Keywords: Numerical Linear Algebra, Singular Value Decomposition, Randomized Algorithm, Numerical Optimization, Dimension Reduction

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Contents

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

List of Figures

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

List of Tables

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Chapter 1

Introduction

Dimension reduction and feature extraction are important issues in data analysis, especially for large scale data, to reduce the size or condense the information of analyzed data. The goal is to reduce the time for further analysis or find out the key features in the data. Singular value decomposition (SVD) is one of the technique to realize dimension reduction. An SVD of an $m \times n$ matrix $\boldsymbol{A}$ takes the form $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U}$ is an $m \times m$ orthogonal matrix, $\boldsymbol{V}$ is an $n \times n$ orthogonal matrix, and $\boldsymbol{\Sigma}$ is an $m \times n$ diagonal matrix with decreasing diagonal entries. In this representation, $\boldsymbol{U}$, $\boldsymbol{V}$ are the left and right singular vectors of $\boldsymbol{A}$ respectively. The diagonal entries of $\boldsymbol{\Sigma}$ are the singular vector of $\boldsymbol{A}$. A rank-k approximation of $\boldsymbol{A}$ via SVD is given as

$$\boldsymbol{A} = \boldsymbol{U}_k\boldsymbol{\Sigma}_k\boldsymbol{V}_k^\top$$

where $\boldsymbol{U}_k$, $\boldsymbol{V}_k$ are leading k singular vectors and $\boldsymbol{\Sigma}_k$ contains leading k singular values. This low-rank approximation is called rank-k SVD of $\boldsymbol{A}$. Rank-k SVD is the best rank-k approximation of $\boldsymbol{A}$ in the sense that it has the smallest 2-norm or Frobenius norm error. Therefore, it is a good choice for dimension reduction in many cases.

Many algorithms are aimed at computing the SVD or rank-k SVD. However, these algorithms usually take at least $O(m^2n + mnk)$ for computing rank-k SVD of a real $m \times n$ ($m \leq n$) matrix. Consequently, these algorithms cost lots of time to obtain the output result. Some research [7, 10] proposed a method to randomly sketch the matrix into a smaller

doi:10.6342/NTU201702973

subspace, and calculate approximate rank-k SVD in that space. This method is called randomized singular value decomposition (rSVD) in this thesis. The accuracy and precision of rSVD depend on the quality of random projection used to generate the subspace. How to increase the quality of random sketching and the accuracy of the approximation is an important issue in rSVD. In [4], integrated singular value decomposition (iSVD) is proposed to enhance the projected subspace and obtain a better result. The main idea of iSVD is using integration method to condense the information from multiple randomly sketching and output a better-sketched subspace.

In this thesis, some properties of iSVD and integration method will be studied. The concept of iSVD is introduced in Chapter 2. The key step of iSVD is the integration, which is a constraint optimization problem. In the Chapter 3, we show the integrated subspace defined previously is the only local maximizer of the constraint optimization problem. The optimal solution and informal explanation of asymptotic behavior are introduced in the same chapter. These properties give the reason for using gradient type methods to solve this optimization problem. These methods are introduced and analyzed in Chapter 4. The similarity and difference between rSVD and iSVD with same sketching number are shown in the Chapter 5. The numerical experiment is shown in Chapter 6. Finally, the discussion and conclusion are given in Chapter 7.

The notations used in this thesis are as follows. The normal letters, such as a, α, denote the scalar. The bold lower case letters, such as $\boldsymbol{a}, \boldsymbol{\alpha}$, denote the vector. The bold upper case letters, such as $\boldsymbol{C}, \boldsymbol{\Gamma}$, denote the matrix. Table 1.1 shows some frequently appeared notations in this thesis.

doi:10.6342/NTU201702973

$\boldsymbol{A}$	The matrix desired solving low-rank SVD.
m, n	The number of rows and columns of $\boldsymbol{A}$ respectively. We assume $m \leq n$.
k	The given desired rank for low-rank approximation.
p	The number of oversampling in rSVD and iSVD.
ℓ	The total number of sampling in rSVD and iSVD for a single sketched subspaces. $\ell = k + p$. $\ell \ll m$.
N	The total number of sketched subspaces in iSVD.
$\otimes$	The Kronecker product.
$\boldsymbol{H}_{a,b}$	The $a \times b$ commutation matrix.[9] For any $a \times b$ matrix $\boldsymbol{M}$, $\text{vec}(\boldsymbol{M}^\top) = \boldsymbol{H}_{a,b}\,\text{vec}(\boldsymbol{M})$.
$\boldsymbol{Q}_c$	The current iterator in the iterative method.
$\boldsymbol{Q}_+$	The iterator of next step in the iterative method.

Table 1.1: Notation in this thesis.

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Chapter 2

Overview of Integrated Singular Value Decomposition

Before introducing iSVD, we shall take a quick view of rSVD. The basic algorithm of rSVD from [7, 10] is stated as Algorithm 1. In this algorithm, SVD is only applied to an $m \times \ell$ matrix and an $\ell \times n$ matrix, which is much cheaper than applied on an $m \times n$ matrix if ℓ is small. The two main phases of rSVD are described in next two paragraphs.

Algorithm 1 Randomized SVD (rSVD)

Require: $\boldsymbol{A}$ (real $m \times n$ matrix), k (number of desired rank for low-rank approximation), p (number of oversampling), $\ell = k + p$ (number of the sketched column),
Ensure: Approximate rank-k SVD of $\boldsymbol{A} \approx \widehat{\boldsymbol{U}}_k \widehat{\boldsymbol{\Sigma}}_k \widehat{\boldsymbol{V}}_k^\top$

1: Generate a random matrix $\boldsymbol{\Omega}$
2: Assign $\boldsymbol{Y} \leftarrow \boldsymbol{A}\boldsymbol{\Omega}$
3: Compute $\boldsymbol{Q}$ whose columns are an orthonormal basis of $\boldsymbol{Y}$
4: Compute the SVD of $\boldsymbol{Q}^\top \boldsymbol{A} = \widehat{\boldsymbol{W}}_\ell \, \widehat{\boldsymbol{\Sigma}}_\ell \, \widehat{\boldsymbol{V}}_\ell^\top$
5: Assign $\widehat{\boldsymbol{U}}_\ell \leftarrow \boldsymbol{Q}\widehat{\boldsymbol{W}}_\ell$
6: Extract the largest k singular pairs from $\widehat{\boldsymbol{U}}_\ell$, $\widehat{\boldsymbol{\Sigma}}_\ell$, $\widehat{\boldsymbol{V}}_\ell$ to obtain $\widehat{\boldsymbol{U}}_k$, $\widehat{\boldsymbol{\Sigma}}_k$, $\widehat{\boldsymbol{V}}_k$

The first phase is randomly sketching a subspace of $\boldsymbol{A}$. More precisely, compute the matrix $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{\Omega}$ and find out the orthogonal basis of the range of $\boldsymbol{Y}$ as the approximate subspace $\boldsymbol{Q}$. In this case, $\boldsymbol{\Omega}$ is a randomly generated matrix. Both [7, 10] propose a commonly used $\boldsymbol{\Omega}$ as Gaussian projection, which means the entries of $\boldsymbol{\Omega}$ are independent identical standard normal distribution. This choice gives $\boldsymbol{Y}$ some structure of the column space from $\boldsymbol{A}$ and controls the error $\left\|\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{A} - \boldsymbol{A}\right\|_2 < \epsilon$ in high probability. Note that

doi:10.6342/NTU201702973

$\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{A}$ is the approximate rank-ℓ SVD of $\boldsymbol{A}$ with the column space spanned by $\boldsymbol{Y}$.

The second phase is constructing an approximate rank-ℓ SVD $\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{A}$. Only the SVD of $\boldsymbol{Q}^\top \boldsymbol{A}$ is needed to construct the SVD of the rank-ℓ matrix $\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{A}$. Once the SVD of $\boldsymbol{Q}^\top \boldsymbol{A} = \widehat{\boldsymbol{W}}_\ell \widehat{\boldsymbol{\Sigma}}_\ell \widehat{\boldsymbol{V}}_\ell^\top$ is obtained, the SVD of $\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{A}$ can be computed as $\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{A} = (\boldsymbol{Q}\widehat{\boldsymbol{W}}_\ell)\widehat{\boldsymbol{\Sigma}}_\ell \widehat{\boldsymbol{V}}_\ell^\top$. Note that the column space of this approximation is already determined in the first phase. The purpose of this phase is revealing the singular values and singular vectors in correct arrangement, and then the leading k singular values and singular vectors can be extracted.

The technique of oversampling is proposed here. Suppose the desired rank is k, the number of sketches in the first phase can be chosen as $\ell = k + p$ for some positive integer p. The approximate rank-k SVD is obtained from rank-ℓ SVD by extracting the first k singular vectors and singular values.

Based on rSVD, iSVD uses multiple random sketched subspaces in the first phase of rSVD to gain more accurate result. The algorithm is stated as Algorithm 2. Three phases are included in this algorithm. The first phase is similar to the first phase of SVD. Instead of choosing only one random sketch in rSVD, iSVD choose multiple random sketches. The second phase is integrating the subspaces obtained in the first phase and get an integrated subspace $\overline{\boldsymbol{Q}}$. The third phase is same as the second phase of rSVD. They both construct the approximate rank-ℓ SVD.

Algorithm 2 Integrated SVD with multiple sketches (iSVD).

Require: $\boldsymbol{A}$ (real $m \times n$ matrix), k (desired rank of approximate SVD), p (oversampling parameter), $\ell = k + p$ (dimension of the sketched column space), q (exponent of the power method), N (number of random sketches)

Ensure: Approximate rank-k SVD of $\boldsymbol{A} \approx \widehat{\boldsymbol{U}}_k \widehat{\boldsymbol{\Sigma}}_k \widehat{\boldsymbol{V}}_k^\top$

1: Generate $n \times \ell$ random matrices $\boldsymbol{\Omega}_{[i]}$ for $i = 1, \ldots, N$
2: Assign $\boldsymbol{Y}_{[i]} \leftarrow \boldsymbol{A}\boldsymbol{\Omega}_{[i]}$ for $i = 1, ..., N$
3: Compute $\boldsymbol{Q}_{[i]}$ whose columns are an orthonormal basis of $\boldsymbol{Y}_{[i]}$
4: Integrate $\overline{\boldsymbol{Q}} \leftarrow \{\boldsymbol{Q}_{[i]}\}_{i=1}^N$
5: Compute the SVD of $\overline{\boldsymbol{Q}}^\top \boldsymbol{A} = \widehat{\boldsymbol{W}}_\ell \, \widehat{\boldsymbol{\Sigma}}_\ell \, \widehat{\boldsymbol{V}}_\ell^\top$
6: Assign $\widehat{\boldsymbol{U}}_\ell \leftarrow \overline{\boldsymbol{Q}}\widehat{\boldsymbol{W}}_\ell$
7: Extract the largest k singular pairs from $\widehat{\boldsymbol{U}}_\ell$, $\widehat{\boldsymbol{\Sigma}}_\ell$, $\widehat{\boldsymbol{V}}_\ell$ to obtain $\widehat{\boldsymbol{U}}_k$, $\widehat{\boldsymbol{\Sigma}}_k$, $\widehat{\boldsymbol{V}}_k$

The key part of iSVD is how to define the integrated subspace in the second phase. The most intuitive idea is taking the arithmetic average of $\boldsymbol{Q}_{[i]}$ as $\overline{\boldsymbol{Q}}$, but the following state-

doi:10.6342/NTU201702973

ments show this is not a reasonable definition of the integrated subspace. The integrated subspace $\overline{\boldsymbol{Q}}$ should represent the subspace integrated by $\boldsymbol{Q}_{[i]}$. Each $\boldsymbol{Q}_{[i]}$ is an orthogonal matrix. Hence $\overline{\boldsymbol{Q}}$ should also be an orthogonal matrix. However, the arithmetic average of $\boldsymbol{Q}_{[i]}$ is not an orthogonal matrix. Therefore, the integrated subspaces should be defined by another form instead of the arithmetic average.

The integrated subspace in the second phase is defined by the following optimization problem.

$$\begin{cases} \overline{\boldsymbol{Q}} := \underset{\boldsymbol{Q}}{\operatorname{argmin}} \dfrac{1}{N} \displaystyle\sum_{i=1}^{N} \left\| \boldsymbol{Q}_{[i]} \boldsymbol{Q}_{[i]}^{\top} - \boldsymbol{Q}\boldsymbol{Q}^{\top} \right\|_F^2 \\ \text{subject to } \boldsymbol{Q}^{\top}\boldsymbol{Q} = \boldsymbol{I}_{\ell} \end{cases} \tag{2.1}$$

The main idea of this definition is to define the integrated subspace that has the minimum summation of distances between each $\boldsymbol{Q}_{[i]}$ and $\overline{\boldsymbol{Q}}$, which is similar to the property of the arithmetic average in Euclidean space. Instead of the Euclidean distance between $\boldsymbol{Q}$ and each $\boldsymbol{Q}_{[i]}$, we use the distance between $\boldsymbol{Q}\boldsymbol{Q}^{\top}$ and $\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}$ to preserve the invariant of right orthogonal transformation. Suppose $\boldsymbol{Q}_{[1]} = \boldsymbol{Q}_{[2]}\boldsymbol{R}$ for some orthogonal matrix $\boldsymbol{R}$. $\boldsymbol{Q}_{[1]}$ and $\boldsymbol{Q}_{[2]}$ represent the same subspaces, so the measurement of error between $\boldsymbol{Q}_{[1]}$, $\boldsymbol{Q}_{[i]}$, and $\boldsymbol{Q}_{[2]}$, $\boldsymbol{Q}_{[i]}$ should be same. By using the relation

$$\boldsymbol{Q}_{[1]}\boldsymbol{Q}_{[1]}^{\top} - \boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top} = \boldsymbol{Q}_{[1]}\boldsymbol{R}\boldsymbol{R}^{\top}\boldsymbol{Q}_{[1]}^{\top} - \boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top} = \boldsymbol{Q}_{[2]}\boldsymbol{Q}_{[2]}^{\top} - \boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}$$

the error measured in (2.1) gives the same value, which means it preserved the invariant of right orthogonal transformation. However, it still needs some theoretical analysis to make sure whether this objective function is suitable for the integration step. Chapter 3 provides an informal explanation that the integrated subspace defined by 2.1 can capture the true singular vectors of the desired matrix $\boldsymbol{A}$ when N tends to be large.

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Chapter 3

Properties of Integrated Subspace

In this chapter, some properties of the integrated subspace defined in (2.1) are analyzed. First, an equivalent maximization problem (3.1) is introduced. Next, we show the solution of the optimization problem in (3.1) is the leading singular vectors of the arithmetic average of $\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top$. Then we provide an informal explanation for the asymptotic behavior of integrated subspace when N is large. Finally, we prove that the local maximizer of the optimization problem in (3.1) is unique up to a right orthogonal transform.

At the beginning, some equivalent problems should be introduced for the simplicity of analysis. The following theorem demonstrates an equivalent optimization problems of (2.1).

Theorem 3.0.1. *The constraint optimization problem* (2.1) *is equivalent to the problem*

$$\begin{cases} \underset{\boldsymbol{Q}}{\text{argmax}}\, \dfrac{1}{2}\,\text{tr}(\boldsymbol{Q}^\top \overline{\boldsymbol{P}}\boldsymbol{Q}) \\ \boldsymbol{Q}^\top\boldsymbol{Q} = \boldsymbol{I}_\ell \end{cases} \tag{3.1}$$

where $\overline{\boldsymbol{P}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top$ *is the arithmetic average of* $\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top$ *over all* i.

Proof. From the relation between trace and Frobenius norm $\|\boldsymbol{M}\|_F^2 = \text{tr}(\boldsymbol{M}^\top\boldsymbol{M})$, the properties of trace $\text{tr}(\boldsymbol{M}_1\boldsymbol{M}_2) = \text{tr}(\boldsymbol{M}_2\boldsymbol{M}_1)$, and the orthogonality of $\boldsymbol{Q}$ and $\boldsymbol{Q}_{[i]}$, the

doi:10.6342/NTU201702973

Frobenius norm in the objective function of (2.1) can be rewritten as

$$\begin{aligned}\left\|\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}-\boldsymbol{Q}\boldsymbol{Q}^{\top}\right\|_F^2 &= \operatorname{tr}((\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}-\boldsymbol{Q}\boldsymbol{Q}^{\top})^{\top}(\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}-\boldsymbol{Q}\boldsymbol{Q}^{\top}))\\ &= \operatorname{tr}(\boldsymbol{Q}\boldsymbol{Q}^{\top})-2\operatorname{tr}(\boldsymbol{Q}\boldsymbol{Q}^{\top}\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top})+\operatorname{tr}(\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top})\\ &= \operatorname{tr}(\boldsymbol{Q}^{\top}\boldsymbol{Q})-2\operatorname{tr}(\boldsymbol{Q}^{\top}\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}\boldsymbol{Q})+\operatorname{tr}(\boldsymbol{Q}_{[i]}^{\top}\boldsymbol{Q}_{[i]})\\ &= 2\ell-2\operatorname{tr}(\boldsymbol{Q}^{\top}\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}\boldsymbol{Q}).\end{aligned}$$

This equation leads to another representation of the objective function

$$\frac{1}{N}\sum_{i=1}^{N}\left\|\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}-\boldsymbol{Q}\boldsymbol{Q}^{\top}\right\|_F^2 = 2\frac{\ell}{N}-2\frac{1}{N}\sum_{i=1}^{N}\operatorname{tr}(\boldsymbol{Q}^{\top}\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^{\top}\boldsymbol{Q}) = 2\frac{\ell}{N}-2\operatorname{tr}(\boldsymbol{Q}^{\top}\overline{\boldsymbol{P}}\boldsymbol{Q}).$$

Since the constant does not affect the problem and the negative scalar changes minimize problem to maximize problem, (2.1) is equivalent to the problem

$$\begin{cases}\underset{\boldsymbol{Q}}{\operatorname{argmax}}\,\dfrac{1}{2}\operatorname{tr}(\boldsymbol{Q}^{\top}\overline{\boldsymbol{P}}\boldsymbol{Q})\\ \boldsymbol{Q}^{\top}\boldsymbol{Q}=\boldsymbol{I}_{\ell}\end{cases}$$

which is the problem (3.1).

□

The optimization problem in the form (3.1) is more simple than the original form (2.1) for computing the derivative and further analyzing. So the simplified form (3.1) will be used in the following analysis.

3.1 Solution of the Optimization Problem

The solution of the maximize problem (3.1) is the integrated subspace we defined in (2.1). Theorem 3.1.1 shows that the optimizer of this problem is consisted by the leading ℓ eigenvectors of $\overline{\boldsymbol{P}}$.

Theorem 3.1.1 (Maximal Value of Objective Function)**.** *Let* $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_\ell > \lambda_{\ell+1} \geq \ldots \geq \lambda_m$ *be the eigenvalues of* $\overline{\boldsymbol{P}}$ *and let* $\boldsymbol{Q}_*$ *be the corresponding leading* ℓ

doi:10.6342/NTU201702973

eigenvectors. Then the objective function in (3.1) *has the upper bound*

$$\frac{1}{2}\operatorname{tr}(\boldsymbol{Q}^\top \overline{\boldsymbol{P}}\boldsymbol{Q}) \le \frac{1}{2}\sum_{i=1}^{\ell}\lambda_i$$

and the equality holds when $\boldsymbol{Q} = \boldsymbol{Q}_*\boldsymbol{R}_*$ *for some* $\ell \times \ell$ *orthogonal matrix* $\boldsymbol{R}_*$.

Proof. The eigenvalue decomposition of $\overline{\boldsymbol{P}}$ can be written as

$$\overline{\boldsymbol{P}} = \boldsymbol{Q}_*\boldsymbol{S}_*\boldsymbol{Q}_*^\top + \boldsymbol{Q}_\perp\boldsymbol{S}_\perp\boldsymbol{Q}_\perp^\top$$

where $\boldsymbol{Q}_\perp$ denote orthonormal basis of the space perpendicular to $\boldsymbol{Q}$, $\boldsymbol{S}_* = \operatorname{diag}(\lambda_1, \ldots, \lambda_\ell)$ and $\boldsymbol{S}_\perp = \operatorname{diag}(\lambda_{\ell+1}, \ldots, \lambda_m)$. Since $\boldsymbol{Q}_*$ and $\boldsymbol{Q}_\perp$ spans the whole $\mathbf{R}^m$, any orthogonal matrix $\boldsymbol{Q}$ can be represented as $\boldsymbol{Q} = \boldsymbol{Q}_*\boldsymbol{B} + \boldsymbol{Q}_\perp\boldsymbol{C}$ with $\boldsymbol{B}^\top\boldsymbol{B} + \boldsymbol{C}^\top\boldsymbol{C} = \boldsymbol{I}_\ell$. The objective function becomes

$$\begin{aligned}
&\frac{1}{2}\operatorname{tr}(\boldsymbol{Q}^\top\overline{\boldsymbol{P}}\boldsymbol{Q}) \\
&= \frac{1}{2}\operatorname{tr}((\boldsymbol{Q}_*\boldsymbol{B} + \boldsymbol{Q}_\perp\boldsymbol{C})^\top(\boldsymbol{Q}_*\boldsymbol{S}_*\boldsymbol{Q}_*^\top + \boldsymbol{Q}_\perp\boldsymbol{S}_\perp\boldsymbol{Q}_\perp^\top)(\boldsymbol{Q}_*\boldsymbol{B} + \boldsymbol{Q}_\perp\boldsymbol{C})) \\
&= \frac{1}{2}\operatorname{tr}(\boldsymbol{B}^\top\boldsymbol{S}_*\boldsymbol{B}) + \operatorname{tr}(\boldsymbol{C}^\top\boldsymbol{S}_\perp\boldsymbol{C})
\end{aligned}$$

by the relation $\boldsymbol{Q}^\top\boldsymbol{Q}_\perp = 0$

Let $\boldsymbol{B} = \boldsymbol{R}\boldsymbol{S}\boldsymbol{T}^\top$ be the SVD of $\boldsymbol{B}$. By multiplying $\boldsymbol{R}\boldsymbol{T}^\top$ from left and $\boldsymbol{T}\boldsymbol{R}^\top$ from right to the both sides of the condition $\boldsymbol{B}^\top\boldsymbol{B} + \boldsymbol{C}^\top\boldsymbol{C} = \boldsymbol{I}_\ell$, the new condition is given as $\boldsymbol{R}\boldsymbol{S}^2\boldsymbol{R}^\top + \boldsymbol{R}\boldsymbol{T}^\top\boldsymbol{C}^\top\boldsymbol{C}\boldsymbol{T}\boldsymbol{R}^\top = \boldsymbol{R}\boldsymbol{T}^\top\boldsymbol{T}\boldsymbol{R}^\top = \boldsymbol{I}_\ell$. By using this new condition and the inequality

$$\operatorname{tr}((\boldsymbol{C}\boldsymbol{T}\boldsymbol{R})^\top\boldsymbol{S}_\perp\boldsymbol{C}\boldsymbol{T}\boldsymbol{R}^\top) = \sum_{i=1}^{m-\ell}\sum_{j=1}^{\ell}\lambda_{\ell+i}c_{ij}^2 \le \sum_{i=1}^{m-\ell}\sum_{j=1}^{\ell}\lambda_j c_{ij}^2 = \operatorname{tr}(\boldsymbol{S}_*\boldsymbol{R}\boldsymbol{T}^\top\boldsymbol{C}^\top\boldsymbol{C}\boldsymbol{T}\boldsymbol{R}^\top)$$

doi:10.6342/NTU201702973

where $\boldsymbol{CTR}^\top = [c_{ij}]$, the upper bound of the objective function can be given as

$$\begin{aligned}
\frac{1}{2}\operatorname{tr}(\boldsymbol{Q}^\top \overline{\boldsymbol{P}}\boldsymbol{Q}) &= \frac{1}{2}\operatorname{tr}(\boldsymbol{B}^\top \boldsymbol{S}_*\boldsymbol{B}) + \operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{S}_\perp \boldsymbol{C}) \\
&= \frac{1}{2}\left(\operatorname{tr}(\boldsymbol{TSR}^\top \boldsymbol{S}_*\boldsymbol{RST}^\top) + \operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{S}_\perp \boldsymbol{CTR}^\top \boldsymbol{RT}^\top)\right) \\
&= \frac{1}{2}\left(\operatorname{tr}(\boldsymbol{S}_*\boldsymbol{RS}^2\boldsymbol{R}^\top) + \operatorname{tr}(\boldsymbol{RT}^\top \boldsymbol{C}^\top \boldsymbol{S}_\perp \boldsymbol{CTR}^\top)\right) \\
&\leq \frac{1}{2}\operatorname{tr}(\boldsymbol{S}_*(\boldsymbol{RS}^2\boldsymbol{R}^\top + \boldsymbol{RT}^\top \boldsymbol{C}^\top \boldsymbol{CTR}^\top)) = \frac{1}{2}\operatorname{tr}(\boldsymbol{S}_*) = \frac{1}{2}\sum_{i=1}^{\ell}\lambda_i.
\end{aligned}$$

The quality holds when $\sum_{i=1}^{m-\ell}\sum_{j=1}^{\ell}\lambda_{\ell+i}c_{ij}^2 = \sum_{i=1}^{m-\ell}\sum_{j=1}^{\ell}\lambda_j c_{ij}^2$. This equation means $\boldsymbol{CTR}^\top = [c_{ij}] = 0$, and hence $\boldsymbol{C} = 0$. Therefore, $\boldsymbol{Q} = \boldsymbol{Q}_*\boldsymbol{R}_*$ with $\boldsymbol{B}^\top\boldsymbol{B} = \boldsymbol{I}_\ell$ and $\boldsymbol{R}_* = \boldsymbol{B}$. □

Theorem 3.1.1 shows that the integrated subspace defined in (2.1) or (3.1) is actually formed by the leading eigenvectors of the arithmetic average of $\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top$. The leading eigenvectors of $\overline{\boldsymbol{P}}$ is same as the leading ℓ singular vectors of the matrix $\left[\boldsymbol{Q}_{[1]}|\boldsymbol{Q}_{[2]}|\cdots|\boldsymbol{Q}_{[N]}\right]$. This form is used for explaining the similarity of rSVD and iSVD with same sketching number in Chapter 5.

3.2 Asymptotic Behavior of the Integrated Subspace

Now we give an informal explanation about the reason why the the integrated subspace defined in (2.1) or (3.1) can work in the iSVD algorithm. Please refer to [4] for more detailed statistical analysis.

To explain the reason, the following theorem in [4] should be introduced first.

Theorem 3.2.1. *Let the SVD of $\boldsymbol{A}$ be $\boldsymbol{A} = \boldsymbol{U\Sigma V}^\top$ with distinct decreasing singular values. Let $\boldsymbol{Q}$ denote an orthogonal subspace spanned by $\boldsymbol{Y} = \boldsymbol{A\Omega}$, where $\boldsymbol{\Omega}$ is randomly generated by i.i.d. standard normal entries. Then*

$$E\left[\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top\right] = \boldsymbol{U\Lambda U}^\top$$

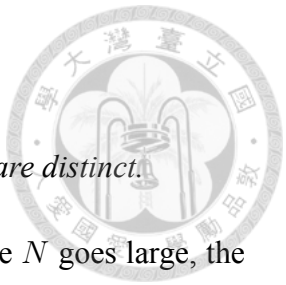
where $\boldsymbol{\Lambda}$ satisfies

doi:10.6342/NTU201702973

1. $\mathbf{\Lambda}$ *is diagonal matrix*

2. *all diagonal entries are in the interval* $(0, 1)$

3. *the diagonal entries are decreasing if the singular values of* $\boldsymbol{A}$ *are distinct.*



By the law of large number and Theorem 3.2.1, as the sample size N goes large, the arithmetic average $\overline{\boldsymbol{P}}$ tends to a matrix with the same singular vectors of $\boldsymbol{A}$ in the correct arrangement. Also, Theorem 3.1.1 shows that the optimizer defined in (3.1) is the leading singular vectors of $\overline{\boldsymbol{P}}$. Hence in the ideal case (N tends to infinity), iSVD can capture the leading singular vectors of $\boldsymbol{A}$ as $\overline{\boldsymbol{Q}}$ and leads to an ideal result for the low-rank approximation.

3.3 Uniqueness of Local Maximizer

Recall that we focus on the problem in the form (3.1)

$$(+) \begin{cases} \max\limits_{\boldsymbol{Q}} \dfrac{1}{2} \operatorname{tr}(\boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q}) \\ \boldsymbol{Q}^\top \boldsymbol{Q} = \boldsymbol{I}_\ell \end{cases}.$$

Theorem 3.1.1 shows the optimal solution of the problem (+) is formed by the leading ℓ eigenvectors of the matrix $\overline{\boldsymbol{P}}$. However, this result dose not provide whether there exists another local maximum of this problem. The goal of the following derivation is Theorem 3.3.5, which shows that the only local maximum of the problem (+) is the orthogonal matrices that formed by the leading ℓ eigenvectors of $\overline{\boldsymbol{P}}$ (up to right orthogonal transformation). The main idea of the following proof is checking the first and second order necessary condition for the nonlinear equality constraint optimization problem. More information for the first and second order condition of optimization with equality constraint can be obtained in the book for introducing optimization problem. (For example, [6].)

We begin with checking the first order necessary condition of the problem (+).

Lemma 3.3.1 (First Order Necessary Condition)**.** *Suppose* $\boldsymbol{Q}$ *is a local maximizer of the problem (+) in the feasible set, i.e., the set collects all the* $m \times \ell$ *orthogonal matrix. Then*

doi:10.6342/NTU201702973

$\boldsymbol{Q}$ satisfies the equation

$$(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\overline{\boldsymbol{P}}\boldsymbol{Q} = \boldsymbol{0}. \tag{3.2}$$

and the corresponding Lagrange multiplier $\boldsymbol{\Lambda} \in \mathbf{R}^{\ell\times\ell}$ satisfies

$$\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^\top = \boldsymbol{Q}^\top\overline{\boldsymbol{P}}\boldsymbol{Q}. \tag{3.3}$$

Proof. This equation is obtained from the first order condition for optimal solution of an equality constraint, which states that if x^* is a local maximizer of an equality constraint optimization problem, then the following equations hold

$$\begin{cases} \nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \boldsymbol{0} \\ \nabla_{\boldsymbol{\lambda}}\mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) = \boldsymbol{0} \end{cases}$$

where $\mathcal{L}$ is the Lagrangian of the equality constraint optimization problem and $\boldsymbol{\lambda}^*$ is the corresponding Lagrange multiplier.

Notice that any $m \times n$ matrix $\boldsymbol{M}$ can be seen as an mn vector by vectorizing $\boldsymbol{M}$ as $\mathrm{vec}(\boldsymbol{M})$. Also, one of the relation between $\mathrm{vec}(\bullet)$ and Kronecker product $\otimes$ is $\mathrm{vec}(\boldsymbol{A}\boldsymbol{X}\boldsymbol{B}) = (\boldsymbol{B}^\top \otimes \boldsymbol{A})\,\mathrm{vec}(\boldsymbol{X})$. By using these technique, the Lagrangian of (+) is given as

$$\begin{aligned} \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) &= \frac{1}{2}\,\mathrm{tr}(\boldsymbol{Q}^\top\overline{\boldsymbol{P}}\boldsymbol{Q}) - \mathrm{tr}(\boldsymbol{\Lambda}^\top(\boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{I}_\ell)) \\ &= \frac{1}{2}\,\mathrm{vec}(\boldsymbol{Q})^\top\,\mathrm{vec}(\overline{\boldsymbol{P}}\boldsymbol{Q}) - \mathrm{vec}(\boldsymbol{\Lambda})^\top\,\mathrm{vec}(\boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{I}_\ell) \\ &= \frac{1}{2}\,\mathrm{vec}(\boldsymbol{Q})^\top(\boldsymbol{I}_\ell \otimes \overline{\boldsymbol{P}})\,\mathrm{vec}(\boldsymbol{Q}) - \mathrm{vec}(\boldsymbol{\Lambda})^\top\,\mathrm{vec}(\boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{I}_\ell) \end{aligned}$$

and the first order conditions can be represented by the derivative of Lagrangian as

$$\begin{cases} \nabla_{\mathrm{vec}(\boldsymbol{Q})}\mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) = (\boldsymbol{I} \otimes \overline{\boldsymbol{P}})\,\mathrm{vec}(\boldsymbol{Q}) - [\boldsymbol{K}_{\ell,m}(\boldsymbol{Q} \otimes \boldsymbol{I}_\ell) + (\boldsymbol{I}_\ell \otimes \boldsymbol{Q})]\,\mathrm{vec}(\boldsymbol{\Lambda}) = \boldsymbol{0} \\ \nabla_{\mathrm{vec}(\boldsymbol{\Lambda})}\mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) = \mathrm{vec}(\boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{I}_\ell) = \boldsymbol{0} \end{cases}$$

where $\boldsymbol{K}_{\ell,m}$ denotes the communication matrix corresponding to $\ell \times m$ matrix. By folding

doi:10.6342/NTU201702973

the vectorized matrix back, the above equations leads to

$$\begin{cases} \overline{\boldsymbol{P}}\boldsymbol{Q} - \boldsymbol{Q}(\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^\top) = \mathbf{0} \\ \boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{I}_\ell = \mathbf{0}. \end{cases}$$

Applying $\boldsymbol{Q}^\top$ to the left of each side in the first equation and Using the second equation derives the relation $\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^\top = \boldsymbol{Q}^\top\overline{\boldsymbol{P}}\boldsymbol{Q}$. By substituting $(\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^\top)$ as $\boldsymbol{Q}^\top\overline{\boldsymbol{P}}\boldsymbol{Q}$ in the first equation, the first order conditions can be rewritten as

$$\begin{cases} (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\overline{\boldsymbol{P}}\boldsymbol{Q} = \mathbf{0} \\ \boldsymbol{Q}^\top\boldsymbol{Q} = \boldsymbol{I}_\ell \end{cases}$$

□

Lemma 3.3 gives the necessary conditions for feasible points and the corresponding Lagrange multiplier. The following lemma improves the result. It provides the explicit solutions that satisfy these conditions.

Lemma 3.3.2. *Let $\boldsymbol{Q}$ be a feasible points of the problem (+). Then $\boldsymbol{Q}$ satisfies the first order condition* (3.2) *if and only if each column of $\boldsymbol{Q}\boldsymbol{R}$ is an eigenvector of $\overline{\boldsymbol{P}}$ for some orthogonal matrix $\boldsymbol{R} \in \mathbf{R}^{\ell\times\ell}$.*

Proof. For the 'if' part, since the columns of $\boldsymbol{Q}\boldsymbol{R}$ are eigenvectors of $\overline{\boldsymbol{P}}$, the connection between $\overline{\boldsymbol{P}}$ and $\boldsymbol{Q}\boldsymbol{R}$ is given as $\overline{\boldsymbol{P}}\boldsymbol{Q}\boldsymbol{R} = \boldsymbol{Q}\boldsymbol{R}\boldsymbol{S}$, where $\boldsymbol{S}$ is a diagonal matrix with the corresponding eigenvalues on its diagonal entries. By directly calculation,

$$(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}\boldsymbol{R} = (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\boldsymbol{Q}\boldsymbol{R}\boldsymbol{S} = \mathbf{0}$$

and hence

$$(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\overline{\boldsymbol{P}}\boldsymbol{Q} = \mathbf{0}$$

by multiplying $\boldsymbol{R}^\top$ to the right of both sides. This equation shows that $\boldsymbol{Q}$ satisfies the first order condition (3.2).

doi:10.6342/NTU201702973

For the 'only if' part, suppose $\boldsymbol{Q}$ satisfies the first order condition $(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\overline{\boldsymbol{P}}\boldsymbol{Q} = \boldsymbol{0}$. Since $(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)$ is the projection matrix onto the orthogonal complement of the range of $\boldsymbol{Q}$, the first order condition means that the component of each columns of $\overline{\boldsymbol{P}}\boldsymbol{Q}$ in the orthogonal complement is $\boldsymbol{0}$, i.e., each column of $\overline{\boldsymbol{P}}\boldsymbol{Q}$ is a vector in the range of $\boldsymbol{Q}$. Therefore, each column of $\overline{\boldsymbol{P}}\boldsymbol{Q}$ can be written as the linear combination of the columns of $\boldsymbol{Q}$ and hence

$$\overline{\boldsymbol{P}}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{B}$$

for some $\boldsymbol{B} \in \mathbf{R}^{\ell\times\ell}$. By applying $\boldsymbol{Q}^\top$ to the left of both sides, the equation becomes $\boldsymbol{B} = \boldsymbol{Q}^\top\overline{\boldsymbol{P}}\boldsymbol{Q}$ which is a symmetric matrix. Therefore, $\boldsymbol{B}$ has eigenvalue decomposition $\boldsymbol{B} = \boldsymbol{R}\boldsymbol{S}\boldsymbol{R}^\top$ for some orthogonal matrix $\boldsymbol{R}$ and diagonal matrix $\boldsymbol{S}$. This decomposition gives the relation

$$\overline{\boldsymbol{P}}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{R}\boldsymbol{S}\boldsymbol{R}^\top$$

and hence

$$\overline{\boldsymbol{P}}(\boldsymbol{Q}\boldsymbol{R}) = (\boldsymbol{Q}\boldsymbol{R})\boldsymbol{S}$$

by multiplying $\boldsymbol{R}$ to the right of both sides. This equation shows each column of $(\boldsymbol{Q}\boldsymbol{R})$ is an eigenvector of $\overline{\boldsymbol{P}}$ since $\boldsymbol{S}$ is diagonal. □

Now we check the second order condition. The preparation for computing second order condition is finding the null space spanned by the gradient of constrains. Let $g(\boldsymbol{Q})$ denote the constraint function $g(\boldsymbol{Q}) = \mathrm{vec}(\boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{I}_\ell) = \boldsymbol{0}$, and $\mathcal{N}_{\boldsymbol{Q}}$ denote the null space of the linear space spanned by the gradients of constrains $\nabla_{\mathrm{vec}(\boldsymbol{Q})}g(\boldsymbol{Q})$ at $\boldsymbol{Q}$. The following lemma gives some explicit representations of the null space $\mathcal{N}_{\boldsymbol{Q}}$.

Lemma 3.3.3. *The null space $\mathcal{N}_{\boldsymbol{Q}}$ is given as*

$$\mathcal{N}_{\boldsymbol{Q}} = \left\{\mathrm{vec}(\boldsymbol{Z}) : \boldsymbol{Z} \in \mathbf{R}^{m\times\ell}, \boldsymbol{Q}^\top\boldsymbol{Z} + \boldsymbol{Z}^\top\boldsymbol{Q} = \boldsymbol{0}\right\}$$

or equivalently,

$$\mathcal{N}_{\boldsymbol{Q}} = \left\{\mathrm{vec}(\boldsymbol{Q}\boldsymbol{B} + \boldsymbol{Q}_\perp\boldsymbol{C}) : \boldsymbol{B} \in \mathbf{R}^{\ell\times\ell}, \boldsymbol{C} \in \mathbf{R}^{(m-\ell)\times\ell}, \boldsymbol{B} = -\boldsymbol{B}^\top\right\}$$

doi:10.6342/NTU201702973

where $\boldsymbol{Q}_\perp$ is a $m \times (m-\ell)$ orthogonal matrix satisfies $\boldsymbol{Q}_\perp^\top \boldsymbol{Q} = 0$, i.e., $\boldsymbol{Q}_\perp$ contains the basis of orthogonal complements of the range of $\boldsymbol{Q}$.

Proof. By direct calculation, the gradient matrix of constraints at $\boldsymbol{Q}$ is given as

$$\nabla_{\mathrm{vec}(\boldsymbol{Q})} g(\boldsymbol{Q}) = \boldsymbol{K}_{\ell,m}(\boldsymbol{Q} \otimes \boldsymbol{I}_\ell) + (\boldsymbol{I}_\ell \otimes \boldsymbol{Q}).$$

According to the definition of $\mathcal{N}_{\boldsymbol{Q}}$, every element $\mathrm{vec}(\boldsymbol{Z})$ in $\mathcal{N}_{\boldsymbol{Q}}$ satisfies the condition

$$\left[\boldsymbol{K}_{\ell,m}(\boldsymbol{Q} \otimes \boldsymbol{I}_\ell) + (\boldsymbol{I}_\ell \otimes \boldsymbol{Q})\right]^\top \mathrm{vec}(\boldsymbol{Z}) = \boldsymbol{0}$$

which leads to

$$\boldsymbol{Q}^\top \boldsymbol{Z} + \boldsymbol{Z}^\top \boldsymbol{Q} = \boldsymbol{0}.$$

This computation shows that $\mathcal{N}_{\boldsymbol{Q}} \subset \left\{\mathrm{vec}(\boldsymbol{Z}) : \boldsymbol{Z} \in \mathbf{R}^{m\times\ell}, \boldsymbol{Q}^\top \boldsymbol{Z} + \boldsymbol{Z}^\top \boldsymbol{Q} = \boldsymbol{0}\right\}$. The equality can be proved by computing the dimension of each space.

Since the columns of $\boldsymbol{Q}$ and $\boldsymbol{Q}_\perp$ can span $\mathbf{R}^m$, any matrix $\boldsymbol{Z} \in \mathbf{R}^{m\times\ell}$ can be represented as $\boldsymbol{Z} = \boldsymbol{Q}\boldsymbol{B} + \boldsymbol{Q}_\perp \boldsymbol{C}$, where $\boldsymbol{B} \in \mathbf{R}^{\ell\times\ell}$ and $\boldsymbol{C} \in \mathbf{R}^{(m-\ell)\times\ell}$. Another equivalent condition for null space is obtained by plugging this representation into the condition

$$\boldsymbol{Q}^\top(\boldsymbol{Q}\boldsymbol{B} + \boldsymbol{Q}_\perp \boldsymbol{C}) + (\boldsymbol{Q}\boldsymbol{B} + \boldsymbol{Q}_\perp \boldsymbol{C})^\top \boldsymbol{Q} = \boldsymbol{B} + \boldsymbol{B}^\top = \boldsymbol{0}.$$

Hence

$$\mathcal{N}_{\boldsymbol{Q}} = \left\{\mathrm{vec}(\boldsymbol{Q}\boldsymbol{B} + \boldsymbol{Q}_\perp \boldsymbol{C}) : \boldsymbol{B} \in \mathbf{R}^{\ell\times\ell}, \boldsymbol{C} \in \mathbf{R}^{(m-\ell)\times\ell}, \boldsymbol{B} = -\boldsymbol{B}^\top\right\}.$$

□

The second order necessary condition can be written out by the explicit form of the null space.

Lemma 3.3.4 (Second Order Necessary Condition)**.** *Suppose $\boldsymbol{Q}$ is a local maximizer of*

doi:10.6342/NTU201702973

(+). Then $\boldsymbol{Q}$ satisfies the inequality

$$\operatorname{tr}(\boldsymbol{Z}^\top \overline{\boldsymbol{P}} \boldsymbol{Z} - \boldsymbol{Z}^\top \boldsymbol{Z} \boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q}) \leq 0.$$

for all $\operatorname{vec}(\boldsymbol{Z}) \in \mathcal{N}_{\boldsymbol{Q}}$.

Proof. The second order necessary condition is

$$\operatorname{vec}(\boldsymbol{Z})^\top \nabla^2_{\boldsymbol{QQ}} \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) \operatorname{vec}(\boldsymbol{Z}) \leq 0$$

for all $\operatorname{vec}(\boldsymbol{Z}) \in \mathcal{N}_{\boldsymbol{Q}}$. By the relation of Lagrange multiplier at the feasible point (3.3) as $\boldsymbol{\Lambda} + \boldsymbol{\Lambda}^\top = \boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q}$, the gradient of $\mathcal{L}$ by $\boldsymbol{Q}$ can be written as

$$\begin{aligned}
\nabla_{\boldsymbol{Q}} \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) &= (\boldsymbol{I} \otimes \overline{\boldsymbol{P}}) \operatorname{vec}(\boldsymbol{Q}) - [\boldsymbol{K}_{\ell,m}(\boldsymbol{Q} \otimes \boldsymbol{I}_\ell) + (\boldsymbol{I}_\ell \otimes \boldsymbol{Q})] \operatorname{vec}(\boldsymbol{\Lambda}) \\
&= (\boldsymbol{I} \otimes \overline{\boldsymbol{P}}) \operatorname{vec}(\boldsymbol{Q}) - \operatorname{vec}(\boldsymbol{Q}\boldsymbol{\Lambda}^\top + \boldsymbol{Q}\boldsymbol{\Lambda}) \\
&= (\boldsymbol{I} \otimes \overline{\boldsymbol{P}}) \operatorname{vec}(\boldsymbol{Q}) - \left[(\boldsymbol{\Lambda}^\top + \boldsymbol{\Lambda}) \otimes \boldsymbol{I}_m\right] \operatorname{vec}(\boldsymbol{Q}) \\
&= (\boldsymbol{I} \otimes \overline{\boldsymbol{P}}) \operatorname{vec}(\boldsymbol{Q}) - (\boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q} \otimes \boldsymbol{I}_m) \operatorname{vec}(\boldsymbol{Q})
\end{aligned}$$

Hence the Hessian matrix is

$$\nabla^2_{\boldsymbol{QQ}} \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) = \boldsymbol{I} \otimes \overline{\boldsymbol{P}} - \boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q} \otimes \boldsymbol{I}_m$$

The second order necessary condition can be rewritten as

$$\begin{aligned}
\operatorname{vec}(\boldsymbol{Z})^\top \nabla^2_{\boldsymbol{QQ}} \mathcal{L}(\boldsymbol{Q}, \boldsymbol{\Lambda}) \operatorname{vec}(\boldsymbol{Z}) &= \operatorname{vec}(\boldsymbol{Z})^\top \operatorname{vec}(\overline{\boldsymbol{P}} \boldsymbol{Z} - \boldsymbol{Z} \boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q}) \\
&= \operatorname{tr}(\boldsymbol{Z}^\top \overline{\boldsymbol{P}} \boldsymbol{Z} - \boldsymbol{Z}^\top \boldsymbol{Z} \boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q}) \leq 0.
\end{aligned}$$

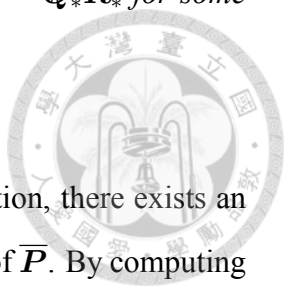
□

Finally, the main theorem of this part can be proved by the first and second order necessary conditions.

Theorem 3.3.5 (Local Maximizer of Optimization Problem)**.** *Let* $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_\ell > \lambda_{\ell+1} \geq \ldots \geq \lambda_m$ *be the eigenvalues of* $\overline{\boldsymbol{P}}$ *and let* $\boldsymbol{Q}_*$ *be the corresponding leading* ℓ

doi:10.6342/NTU201702973

eigenvectors. Then $\boldsymbol{Q}$ is an local maximizer of (+) if and only if $\boldsymbol{Q} = \boldsymbol{Q}_*\boldsymbol{R}_*$ for some $\ell \times \ell$ orthogonal matrix $\boldsymbol{R}_*$.



Proof. By Lemma 3.3.2, for all $\boldsymbol{Q}$ that satisfies the first order condition, there exists an orthogonal matrix $\boldsymbol{R}$ such that each column of $\boldsymbol{QR}$ is an eigenvector of $\overline{\boldsymbol{P}}$. By computing eigenvalue value decomposition,

$$\overline{\boldsymbol{P}} = (\boldsymbol{QR})\boldsymbol{S}(\boldsymbol{QR})^\top + \boldsymbol{Q}_\perp \boldsymbol{S}_\perp \boldsymbol{Q}_\perp^\top$$

where $\boldsymbol{Q}_\perp$ contains $(m-\ell)$ eigenvector that orthogonal to $\boldsymbol{QR}$, $\boldsymbol{S} = \text{diag}(s_1, s_2, \ldots, s_\ell)$, $\boldsymbol{S}_\perp = \text{diag}(s_1^\perp, s_2^\perp, \ldots, s_{m-\ell}^\perp)$ and $s_1, s_2, \ldots, s_\ell, s_1^\perp, s_2^\perp, \ldots, s_{m-\ell}^\perp$ form all eigenvalues of $\overline{\boldsymbol{P}}$. By Lemma 3.3.3, the element $\vec{(\boldsymbol{Z})} \in \mathcal{N}_{\boldsymbol{Q}}$ can be represented as $\boldsymbol{Z} = \boldsymbol{QB} + \boldsymbol{Q}_\perp \boldsymbol{C}$ with $\boldsymbol{B} = -\boldsymbol{B}$. These relations lead to a representation of second order condition as

$$\begin{aligned}
&\text{tr}(\boldsymbol{Z}^\top \overline{\boldsymbol{P}} \boldsymbol{Z} - \boldsymbol{Z}^\top \boldsymbol{Z} \boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q}) \\
=&\text{tr}((\boldsymbol{QB} + \boldsymbol{Q}_\perp \boldsymbol{C})^\top ((\boldsymbol{QR})\boldsymbol{S}(\boldsymbol{QR})^\top + \boldsymbol{Q}_\perp \boldsymbol{S}_\perp \boldsymbol{Q}_\perp^\top)(\boldsymbol{QB} + \boldsymbol{Q}_\perp \boldsymbol{C}) \\
&- (\boldsymbol{QB} + \boldsymbol{Q}_\perp \boldsymbol{C})^\top (\boldsymbol{QB} + \boldsymbol{Q}_\perp \boldsymbol{C}) \boldsymbol{Q}^\top ((\boldsymbol{QR})\boldsymbol{S}(\boldsymbol{QR})^\top + \boldsymbol{Q}_\perp \boldsymbol{S}_\perp \boldsymbol{Q}_\perp^\top) \boldsymbol{Q}) \\
=&\text{tr}(\boldsymbol{B}^\top \boldsymbol{RSR}^\top \boldsymbol{B}) + \text{tr}(\boldsymbol{C}^\top \boldsymbol{S}_\perp \boldsymbol{C}) - \text{tr}(\boldsymbol{B}^\top \boldsymbol{BRSR}^\top) - \text{tr}(\boldsymbol{C}^\top \boldsymbol{CRSR}^\top) \leq 0
\end{aligned}$$

By the properties of trace and the relation $\boldsymbol{B} = -\boldsymbol{B}^\top$, $\text{tr}(\boldsymbol{B}^\top \boldsymbol{BRSR}^\top) = \text{tr}(\boldsymbol{BRSR}^\top \boldsymbol{B}^\top) = \text{tr}(\boldsymbol{B}^\top \boldsymbol{RSR}^\top \boldsymbol{B})$. Also, since $\boldsymbol{RR}^\top = \boldsymbol{I}_\ell$, $\text{tr}(\boldsymbol{C}^\top \boldsymbol{S}_\perp \boldsymbol{C}) = \text{tr}(\boldsymbol{RR}^\top \boldsymbol{C}^\top \boldsymbol{S}_\perp \boldsymbol{C}) = \text{tr}((\boldsymbol{CR})^\top \boldsymbol{S}_\perp (\boldsymbol{CR})$ Hence the second order condition in Lemma 3.3.4 can be written as

$$\begin{aligned}
&\text{tr}(\boldsymbol{B}^\top \boldsymbol{RSR}^\top \boldsymbol{B}) + \text{tr}(\boldsymbol{C}^\top \boldsymbol{S}_\perp \boldsymbol{C}) - \text{tr}(\boldsymbol{B}^\top \boldsymbol{BRSR}^\top) - \text{tr}(\boldsymbol{C}^\top \boldsymbol{CRSR}^\top) \\
=&\text{tr}((\boldsymbol{CR})^\top \boldsymbol{S}_\perp \boldsymbol{CR}) - \text{tr}((\boldsymbol{CR})^\top (\boldsymbol{CR}) \boldsymbol{S}_\perp) \\
=&\sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_i^\perp c_{ij}^2 - \sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_j c_{ij}^2 \leq 0
\end{aligned}$$

for all $\boldsymbol{CR} = [c_{ij}] \in \mathbf{R}^{m \times (m-\ell)}$.

Let $\boldsymbol{Q}_*$ denotes the leading ℓ eigenvectors of $\overline{\boldsymbol{P}}$. Suppose $\boldsymbol{QR}$ contains eigenvector other than leading ℓ singular vectors of $\overline{\boldsymbol{P}}$. Then $s_a^\perp > s_b$ for some a and b. Now pick $[c_{ij}]$

doi:10.6342/NTU201702973

as $c_{ab} = 1$ and $c_{ij} = 0$ otherwise. Then

$$\sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_i^{\perp} c_{ij}^2 - \sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_j c_{ij}^2 = s_a^{\perp} - s_b > 0.$$

This inequality conflicts to the second order necessary condition. On the other hand, suppose $\boldsymbol{QR} = \boldsymbol{Q}_*$. Then $\boldsymbol{S} = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_\ell)$ and $\boldsymbol{S}_\perp = \text{diag}(\lambda_{\ell+1}, \ldots, \lambda_m)$. Hence for all $[c_{ij}] \in \mathbf{R}^{m\times(m-\ell)}$.

$$\sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_i^{\perp} c_{ij}^2 - \sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_j c_{ij}^2 \leq (\lambda_{\ell+1} - \lambda_\ell) \sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_j c_{ij}^2 \leq 0$$

by $(\lambda_{\ell+1} - \lambda_\ell) < 0$ and $\sum_{i=1}^{(m-\ell)} \sum_{j=1}^{\ell} s_j c_{ij}^2 \leq 0$. This inequality obeys the second order necessary condition.

To sum up, a feasible point $\boldsymbol{Q}$ satisfies the first and second order necessary condition if and only if $\boldsymbol{Q} = \boldsymbol{Q}_*\boldsymbol{R}_*$ for some orthogonal matrix $\boldsymbol{R}_* = \boldsymbol{R}^\top$. Lemma 3.1.1 shows that $\boldsymbol{Q}_*\boldsymbol{R}_*$ is global maximizer, and hence local maximizer. □

Theorem 3.3.5 provides that the only type of local maximizer is the orthogonal matrix that formed by the leading ℓ eigenvectors of the matrix $\overline{\boldsymbol{P}}$ (with some right orthogonal transformation), which is the integrated subspaces defined in (2.1) or (3.1). This result shows that if a line search can prevent the iterator from saddle points, it can find out the integrated subspace successfully.

doi:10.6342/NTU201702973

Chapter 4

Integration Method

As shown in Chapter 2, the integrated subspace is the leading eigenvectors of the matrix $\overline{\boldsymbol{P}}$, or the singular vectors of the matrix $\left[\boldsymbol{Q}_{[1]}|\boldsymbol{Q}_{[2]}|\cdots|\boldsymbol{Q}_{[N]}\right]$ .. Therefore, canonical SVD (for example, the SVD routine in MATLAB or LAPACK [2]) can be applied to solve this eigenvectors problem. However, the computational complexity of canonical SVD is $O(N^2m\ell^2)$, which is much slower when N increasing. Hence in this chapter, we introduce some other methods to compute the integrated subspace with different computational complexity than $O(N^2m\ell^2)$.

4.1 Line Search Type Method

Since the integrated subspace is defined by an optimization problem, it is reasonable to use the algorithm for solving the optimization problem for computing the integrated subspace. A classical method for solving the optimization problem is line search methods. Gradient descent method (in our case, gradient ascent method) is widely used among the line search methods. The update scheme for unconstraint gradient ascent method is

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \tau_k \nabla f(\boldsymbol{x}_k)$$

where f is the objective function and τ_k is the step size needed to search in the n-th iteration. In the viewpoint of line search, in each iteration, we define a curve $\boldsymbol{\gamma}_k(\tau) =$

doi:10.6342/NTU201702973

$\boldsymbol{x}_k + \tau_k \nabla f(\boldsymbol{x}_k)$. This curve is a straight line stating at the point $\boldsymbol{x}_k$ and point to the direction $\nabla f(\boldsymbol{x}_k)$, which is the steeps ascent direction of objective function at $\boldsymbol{x}$. The point for next iteration is given by finding a suitable step size τ and define $\boldsymbol{x}_{k+1} = \gamma_k(\tau)$. These explanations shows the main concept of gradient ascent method. It tries to find the optimal solution by walking along the curve with steepest direction in each step with suitable step size.

The main issue for this algorithm is to find a suitable step size τ. If the step size is too large, it may be difficult to reach the optimal solution because it may walk too far. However, if the step size is too small, it may need more iteration to reach the optimal solution. One of the popular methods for selecting step size is using the back tracking method with Armijo-Wolfe condition (for maximization problem)

$$f(\boldsymbol{x}_k + \tau_k \boldsymbol{p}_k) \geq f(\boldsymbol{x}_k) + \rho \tau_k \boldsymbol{p}_k^\top \nabla f(\boldsymbol{x}_k)$$
$$\boldsymbol{p}_k^\top \nabla f(\boldsymbol{x}_k + \tau_k \boldsymbol{p}_k) \leq \rho_2 \boldsymbol{p}_k^\top \nabla f(\boldsymbol{x}_k)$$

where ρ, ρ_2 is parameters that can be chosen and $\boldsymbol{p}_k = \nabla f(\boldsymbol{x}_k)$ in gradient ascend. The second condition always holds due to the design of algorithm. Hence we just need to focus on the first condition. We use Armijo rule for simplicity in the following content. The back tracking method tries to find the step size τ_k satisfied the Armijo rule by the following steps. First, assign an initial guess of step size, such as 1. Then test whether this step size satisfies the Armijo rule. Suppose not, multiply this step size by a positive constant $\beta < 1$ and test it again. Repeat these processes until the step size satisfies the Armijo rule and set it into τ_k.

For the constraint optimization problem, the gradient ascent method needs to be modified. Here we choose to use the viewpoint of the gradient ascent method on Stiefel manifold. The Stiefel manifold is the manifold consisted of all $m \times \ell$ orthogonal matrices

$$\mathcal{S}_{m,\ell} = \left\{ \boldsymbol{Q} \in \mathbf{R}^{m \times \ell} : \boldsymbol{Q}^\top \boldsymbol{Q} = \boldsymbol{I} \right\}$$

which is exactly our constraint in the optimization problem.

doi:10.6342/NTU201702973

Define

$$F(\boldsymbol{Q}) = \frac{1}{2}\operatorname{tr}(\boldsymbol{Q}^\top \overline{\boldsymbol{P}} \boldsymbol{Q})$$

as our objective function. We denote the gradient of F at $\boldsymbol{Q}$ as

$$\boldsymbol{G}_F(\boldsymbol{Q}) = \overline{\boldsymbol{P}}\boldsymbol{Q}$$

where the equality can be computed easily. On the manifold, the gradient in the Euclidean space may not represent the direction (tangent) at a point. Hence we need to project the gradient to the tangent space on the manifold for further derivation.

Lemma 4.1.1. *The projected gradient of F onto the tangent space $\mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$ of Stiefel manifold $\mathcal{S}_{m,\ell}$ is*

$$\boldsymbol{D}_F(\boldsymbol{Q}) = (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}.$$

Proof. First we find a necessary and sufficient condition for $\boldsymbol{X}$ being in $\mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$. For all $\boldsymbol{X} \in \mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$, find a path $\boldsymbol{\Gamma}(t)$ in $\mathcal{S}_{m,\ell}$ with $\boldsymbol{\Gamma}(0) = \boldsymbol{Q}$ and $\boldsymbol{\Gamma}'(0) = \boldsymbol{X}$. From $\boldsymbol{\Gamma}(t)^\top\boldsymbol{\Gamma}(t) = \boldsymbol{I}$, differentiate each side by t and take $t = 0$, we have

$$\boldsymbol{X}^\top\boldsymbol{Q} + \boldsymbol{Q}^\top\boldsymbol{X} = \boldsymbol{0}, \tag{4.1}$$

which gives a necessary condition for $\boldsymbol{X} \in \mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$. There are $\ell(\ell+1)/2$ conditions for $\boldsymbol{X}$ in (4.1) and the dimension of $\mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$ is $m\ell - \ell(\ell+1)/2$, which means (4.1) is also a sufficient condition for $\boldsymbol{X} \in \mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$. By taking vec to each sides of (4.1), we get the equality

$$[(\boldsymbol{Q}^\top \otimes \boldsymbol{I}_\ell)\boldsymbol{K}_{m,\ell} + (\boldsymbol{I}_\ell \otimes \boldsymbol{Q}^\top)]\operatorname{vec}(\boldsymbol{X}) = \boldsymbol{0}.$$

Define $\boldsymbol{T} = \boldsymbol{K}_{\ell,m}(\boldsymbol{Q} \otimes \boldsymbol{I}_\ell) + (\boldsymbol{I}_\ell \otimes \boldsymbol{Q})$ and get $\boldsymbol{T}^\top \operatorname{vec}(\boldsymbol{X}) = \boldsymbol{0}$. This shows that the tangent space (after vectorizing each elements) is contained in the null space of $\boldsymbol{T}^\top$. One can compute the rank of $\boldsymbol{T}$ and shows that the null space of $\boldsymbol{T}^\top$ is actually the tangent space. Hence the projection matrix onto the tangent space is given by $(\boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{T}})$, where $\boldsymbol{P}_{\boldsymbol{T}} = \boldsymbol{T}(\boldsymbol{T}^\top\boldsymbol{T})^+\boldsymbol{T}^\top$ and $(\boldsymbol{T}^\top\boldsymbol{T})^+$ denoted the Moore-Penrose pseudo-inverse. With $\boldsymbol{P}_{\boldsymbol{T}}$,

doi:10.6342/NTU201702973

$\boldsymbol{D}_F$ can be given via $\text{vec}(\boldsymbol{D}_F) = (\boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{T}})\,\text{vec}(\boldsymbol{G}_F)$. With some calculation, we have $\boldsymbol{T} = (\boldsymbol{I}_\ell \otimes \boldsymbol{Q})(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell})$ and thus

$$\begin{aligned}\boldsymbol{T}^\top\boldsymbol{T} &= (\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell})^\top(\boldsymbol{I}_\ell \otimes \boldsymbol{Q})^\top(\boldsymbol{I}_\ell \otimes \boldsymbol{Q})(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell}) \\ &= (\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell})(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell}) = 2(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell}).\end{aligned}$$

Then the projection matrix $\boldsymbol{P}_{\boldsymbol{T}}$ can be calculated as:

$$\begin{aligned}\boldsymbol{P}_{\boldsymbol{T}} &= \boldsymbol{T}(\boldsymbol{T}^\top\boldsymbol{T})^+\boldsymbol{T}^\top \\ &= (\boldsymbol{I}_\ell \otimes \boldsymbol{Q})(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell})\frac{1}{2}(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell})^+(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell})^\top(\boldsymbol{I}_\ell \otimes \boldsymbol{Q})^\top \\ &= \frac{1}{2}(\boldsymbol{I}_\ell \otimes \boldsymbol{Q})(\boldsymbol{I}_{\ell^2} + \boldsymbol{K}_{\ell,\ell})(\boldsymbol{I}_\ell \otimes \boldsymbol{Q}^\top)\frac{1}{2}(\boldsymbol{I}_\ell \otimes \boldsymbol{Q}\boldsymbol{Q}^\top) + \frac{1}{2}(\boldsymbol{Q}^\top \otimes \boldsymbol{Q})\boldsymbol{K}_{m,\ell}.\end{aligned}$$

Hence, by $\text{vec}(\boldsymbol{D}_F) = (\boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{T}})\,\text{vec}(\boldsymbol{G}_F)$,

$$\begin{aligned}\text{vec}(\boldsymbol{D}_F) &= (\boldsymbol{I}_{\ell^2} - \frac{1}{2}(\boldsymbol{I}_\ell \otimes \boldsymbol{Q}\boldsymbol{Q}^\top) - \frac{1}{2}(\boldsymbol{Q}^\top \otimes \boldsymbol{Q})\boldsymbol{K}_{m,\ell})\,\text{vec}(\boldsymbol{G}_F) \\ &= \text{vec}(\boldsymbol{G}_F) - \frac{1}{2}(\boldsymbol{I}_\ell \otimes \boldsymbol{Q}\boldsymbol{Q}^\top)\,\text{vec}(\boldsymbol{G}_F) - \frac{1}{2}(\boldsymbol{Q}^\top \otimes \boldsymbol{Q})\boldsymbol{K}_{m,\ell}\,\text{vec}(\boldsymbol{G}_F) \\ &= \text{vec}(\boldsymbol{G}_F) - \frac{1}{2}\,\text{vec}(\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}_F) - \frac{1}{2}\,\text{vec}(\boldsymbol{Q}\boldsymbol{G}_F^\top\boldsymbol{Q})\end{aligned}$$

and $\boldsymbol{D}_F$ can be written as

$$\boldsymbol{D}_F = \left(\boldsymbol{I} - \frac{1}{2}\boldsymbol{Q}\boldsymbol{Q}^\top\right)\boldsymbol{G}_F - \frac{1}{2}\boldsymbol{Q}\boldsymbol{G}_F^\top\boldsymbol{Q}. \tag{4.2}$$

Since we have the property $\boldsymbol{Q}^\top\boldsymbol{G}_F(\boldsymbol{Q}) = \boldsymbol{G}_F(\boldsymbol{Q})^\top\boldsymbol{Q}$ here, we can get $\boldsymbol{D}_F(\boldsymbol{Q}) = (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\boldsymbol{G}_F(\boldsymbol{Q})$. This completes the proof. □

Algorithm 3 is rewritten from [1] by the notation in this thesis and the projected gradient as above. Note that the function $\overline{F}(\boldsymbol{M})$ denote the function that orthogonalize $\boldsymbol{M}$ first (hence this point is in the Stiefel manifold) and then plug into F.

The basic concept of Algorithm 3 is same as the gradient ascent method described previously. For each step, we find the projected gradient, which is the steepest direction on the manifold. Then we walk along this direction with a suitable step size decide by backtracking and Armijo-Wolfe condition. The difference is that we need to retract back

doi:10.6342/NTU201702973

Algorithm 3 Integration of $\{\boldsymbol{Q}_{[i]}\}_{i=1}^{N}$ based on Armijo line search.

Require: $\boldsymbol{Q}_{[1]}, \boldsymbol{Q}_{[2]}, \ldots, \boldsymbol{Q}_{[N]}$ (subspace matrices), $\boldsymbol{Q}_{\text{ini}}$ (initial guess), $\tau_0 > 0$ (initial step size), $\beta \in (0,1)$ (scaling parameter for step size searching), $\rho \in (0,1)$ (parameters for step size searching)

Ensure: Integrated subspace matrix $\overline{\boldsymbol{Q}}$ based on Armijo line search

1: Initialize the current iterate $\boldsymbol{Q}_c \leftarrow \boldsymbol{Q}_{\text{ini}}$
2: **while** (not convergent) **do**
3: Compute the gradient on manifold $\boldsymbol{X} = (\boldsymbol{I}_m - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}_c$
4: Find the smallest integer $j \geq 0$ such that the following inequality holds:

$$\overline{F}(\boldsymbol{Q}_c + \tau_0\beta^j\boldsymbol{X}) \geq F(\boldsymbol{Q}_c) + \tau_0\beta^j\rho\,\|\boldsymbol{X}\|_F^2$$

5: Orthogonalize $(\boldsymbol{Q}_c + \tau_0\beta^j\boldsymbol{X})$ (for example, by QR-decomposition or polar decomposition) as $\boldsymbol{Q}_+$
6: Assign $\boldsymbol{Q}_c \leftarrow \boldsymbol{Q}_+$
7: **end while**
8: Output $\overline{\boldsymbol{Q}} = \boldsymbol{Q}_c$

to the manifold before we calculate the objective function and before the next iteration. In this algorithm, it uses orthogonalization as the retraction.

In [1], the convergent theorem is also provided. Here we translate the theorem and write down the most important part as Theorem 4.1.2. Note that in the theorem, a critical point x_* (the points that make projected gradient is $\mathbf{0}$) is stable, if for any neighborhood $\mathcal{U}$ around x_*, there exists another neighborhood $\mathcal{V}$ around x_*, such that if the initial value start in $\mathcal{V}$, then after any finite steps, the result will be in $\mathcal{U}$. A critical point is asymptotically stable if it is stable, and the condition also holds as the number of steps tends to infinity. A critical point is unstable if it is not stable.

Theorem 4.1.2 (Convergence theory for Algorithm 3)**.** *Let* $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_\ell > \lambda_{\ell+1} \geq \ldots \geq \lambda_m$ *be the eigenvalues of* $\overline{\boldsymbol{P}}$ *and let* $\boldsymbol{Q}_*$ *be the corresponding leading* ℓ *eigenvectors. Algorithm 3 converge to the orthogonal matrix* $\boldsymbol{Q}$ *with each column are eigenvectors of* $\overline{\boldsymbol{P}}$*. Among all the critical points,* $\boldsymbol{Q}_*$ *(up to a right orthogonal transform) is the only asymptotically stable point with the linear convergent rate. Other critical points are unstable.*

This theorem gives the promise of the convergence. Also, our result for the property of optimization problem in Chapter 3 can support this result. The critical points are actually the points satisfying first order condition 3.2. Among these points, only $\boldsymbol{Q}_*$ satisfies the

doi:10.6342/NTU201702973

second order condition, which is related to the condition of stable points.

However, same as the traditional gradient ascent method, the convergent rate of the Algorithm 3 is linear. This could be a problem since it might take many steps to converge. Here we provide another algorithm. Algorithm 4 is rewritten by the notation and objective function in this thesis from the algorithm proposed by Wen and Yin [11].

This algorithm is based on gradient ascent method which is similar to Algorithm 3. However, some modification is introduced for this algorithm. The method to retract the point back to the manifold is changed in this modified algorithm. The searching path at current point $\boldsymbol{Q}_c$ is defined as $\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\tau) = (\boldsymbol{I} - \frac{\tau}{2}\boldsymbol{M})^{-1}(\boldsymbol{I} + \frac{\tau}{2}\boldsymbol{M})\boldsymbol{Q}_c$, where $\boldsymbol{M} = \boldsymbol{G}\boldsymbol{Q}_c^\top - \boldsymbol{Q}_c\boldsymbol{G}^\top$ and $\boldsymbol{G} = \overline{\boldsymbol{P}}\boldsymbol{Q}_c$ is the gradient of the objective function in Euclidean space. By using Woodbury matrix identity, the searching path is same as

$$\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\tau) = \boldsymbol{Q}_c - \tau\boldsymbol{L}(\boldsymbol{I}_{2\ell} + \frac{1}{2}\tau\boldsymbol{R}^\top\boldsymbol{L})^{-1}\boldsymbol{R}^\top\boldsymbol{Q}_c \tag{4.3}$$

where $\boldsymbol{L} = [-\boldsymbol{G} \;\; \boldsymbol{Q}_c]$ and $\boldsymbol{R} = [\boldsymbol{Q}_c \;\; \boldsymbol{G}]$. This modification decreases the matrix size to compute inverse. As mentioned in [11], this path also satisfies some properties, hence Theorem 4.1.2 also holds for this algorithm.

Algorithm 4 also uses the Barzilai-Borwein step size [3] (BB step size) to accelerate this gradient method, and the nonmonotone strategy in [12] to prevent stuck in the local optimal points. The Armijo rule in the Algorithm 4 is

$$F(\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\tau)) \geq F(\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(0)) + \tau\sigma \left.\frac{\mathrm{d}F(\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(t))}{\mathrm{d}t}\right|_{t=0}.$$

The nonmonotone strategy in [12] modified the Armijo rule by

$$F(\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\tau)) \geq c + \tau\sigma \left.\frac{\mathrm{d}F(\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\tau))}{\mathrm{d}t}\right|_{t=0}$$

where c is a scalar updated in each iteration by $c \leftarrow (\eta\zeta c + F((\boldsymbol{Q}_+))/(\eta\zeta + 1)$ and then $\zeta \leftarrow \eta\zeta + 1$ for a given parameter η. The initial value of c is $F(\boldsymbol{Q}_{\text{ini}})$. This condition soften the Armijo rule and increase the chance to jump out the local minimum.

doi:10.6342/NTU201702973

The BB step size is the simulation for Newton's method, in a more efficient way. They consider the update scheme $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{S}_k\boldsymbol{g}_k$. In Newton's method, the matrix $\boldsymbol{S}_k$ is the Hessian matrix of objective function. They set the matrix $\boldsymbol{S}_k = \tau_k\boldsymbol{I}$, where τ_k is the step size needed to computed. Also, they wish $\boldsymbol{S}_k$ can approximately satisfies the secant condition $\Delta\boldsymbol{x} = \boldsymbol{S}_k\Delta\boldsymbol{g}$ or $\boldsymbol{S}_k\Delta\boldsymbol{x} = \Delta\boldsymbol{g}$ in quasi-Newton's method, where $\Delta\boldsymbol{x} = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$, $\Delta\boldsymbol{g} = \boldsymbol{g}_k - \boldsymbol{g}_{k-1}$. Hence the step size τ_k is determined by

$$\tau_k = \underset{\tau}{\operatorname{argmin}} \|\Delta\boldsymbol{x} - \tau\Delta\boldsymbol{g}\| \text{ or } \tau_k = \underset{\tau}{\operatorname{argmin}} \|\tau\Delta\boldsymbol{x} - \Delta\boldsymbol{g}\|.$$

The solutions of these minimization problems are

$$\alpha_k = \frac{\langle\Delta\boldsymbol{x}, \Delta\boldsymbol{g}\rangle}{\langle\Delta\boldsymbol{g}, \Delta\boldsymbol{g}\rangle} \text{ or } \alpha_k = \frac{\langle\Delta\boldsymbol{x}, \Delta\boldsymbol{x}\rangle}{\langle\Delta\boldsymbol{x}, \Delta\boldsymbol{g}\rangle}$$

These tow step size are called BB step size. In our case, the BB step size are

$$\tau_{guess} = \frac{\operatorname{tr}(\boldsymbol{D}_1^\top\boldsymbol{D}_1)}{|\operatorname{tr}(\boldsymbol{D}_1^\top\boldsymbol{D}_2)|} \text{ or } \frac{|\operatorname{tr}(\boldsymbol{D}_1^\top\boldsymbol{D}_2)|}{\operatorname{tr}(\boldsymbol{D}_2^\top\boldsymbol{D}_2)}$$

where $\boldsymbol{D}_1 = \boldsymbol{Q}_+ - \boldsymbol{Q}_c$ and $\boldsymbol{D}_2 = \boldsymbol{X}_+ - \boldsymbol{X}_c$. However, BB step size does not imply the convergence. Therefore, Algorithm 4 still need to use back tracking method with Armijo-Wolfe conditions to ensure the convergence of the algorithm.

In each iteration, the most intensive calculation in Algorithm 4 is the multiplication $\boldsymbol{G} = \overline{\boldsymbol{P}}\boldsymbol{Q}_c$. This multiplication can be computed by $\sum_{i=1}^N \boldsymbol{Q}_{[i]}^\top\boldsymbol{Q}_c$ first and then $\boldsymbol{G} = \sum_{i=1}^N \boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top\boldsymbol{Q}_c$. The first step contains N matrix multiplication with size $\ell \times m$ and $m \times \ell$. The second step contains N matrix multiplication with size $m \times \ell$ and $\ell \times \ell$. Hence the complexity for computing $\boldsymbol{G}$ is $O(Nm\ell^2)$.

The remaining steps in each iteration can be computed by using $\boldsymbol{G}$. First, we consider the part for line search. The value of objective function can be computed by $F(\boldsymbol{Q}) = \operatorname{tr}(\boldsymbol{Q}^\top\boldsymbol{G})$, which is the summation of dot product of two $m \times \ell$ matrices. The complexity is $O(m\ell)$. The point on the curve $\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\tau)$ can be solved by a linear system with dimension ℓ matrix multiplication of size $m \times 2\ell$ and $2\ell \times m$, and $m \times 2\ell$ and $2\ell \times 2\ell$. The complexity

doi:10.6342/NTU201702973

Algorithm 4 Integration of $\{\boldsymbol{Q}_{[i]}\}_{i=1}^{N}$ based on nonmonotone line search with BB step size.

Require: $\boldsymbol{Q}_{[1]}, \boldsymbol{Q}_{[2]}, \ldots, \boldsymbol{Q}_{[N]}$ (subspace matrices), $\boldsymbol{Q}_{\text{ini}}$ (initial guess), $\tau_0 > 0$ (initial step size), $\beta \in (0,1)$ (scaling parameter for step size searching), $\rho \in (0,1)$ (parameter for step size searching), $\eta \in (0,1)$ (parameter for next step searching), τ_M, τ_m (maximum and minimum for predicting step size)

Ensure: Integrated subspace matrix $\overline{\boldsymbol{Q}}$ based on Armijo line search with BB step size

1: Initialize $\boldsymbol{Q}_c \leftarrow \boldsymbol{Q}_{\text{ini}}$, $\bar{\tau} \leftarrow \tau_0$, $\zeta = 1$, $c = F(\boldsymbol{Q}_c)$

2: **while** (not convergent) **do**

3: Compute the gradient in Euclidean space $\boldsymbol{G} = \overline{\boldsymbol{P}}\boldsymbol{Q}_c$

4: Set $\boldsymbol{L} = [-\boldsymbol{G} \;\; \boldsymbol{Q}_c]$ and $\boldsymbol{R} = [\boldsymbol{Q}_c \;\; \boldsymbol{G}]$.

5: Find the smallest integer $j \geq 0$ such that the following inequality holds:

$$F(\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\bar{\tau}\beta^j)) \geq c + \bar{\tau}\beta^j\rho \left\|\boldsymbol{Q}_c\boldsymbol{G}^\top - \boldsymbol{G}\boldsymbol{Q}_c^\top\right\|_F^2$$

where $\boldsymbol{\Gamma}_{\boldsymbol{Q}_c}(\tau) = \boldsymbol{Q}_c - \tau\boldsymbol{L}(\boldsymbol{I}_{2\ell} + \frac{1}{2}\tau\boldsymbol{R}^\top\boldsymbol{L})^{-1}\boldsymbol{R}^\top\boldsymbol{Q}_c$

6: Assign $\boldsymbol{Q}_+ = \boldsymbol{Q}_d(\bar{\tau}\beta^j))$

7: Update $c \leftarrow (\eta\zeta c + F((\boldsymbol{Q}_+))/(\eta\zeta + 1)$ and then $\zeta \leftarrow \eta\zeta + 1$

8: Compute the differences $\boldsymbol{D}_1 = \boldsymbol{Q}_+ - \boldsymbol{Q}_c$ and $\boldsymbol{D}_2 = \boldsymbol{X}_+ - \boldsymbol{X}_c$, where

$$\boldsymbol{X}_c = (\boldsymbol{I}_m - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}_c$$
$$\boldsymbol{X}_+ = (\boldsymbol{I}_m - \boldsymbol{Q}_+\boldsymbol{Q}_+^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}_+$$

9: Assign $\bar{\tau} \leftarrow \max(\min(\tau_{guess}, \tau_M), \tau_m)$, where

$$\tau_{guess} = \frac{\text{tr}(\boldsymbol{D}_1^\top\boldsymbol{D}_1)}{|\,\text{tr}(\boldsymbol{D}_1^\top\boldsymbol{D}_2)|} \text{ or } \frac{|\,\text{tr}(\boldsymbol{D}_1^\top\boldsymbol{D}_2)|}{\text{tr}(\boldsymbol{D}_2^\top\boldsymbol{D}_2)}$$

10: Assign $\boldsymbol{Q}_c \leftarrow \boldsymbol{Q}_+$

11: **end while**

12: Output $\overline{\boldsymbol{Q}} = \boldsymbol{Q}_c$

doi:10.6342/NTU201702973

is $O(m\ell^2 + \ell^3)$. Hence the total complexity for line search is $O(I_{inner}(m\ell^2 + \ell^3))$, where I_{inner} denotes the number of iteration of line search (inner loop). Second, we consider the part for updating c. The main calculation of this part is the computation of the objective function. Hence the complexity of this part is $O(m\ell)$. Finally, we consider the part for computing BB step size. The main computation in this part is $\boldsymbol{X}_+$, which need to calculate $\boldsymbol{G}_+ = \overline{\boldsymbol{P}}\boldsymbol{Q}_+$. However, this computation can be directly used in next iteration. Hence we compute the complexity of this part in the next iteration. For other components, they need matrix multiplication with complexity $O(m\ell^2)$. Also, it needs $O(m\ell)$ to compute each trace for computing τ_{guess}.

To sum up, suppose we use I_{WY} iteration to converge, then the computational complexity of Algorithm 4 is $I_{WY}(O(Nm\ell^2)+O(I_{inner}(m\ell^2+\ell^3))+O(m\ell)+O(m\ell^2+m\ell))$, which is dominate by the term $O(I_{WY}Nm\ell^2)$, suppose I_{inner} is controlled. (This assumption is reasonable since I_{inner} is often restricted within some number.)

4.2 Kolmogorov-Nagumo-Type Average

Besides using the viewpoint from line search method, one can use another viewpoint from the average on Stiefel manifold to compute the integrated subspace. Algorithm 5 is inspired by the Kolmogorov-Nagumo-type average on Stiefel manifold [8, 5]. If a pair of retraction map and lifting map is defined, then the algorithm can be generate by a fixed point method scheme. In these paper, a retraction map at a point $\boldsymbol{Q}$ on Stiefel manifold is defined as a map $\varphi_{\boldsymbol{Q}}^{-1} : \mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell} \to \mathcal{S}_{m,\ell}$ from the tangent space at $\boldsymbol{Q}$ to the Stiefel manifold and satisfies the following three conditions: (1) $\varphi_{\boldsymbol{Q}}^{-1}$ can be defined around $\mathbf{0} \in \mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$, (2) $\varphi_{\boldsymbol{Q}}^{-1}(0) = \boldsymbol{Q}$ and (3) $\left.\frac{d\varphi_{\boldsymbol{Q}}^{-1}(t\boldsymbol{X})}{dt}\right|_{t=0} = \boldsymbol{X}$. The corresponding lifting map is a map $\varphi_{\boldsymbol{Q}} : \mathcal{S}_{m,\ell} \to \mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$ from the Stiefel manifold to the tangent space and satisfies $\varphi_{\boldsymbol{Q}}^{-1}(\varphi_{\boldsymbol{Q}}(\boldsymbol{W})) = \boldsymbol{W}$. The fixed point scheme defied by this pair of lifting and retraction map is

$$\boldsymbol{Q}_+ = \varphi_{\boldsymbol{Q}_c}^{-1}\left(\frac{1}{N}\sum_{i=1}^{N}\varphi_{\boldsymbol{Q}_c}(\boldsymbol{Q}_{[i]})\right).$$

Note that $\boldsymbol{Q}_+$ is an Kolmogorove-Nagumo average of $\boldsymbol{Q}_{[i]}$.

doi:10.6342/NTU201702973

Algorithm 5 Integration of $\{\boldsymbol{Q}_{[i]}\}_{i=1}^{N}$ as Kolmogorov-Nagumo-type average.

Require: $\boldsymbol{Q}_{[1]}, \boldsymbol{Q}_{[2]}, \ldots, \boldsymbol{Q}_{[N]}$ (subspaces matrices), $\boldsymbol{Q}_{\text{ini}}$ (initial guest)
Ensure: Integrated subspace matrix $\overline{\boldsymbol{Q}}$ based on the Kolmogorov-Nagumo-type average

1: Initialize the current iterate $\boldsymbol{Q}_c \leftarrow \boldsymbol{Q}_{\text{ini}}$
2: **while** (not convergent) **do**
3: Perform the lifting map and average $\boldsymbol{X} = (\boldsymbol{I}_m - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}_c$
4: Perform the retraction map $\boldsymbol{Q}_+ \leftarrow \boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{X}\boldsymbol{C}^{-1}$,
where $\boldsymbol{C} = \left\{\frac{\boldsymbol{I}}{2} + \left(\frac{\boldsymbol{I}}{4} - \boldsymbol{X}^\top\boldsymbol{X}\right)^{1/2}\right\}^{1/2}$
5: Assign $\boldsymbol{Q}_c \leftarrow \boldsymbol{Q}_+$
6: **end while**
7: Output $\overline{\boldsymbol{Q}} = \boldsymbol{Q}_c$

To connect the KN-type average to our integration problem (3.1), the lifting map is chosen as

$$\varphi_{\boldsymbol{Q}_c}(\boldsymbol{W}) = (\boldsymbol{I}_m - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c$$

which is an element in tangent space since $\boldsymbol{Q}_c^\top\varphi_{\boldsymbol{Q}_c}(\boldsymbol{W}) + \varphi_{\boldsymbol{Q}_c}(\boldsymbol{W})^\top\boldsymbol{Q}_c = \boldsymbol{0} + \boldsymbol{0} = \boldsymbol{0}$ for any $\boldsymbol{W} \in \mathcal{S}_{m,\ell}$. This choice makes the average of $\boldsymbol{Q}_{[i]}$ on tangent space becomes

$$\sum_{i=1}^{N}\varphi_{\boldsymbol{Q}_c}(\boldsymbol{Q}_{[i]}) = \sum_{i=1}^{N}(\boldsymbol{I}_m - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top\boldsymbol{Q}_c = \boldsymbol{D}_F(\boldsymbol{Q}_c)$$

which is the projected gradient at $\boldsymbol{Q}_c$. The problem now is what the corresponding retraction map is. As proposed in [4], the corresponding retraction map is chosen as

$$\varphi_{\boldsymbol{Q}_c}^{-1}(\boldsymbol{X}) = \boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{X}\boldsymbol{C}^{-1} \tag{4.4}$$

where $\boldsymbol{C} = \left\{\frac{\boldsymbol{I}}{2} + \left(\frac{\boldsymbol{I}}{4} - \boldsymbol{X}^\top\boldsymbol{X}\right)^{1/2}\right\}^{1/2}$. Algorithm 5 summarize these two maps with the fixed point scheme.

Here we give a quick derivation and check for the retraction map. For more detailed results, please refer to [4]. We want to find the retraction map $\varphi_{\boldsymbol{Q}_c}^{-1}(\boldsymbol{X})$. Observe that from our choice of lifting map, we have the condition $\boldsymbol{Q}_c^\top\varphi_{\boldsymbol{Q}_c}(\boldsymbol{W}) = \boldsymbol{0}$ for any $\boldsymbol{W} \in \mathcal{S}_{m,\ell}$. Also, the condition we want for the retraction map is $\varphi_{\boldsymbol{Q}}^{-1}(\varphi_{\boldsymbol{Q}}(\boldsymbol{W})) = \boldsymbol{W}$. Hence the condition $\boldsymbol{X} = \varphi_{\boldsymbol{Q}}(\boldsymbol{W})$ is a reasonable assumption. Also, assume $\varphi_{\boldsymbol{Q}_c}^{-1}(\boldsymbol{X}) = \boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{X}\boldsymbol{B}$. This assumption means that $\varphi_{\boldsymbol{Q}_c}^{-1}(\boldsymbol{X})$ is spanned by the columns of $\boldsymbol{Q}_c$ and $\boldsymbol{X}$.

doi:10.6342/NTU201702973

By using the condition $\varphi_{\boldsymbol{Q}_c}^{-1}(\boldsymbol{X}) \in \mathcal{S}_{m,\ell}$ and $\varphi_{\boldsymbol{Q}}^{-1}(\varphi_{\boldsymbol{Q}}(\boldsymbol{W})) = \boldsymbol{W}$, we get the following equations

$$\begin{cases} \boldsymbol{I} = (\boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{X}\boldsymbol{B})^\top(\boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{X}\boldsymbol{B}) \\ \boldsymbol{W} = \boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{X}\boldsymbol{B} \\ \boldsymbol{X} = (\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c \end{cases}.$$

Plug the second equation into third equation, we get $\boldsymbol{X} = \boldsymbol{X}\boldsymbol{B}\boldsymbol{C}^\top$. Here we add an assumption that $\boldsymbol{C}$ is invertible and symmetric. Hence we have $\boldsymbol{X}\boldsymbol{B} = \boldsymbol{X}\boldsymbol{C}^{-1}$. Plug this back to the first equation and get $\boldsymbol{I} = \boldsymbol{C}^2 + \boldsymbol{C}^{-1}\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{C}^{-1}$. This equation leads to $\boldsymbol{C}^2 = \boldsymbol{C}^4 + \boldsymbol{X}^\top\boldsymbol{X}$. Square this equation and get $(\boldsymbol{C}^2 - \frac{\boldsymbol{I}}{2})^2 = \frac{\boldsymbol{I}}{4} - \boldsymbol{X}^\top\boldsymbol{X}$, and hence $\boldsymbol{C} = \left\{\frac{\boldsymbol{I}}{2} + \left(\frac{\boldsymbol{I}}{4} - \boldsymbol{X}^\top\boldsymbol{X}\right)^{1/2}\right\}^{1/2}$. To sum up, our guessed retraction map is same as the one given in (4.4).

No matter how many assumptions that we use for guessing the retraction map, we just need to check whether (4.4) is a corresponding retraction. We need to check the three conditions for retraction first. The guessed retraction map $\varphi_{\boldsymbol{Q}_c}^{-1}(\boldsymbol{X})$ can be defined around $\boldsymbol{0} \in \mathcal{T}_{\boldsymbol{Q}}\mathcal{S}_{m,\ell}$ since $\boldsymbol{C}$ can be defined for $\text{tr}(\boldsymbol{X}^\top\boldsymbol{X}) \leq \frac{1}{4}$. The guessed retraction map satisfies $\varphi_{\boldsymbol{Q}}^{-1}(0) = \boldsymbol{Q}$ by directly computation. To check the third condition, we write $\varphi_{\boldsymbol{Q}}^{-1}(t\boldsymbol{X}) = \boldsymbol{Q}\boldsymbol{C}(t) + t\boldsymbol{X}\boldsymbol{C}^{-1}(t)$. Derivative t to the both sides of condition $\boldsymbol{C}(t)^4 - \boldsymbol{C}(t)^2 + t^2\boldsymbol{X}^\top\boldsymbol{X} = \boldsymbol{0}$, we get

$$\begin{aligned} &\frac{d\boldsymbol{C}(t)}{dt}\boldsymbol{C}(t)^3 + \boldsymbol{C}(t)\frac{d\boldsymbol{C}(t)}{dt}\boldsymbol{C}(t)^2 + \boldsymbol{C}(t)^2\frac{d\boldsymbol{C}(t)}{dt}\boldsymbol{C}(t) + \boldsymbol{C}(t)^3\frac{d\boldsymbol{C}(t)}{dt} \\ &+ \frac{d\boldsymbol{C}(t)}{dt}\boldsymbol{C}(t) + \boldsymbol{C}(t)\frac{d\boldsymbol{C}(t)}{dt} + 2t\boldsymbol{X}^\top\boldsymbol{X} = \boldsymbol{0}. \end{aligned}$$

When $t = 0$, $\boldsymbol{C}(0) = \boldsymbol{I}$ and hence $\left.\frac{d\boldsymbol{C}(t)}{dt}\right|_{t=0} = \boldsymbol{0}$. Hence we can get $\left.\frac{d\varphi_{\boldsymbol{Q}}^{-1}(t\boldsymbol{X})}{dt}\right|_{t=0} = \boldsymbol{X}$.

Now we check the condition that $\varphi_{\boldsymbol{Q}}^{-1}(\varphi_{\boldsymbol{Q}}(\boldsymbol{W})) = \boldsymbol{W}$. Here we may assume $\boldsymbol{W}^\top\boldsymbol{Q}$ is symmetric. Suppose not, compute the SVD of $\boldsymbol{W}^\top\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$ and replace $\boldsymbol{W}$ by $\boldsymbol{W}\boldsymbol{U}\boldsymbol{V}^\top$. Let $\boldsymbol{X} = \varphi_{\boldsymbol{Q}}(\boldsymbol{W}) = (\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c$. Hence we get the equation

$$\boldsymbol{X}^\top\boldsymbol{X} = (\boldsymbol{W}^\top\boldsymbol{Q})^2 - (\boldsymbol{W}^\top\boldsymbol{Q})^4.$$

doi:10.6342/NTU201702973

Solve for $(\boldsymbol{W}^\top \boldsymbol{Q})$ and get $\boldsymbol{W}^\top \boldsymbol{Q} = \boldsymbol{C}$. Hence we can directly compute

$$\begin{aligned}\varphi_{\boldsymbol{Q}}^{-1}(\varphi_{\boldsymbol{Q}}(\boldsymbol{W})) &= \boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{X}\boldsymbol{C}^{-1} \\ &= \boldsymbol{Q}_c\boldsymbol{C} + (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}\boldsymbol{C}^{-1} \\ &= \boldsymbol{Q}_c\boldsymbol{C} + \boldsymbol{W}\boldsymbol{C}\boldsymbol{C}^{-1} - \boldsymbol{C}^2\boldsymbol{C}^{-1} = \boldsymbol{W}\end{aligned}.$$

Note that in the definition of integrated subspace, it is defined with orthogonal invariance of $\boldsymbol{Q}_{[i]}$. Hence it is reasonable to assume $\boldsymbol{W}$ can be transformed by $\boldsymbol{U}\boldsymbol{V}^\top$ and satisfy the condition that $\boldsymbol{W}^\top\boldsymbol{Q}$ is symmetric.

An important issue in Algorithm 5 is the well definite of the matrix $\boldsymbol{C}$. We need to check that $(\frac{\boldsymbol{I}}{4} - \boldsymbol{X}^\top\boldsymbol{X})$ is semi positive definite for $\boldsymbol{X} = (\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}_c$, and hence the square root of matrix can be defined (by SVD). It is equivalent to check $\left\|\boldsymbol{X}^\top\boldsymbol{X}\right\|_2 \leq \frac{1}{4}$. It is also equivalent to check $\|\boldsymbol{X}\|_2 \leq \frac{1}{2}$ by the relation $\left\|\boldsymbol{X}^\top\boldsymbol{X}\right\|_2 = \|\boldsymbol{X}\|_2^2$. By triangle inequality, we have

$$\|\boldsymbol{X}\|_2 \leq \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{Q}_{[i]}\boldsymbol{Q}_{[i]}^\top\boldsymbol{Q}_c.$$

Now we check that for every $\boldsymbol{W} \in \mathcal{S}_{m,\ell}$, the inequality $\left\|\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c\right\|_2 \leq \frac{1}{2}$ holds. It is equivalent to prove $\left\|(\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c)^\top(\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c)\right\|_2 \leq \frac{1}{4}$. Suppose the SVD of $\boldsymbol{Q}_c^\top\boldsymbol{W} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}$. Then by directly computation

$$\begin{aligned}&\left\|(\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c)^\top(\boldsymbol{I} - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\boldsymbol{W}\boldsymbol{W}^\top\boldsymbol{Q}_c)\right\|_2 \\ =& \left\|\boldsymbol{U}\boldsymbol{S}^2\boldsymbol{U} - \boldsymbol{U}\boldsymbol{S}^4\boldsymbol{U}\right\|_2 \\ =& \left\|\boldsymbol{S}^2 - \boldsymbol{S}^4\right\|_2.\end{aligned}$$

Note that $\boldsymbol{S}$ is diagonal matrix, and hence $(\boldsymbol{S}^2 - \boldsymbol{S}^4)$ is also diagonal matrix. For any real number x, we have the inequality $(x^2 - x^4) = -(x^2 - \frac{1}{2})^2 + \frac{1}{4} \leq \frac{1}{4}$. This equation shows that all diagonal entries of $(\boldsymbol{S}^2 - \boldsymbol{S}^4)$ is no greater than $\frac{1}{4}$. This result leads to $\|\boldsymbol{S}^2 - \boldsymbol{S}^4\|_2 \leq \frac{1}{4}$. The check is completed by tracing back the statements.

The convergence of Algorithm 5 is given as the following theorem. For the detailed

doi:10.6342/NTU201702973

proof, please refer to [4].

Theorem 4.2.1. *There exists an $\varepsilon > 0$ such that KN-average algorithm converges, provided that the iteration starts from an initial $\boldsymbol{Q}_{\text{ini}} \in \mathcal{N}_\varepsilon(\boldsymbol{Q}_*\boldsymbol{R}_0)$, where $\boldsymbol{R}_0$ is an arbitrary orthogonal matrix.*

The Algorithm 5 can be also seen as a kind of gradient ascent method due to the selection of lifting map. Therefore, the computational complexity of this algorithm is similar to Algorithm 4. The dominant term of computational complexity is $O(I_{KN}Nm\ell^2)$ if the algorithm needs I_{KN} iteration to converge. The dominant term is also comes from the computation of the projected gradient $\boldsymbol{X} = (\boldsymbol{I}_m - \boldsymbol{Q}_c\boldsymbol{Q}_c^\top)\overline{\boldsymbol{P}}\boldsymbol{Q}_c$, which is the average of lifting map for all sample subspaces $\boldsymbol{Q}_{[i]}$.

4.3 Reduction-Type Average

The main idea for this method is grouping the sample subspaces $\boldsymbol{Q}_{[i]}$ into several group, computing the integrated subspace of each groups, and then integrating these integrated subspaces. For example, suppose we have 4 sample subspaces need to be integrated and we choose the grouping number as 2. This method computes the leading ℓ left singular vectors of $[\boldsymbol{Q}_{[1]}\ \boldsymbol{Q}_{[2]}]$ as $\boldsymbol{Q}_{[1,2]}$, which is the integrated subspaces of $\boldsymbol{Q}_{[1]}$ and $\boldsymbol{Q}_{[2]}$. Next this method computes the leading ℓ left singular vectors of $[\boldsymbol{Q}_{[3]}\ \boldsymbol{Q}_{[4]}]$ as $\boldsymbol{Q}_{[3,4]}$. Then this method computes the average $\overline{\boldsymbol{Q}}$ as leading ℓ left singular vectors of $[\boldsymbol{Q}_{[1,2]}\ \boldsymbol{Q}_{[3,4]}]$. If the number of sample subspaces is more than 4, we can do it hierarchically.

The concept of reduction can be extended to any grouping number. However, we only consider the case that the grouping number is 2. The advantage of this case is that the leading left singular vectors of $\boldsymbol{M} = [\boldsymbol{Q}_1\ \boldsymbol{Q}_2]$ can be written explicitly by the SVD of a small matrix $\boldsymbol{Q}_1^\top\boldsymbol{Q}_2 = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$. Suppose the SVD of $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{\Sigma}\boldsymbol{R}^\top$ and the SVD of $\boldsymbol{Q}_1^\top\boldsymbol{Q}_2 = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$. The leading left singular vectors $\boldsymbol{L}_\ell$ can be obtained via the leading right singular vectors $\boldsymbol{R}_\ell$ and leading singular values $\boldsymbol{\Sigma}_\ell$ as

$$\boldsymbol{L}_\ell = \boldsymbol{M}\boldsymbol{R}_\ell\boldsymbol{\Sigma}_\ell.$$

doi:10.6342/NTU201702973

Now, the problem is how to find the matrix $\boldsymbol{R}_\ell$ and $\boldsymbol{\Sigma}_\ell$. These two matrices can be obtained by the eigenvalue decomposition of $\boldsymbol{M}^\top\boldsymbol{M} = \boldsymbol{R}\boldsymbol{\Sigma}^2\boldsymbol{R}^\top$. The singular value (eigenvalue) decomposition of $\boldsymbol{M}^\top\boldsymbol{M}$ can be write down explicitly by the SVD of $\boldsymbol{Q}_1^\top\boldsymbol{Q}_2 = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$ as

$$\begin{aligned}
\boldsymbol{M}^\top\boldsymbol{M} &= \begin{bmatrix} \boldsymbol{I}_\ell & \boldsymbol{Q}_1^\top\boldsymbol{Q}_2 \\ \boldsymbol{Q}_2^\top\boldsymbol{Q}_1 & \boldsymbol{I}_\ell \end{bmatrix} \\
&= \begin{bmatrix} \boldsymbol{I}_\ell & \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top \\ \boldsymbol{V}\boldsymbol{S}\boldsymbol{U}^\top & \boldsymbol{I}_\ell \end{bmatrix} \\
&= \left(\frac{1}{\sqrt{2}}\begin{bmatrix} \boldsymbol{U} & \boldsymbol{U} \\ \boldsymbol{V} & -\boldsymbol{V} \end{bmatrix}\right)\begin{bmatrix} \boldsymbol{I}+\boldsymbol{S} & \\ & \boldsymbol{I}-\boldsymbol{S} \end{bmatrix}\left(\frac{1}{\sqrt{2}}\begin{bmatrix} \boldsymbol{U} & \boldsymbol{U} \\ \boldsymbol{V} & -\boldsymbol{V} \end{bmatrix}\right)^\top.
\end{aligned}$$

Hence $\boldsymbol{\Sigma}_\ell = \boldsymbol{I}+\boldsymbol{S}$, $\boldsymbol{R}_\ell = \frac{1}{\sqrt{2}}\begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{V} \end{bmatrix}$ and

$$\boldsymbol{L}_\ell = \boldsymbol{M}\frac{1}{\sqrt{2}}\begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{V} \end{bmatrix}(\boldsymbol{I}+\boldsymbol{S}) = (\boldsymbol{Q}_1\boldsymbol{U}+\boldsymbol{Q}_2\boldsymbol{V})(2(\boldsymbol{I}+\boldsymbol{S}))^{-\frac{1}{2}}.$$

Combine the above derivation and the hierarchical structure of the reduction, we can write down the algorithm as Algorithm 6.

Algorithm 6 Reduction

Require: The orthogonal matrices to be integrated $\boldsymbol{Q}_{[1]}, \boldsymbol{Q}_{[2]}, \ldots, \boldsymbol{Q}_{[N]}$.
Ensure: The average $\overline{\boldsymbol{Q}}$.

```
Set n = N.
while n > 1 do
    Set m = ⌊n/2⌋
    for i = 1, 2, ..., m do
        Find SVD of Q_[i]^⊤ Q_[i+m] as U S V^⊤.
        Q_[i] ← (Q_[i] U + Q_[i+m] V)(2(I + S))^{-1/2}.
    end for
    n ← ⌈n/2⌉
end while
Q̄ = Q_[1].
```


doi:10.6342/NTU201702973

Since we have the explicit form of the integration of two subspaces, the computational cost of reduction becomes very cheap. The computation of integration of each pair of subspaces only needs matrix multiplication for $m \times \ell$ and $\ell \times \ell$ matrices. The computational complexity of this part is $O(m\ell^2)$. Also, only an $\ell \times \ell$ SVD needs to be computed for each pair. The computational complexity of this part is $O(\ell^3)$. For each pair, the computational complexity is $O(m\ell^2 + \ell^3)$, which is dominated by $O(m\ell^2)$. Due to the hierarchical structure of reduction, there are only $(N-1)$ pairs need to be applied these processes. Therefore, the computational complexity of reduction is $O(Nm\ell^2)$, which is lower than the line search methods and canonical SVD.

The intuition of this algorithm is that if each group can get a good integrated subspace, the final integrated subspaces is also a good subspace. Since $\boldsymbol{Q}_{[i]}$ are sketched from a same matrix $\boldsymbol{A}$, the last statement should be correct in some sense. However, there is no related theory so far. In theory, the leading ℓ left singular vectors of $[\boldsymbol{Q}_{[1,2]}\ \boldsymbol{Q}_{[3,4]}]$ is in general not the integrated subspace defined in (2.1), which is the left singular vectors of $[\boldsymbol{Q}_{[1]}\ \boldsymbol{Q}_{[2]}\ \boldsymbol{Q}_{[3]}\ \boldsymbol{Q}_{[4]}]$. Although this fact, reduction still gives a roughly integrated subspace in our numerical experiment. Also, in the experiment, we try to use the result from reduction as the initial value of the line search algorithm.

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Chapter 5

Comparison of rSVD and iSVD

Some comparison of rSVD and iSVD with same sketching number is shown in this chapter. In the beginning, we shall explain the meaning of rSVD and iSVD with same sketching number. Suppose in iSVD, we generate N sample subspace with rank ℓ. These numbers mean the sample subspace $\boldsymbol{Q}_{[i]}$ is an $m \times \ell$ orthogonal matrix for $i = 1, 2, \ldots, N$. To generate these subspaces, we need to compute $\boldsymbol{Y}_{[i]} = \boldsymbol{A}\boldsymbol{\Omega}_{[i]}$ for all i. In the block matrix form

$$\left[\boldsymbol{Y}_{[1]}|\boldsymbol{Y}_{[2]}|\cdots|\boldsymbol{Y}_{[N]}\right] = \boldsymbol{A}\left[\boldsymbol{\Omega}_{[1]}|\boldsymbol{\Omega}_{[2]}|\cdots|\boldsymbol{\Omega}_{[N]}\right].$$

Now we change the viewpoint from iSVD to rSVD. Suppose the random matrix for sketching is $\boldsymbol{\Omega} = \left[\boldsymbol{\Omega}_{[1]}|\boldsymbol{\Omega}_{[2]}|\cdots|\boldsymbol{\Omega}_{[N]}\right]$ which is an $n \times N\ell$ matrix. Then the sketched matrix is $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{\Omega} = \left[\boldsymbol{Y}_{[1]}|\boldsymbol{Y}_{[2]}|\cdots|\boldsymbol{Y}_{[N]}\right]$ which is same as the case in iSVD. This derivation shows that using N sample subspaces with rank k in iSVD is equivalent to using $N\ell$ sketches in rSVD. Actually, both of them need to compute the same number of sketches. This is the meaning of rSVD and iSVD with same sketching number.

In common case, rSVD finds the orthogonal matrix of $\boldsymbol{Y}$ as $\boldsymbol{Q}$ and compute the SVD of $\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{A}$ as an approximation of the low-rank SVD. However, one can also find the leading singular vectors of $\boldsymbol{Y}$ as $\boldsymbol{Q}$ instead of using all the information from $\boldsymbol{Y}$. This can reduce the computational cost for computing the SVD of $\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{A}$. In the following discussion and the numerical experiment, this technique is included when we write the term 'iSVD and rSVD with same sketching number.'

doi:10.6342/NTU201702973

Now we can compare iSVD and rSVD with same sketching number. For rSVD, after sketching, we need to find the leading singular vectors of

$$\left[\boldsymbol{Y}_{[1]}|\boldsymbol{Y}_{[2]}|\cdots|\boldsymbol{Y}_{[N]}\right]$$

as $\boldsymbol{Q}$, and then computing the SVD of $\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{A}$. In iSVD, after sketching, we need to orthogonalize $\boldsymbol{Y}_{[i]}$ into $\boldsymbol{Q}_{[i]}$, and then find the integrated subspace $\overline{\boldsymbol{Q}}$. As mentioned in the previous chapter, we need to find the leading singular vectors of the matrix

$$\left[\boldsymbol{Q}_{[1]}|\boldsymbol{Q}_{[2]}|\cdots|\boldsymbol{Q}_{[N]}\right]$$

as $\overline{\boldsymbol{Q}}$, and then computing the SVD of $\overline{\boldsymbol{Q}\boldsymbol{Q}}^\top\boldsymbol{A}$. Therefore, the difference between iSVD and rSVD is finding the different subspace to approximate the original matrix $\boldsymbol{A}$.

Intuitively, the approximation by rSVD may be accurate than iSVD since the block matrix in iSVD is the orthogonalization of each block in rSVD, which may loose the information of length. The numerical result shows that rSVD with same sketching number is slightly better than iSVD for accuracy. However, in theory, iSVD still get the exact low-rank approximation of $\boldsymbol{A}$ if the number of sample subspace N goes large.

Due to the similarity between rSVD and iSVD, we can explain rSVD in the viewpoint of integration same as iSVD. Similar to the averaging concept as Theorem 3.2.1, the corresponding theorem for rSVD can be described as the following theorem.

Theorem 5.0.1. *Let the SVD of $\boldsymbol{A}$ be $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$. Let $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{\Omega}$, where $\boldsymbol{\Omega}$ is randomly generated by i.i.d. standard normal entries. Then*

$$E\left[\boldsymbol{Y}_{[i]}\boldsymbol{Y}_{[i]}^\top\right] = \boldsymbol{U}\boldsymbol{\Sigma}^2\boldsymbol{U}^\top.$$

doi:10.6342/NTU201702973

Proof. By direct computation,

$$\begin{aligned} E\left[\boldsymbol{Y}_{[i]}\boldsymbol{Y}_{[i]}^{\top}\right] &= E\left[\boldsymbol{A}\boldsymbol{\Omega}_{[i]}\boldsymbol{\Omega}_{[i]}^{\top}\boldsymbol{A}^{\top}\right] \\ &= \boldsymbol{A}E\left[\boldsymbol{\Omega}_{[i]}\boldsymbol{\Omega}_{[i]}^{\top}\right]\boldsymbol{A}^{\top} \\ &= \boldsymbol{A}\boldsymbol{I}_n\boldsymbol{A}^{\top} = \boldsymbol{U}\boldsymbol{\Sigma}^2\boldsymbol{U}^{\top} \end{aligned}$$

where $E\left[\boldsymbol{\Omega}_{[i]}\boldsymbol{\Omega}_{[i]}^{\top}\right] = \boldsymbol{I}_n$ is derived directly from the condition that $\boldsymbol{\Omega}$ is randomly generated by i.i.d. standard normal entries. □

The difference between the theorem for iSVD and rSVD is the spectrum matrix of the expected value. For rSVD, the spectrum matrix is $\boldsymbol{\Sigma}^2$, which is directly related to the spectrum of $\boldsymbol{A}$. For iSVD, the spectrum matrix is $\boldsymbol{\Lambda}$, which we can just describe some properties in the theorem.

For computing the leading singular vectors in rSVD, one can also use Algorithm 4 (WY) by replacing the matrix $\overline{\boldsymbol{P}}$ as

$$\overline{\boldsymbol{P}} = \frac{1}{N}\sum_{i=1}^{N}\boldsymbol{Y}_{[i]}\boldsymbol{Y}_{[i]}^{\top}.$$

However, Algorithm 5 (KN) can not be applied directly since the well definite of the retraction map will use the orthogonal property of $\boldsymbol{Q}_{[i]}$. The idea of reduction can be applied to rSVD. However, there is no explicit form for two subspace $\boldsymbol{Y}_{[1]}, \boldsymbol{Y}_{[2]}$. It can not generate the fast algorithm as Algorithm 6.

doi:10.6342/NTU201702973

doi:10.6342/NTU201702973

Chapter 6

Numerical Experiment

In this section, the performance of iSVD will be tested through the numerical experiment. The codes for testing are implemented in MATLAB. The desired rank in all the test is $k = 10$ and the exact sampling rank is $\ell = 22$. If a test only needs to observe the performance without timing, it is run on the machines with the larger size of memory. If a test needs to record the timing result, it is run on the MacBook Pro (Mid. 2014). (Processor: 2.6 GHz Intel Core i5. 2 cores. 4 threads. Memory: 8 GB 1600 MHz DDR3). All the tests follows the same steps as Algorithm 1 (rSVD) or Algorithm 2 (iSVD), but may use the different integration method. Table 6.1 shows the abbreviation and detail information of the integration algorithm used in this section.

The test matrices used in this paper are modified from the test matrix in [10]. These matrices are generated by the form $\boldsymbol{A} = \boldsymbol{H}_m \boldsymbol{\Sigma} \boldsymbol{H}_n^\top$, where $\boldsymbol{H}_m$ denotes the $m \times m$ Hadamard matrix (a kind of orthogonal matrix with all the entries are 1 or -1), $\boldsymbol{H}_n$ denotes the $n \times n$ Hadamard matrix, $m = 2^d$, $n = 2m = 2^{d+1}$, and $\boldsymbol{\Sigma}$ is an $m \times n$ diagonal matrix. In this

svds	The MATLAB built-in command svds
WY	Algorithm 4 with parameter $\beta = 0.5$, $\rho = 10^{-4}$, $\eta = 0.85$. Convergent condition: $\|\boldsymbol{D}_F(\boldsymbol{Q}_c)\|_2 < 10^{-3}$ (iSVD) Convergent condition: $\|\boldsymbol{D}_F(\boldsymbol{Q}_c)\|_2 < 10^{-3}\,\mathrm{tr}(\boldsymbol{Y}_{[1]}^\top \boldsymbol{Y}_{[1]})/\ell$ (rSVD) Initial value $\boldsymbol{Q}_{\text{ini}} = \boldsymbol{Q}_{[1]}$
KN	Algorithm 5. Convergent condition: $\|\boldsymbol{I} - \boldsymbol{C}\|_F < 10^{-5}$ Initial value $\boldsymbol{Q}_{\text{ini}} = \boldsymbol{Q}_{[1]}$
red.	Reduction, Algorithm 6

Table 6.1: Abbreviation and detail information of the algorithm used in this section.

doi:10.6342/NTU201702973

thesis, $\mathbf{\Sigma}$ is given by setting its diagonal entries as

$$\sigma_{i,i} = \begin{cases} s^{\frac{i-1}{k}} & \text{if } i \leq k \\ \dfrac{s(m-i)}{m-k-1} & \text{otherwise} \end{cases}$$

for $s = 10^{-1}, 10^{-3}$ and $k = 10$. We use $\boldsymbol{A}_H(10^{-1})$ and $\boldsymbol{A}_H(10^{-3})$ to denote the matrix generated with the case $s = 10^{-1}$ and $s = 10^{-3}$ respectively.

The similarity between each column of approximate leading singular vectors $\widehat{\boldsymbol{U}}$ and exact leading singular vectors $\boldsymbol{U}_k$ are used to measure the accuracy of the approximate leading left singular vectors $\widehat{\boldsymbol{U}}$ from each test with different methods. The arranged left singular vectors $\boldsymbol{U}$ of the test matrix is exactly the Hadamard matrix $\boldsymbol{H}_m$ due to the construction of test matrix. Therefore, we compute the similarity between i-th column by computing the absolute value of inner product between the i-th column of $\boldsymbol{U}_k$ and the i-th column of $\widehat{\boldsymbol{U}}$. The more inner product close to 1, the better the approximation.

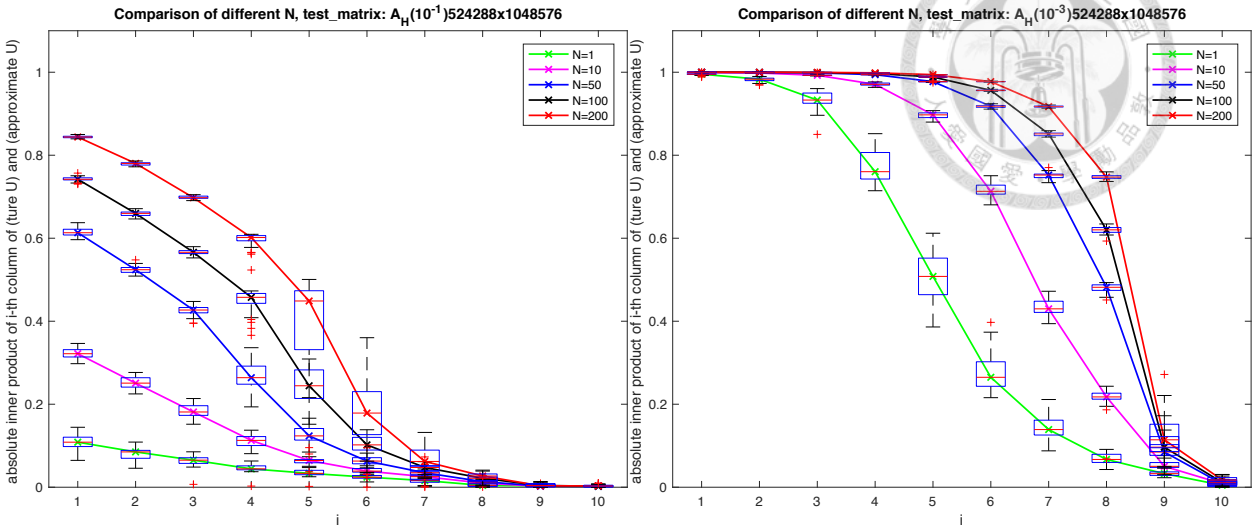
6.1 Different Number of Sketched Subspaces

The purpose of this test is to compare the result for the different choice of the number of sketched subspaces N with the different test matrix $\boldsymbol{A}_H(10^{-1})$ and $\boldsymbol{A}_H(10^{-3})$.

Figure 6.1 shows the similarity for different N with test matrix $\boldsymbol{A}_H(10^{-1})$ and $\boldsymbol{A}_H(10^{-3})$. The accuracy increases as N increasing, which is coherent to the explanation of 3.2.1 in Chapter 3. For $\boldsymbol{A}_H(10^{-3})$, as N increasing, the boxes in box plot become shorter. This provides that the variance of the similarity due to the randomness in sketching step is reduced as the number of sampling subspaces goes large. For $\boldsymbol{A}_H(10^{-1})$, the decreasing of variance is not same as $\boldsymbol{A}_H(10^{-3})$. The reason may be the quality for single subspaces ($N = 1$) is not good enough. Hence it is hard to capture the singular vectors after the fifth one. However, the accuracy still improves as N increasing, which is still a good result and coherent to the explanation in Chapter 3.

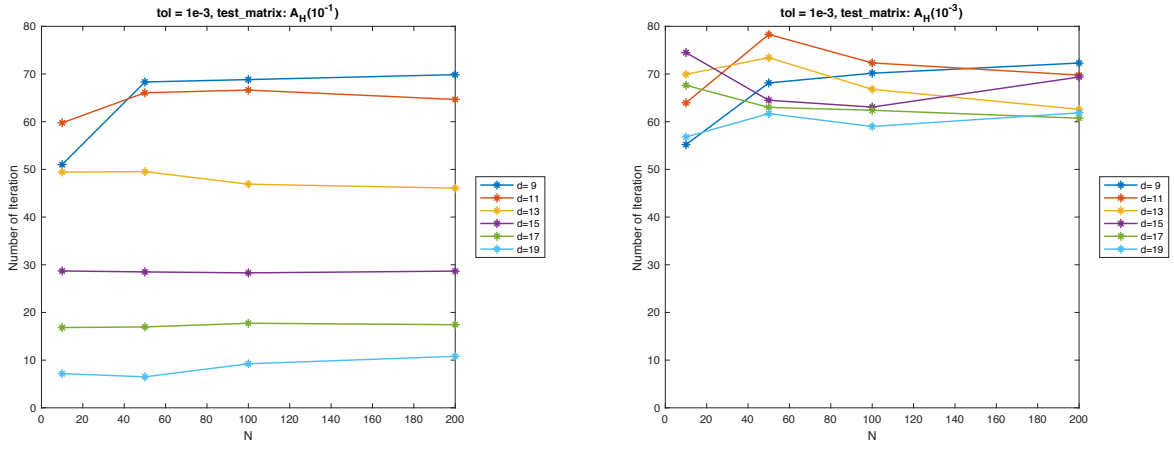
Next, we study the relation of N and the convergent iteration number of WY. Figure 6.2 shows that no matter the size of the matrix, the number of iteration for convergence

doi:10.6342/NTU201702973



(a) Test matrix: $\boldsymbol{A}_H(10^{-1})$ (b) Test matrix: $\boldsymbol{A}_H(10^{-3})$

Figure 6.1: Similarity for different N. The size of test matrix is $m = 2^{19}$, $n = 2^{20}$. For each cases, we repeat 30 times iSVD with integration method WY and plot out the box plot of similarity. The box plot represent the maximum, Q3, median, Q1, minimum for each inner product among 30 times.



(a) Test matrix: $\boldsymbol{A}_H(10^{-1})$ (b) Test matrix: $\boldsymbol{A}_H(10^{-3})$

Figure 6.2: Average iteration number to converge for different N and different size of test matrix. The size of test matrix is $m = 2^d$, $n = 2^{d+1}$ for $d = 9, 11, 13, 15, 17, 19$. Each point shows the average iteration number among 30 tests.

doi:10.6342/NTU201702973

has no clear relation to the number of sampling subspaces N. This result is not surprising since the matrix $\overline{\boldsymbol{P}}$ tends to be a fixed matrix (expected value in Theorem 3.2.1) as N goes large. This is a good news since the computational complexity of WY is $O(I_{WY}Nm\ell^2)$ and the computational complexity of canonical SVD is $O(N^2m\ell^2)$. This figure shows I_{WY} do not increase as the number of sketched subspaces N increase. Hence using WY for integration is better in computational complexity than using canonical SVD.

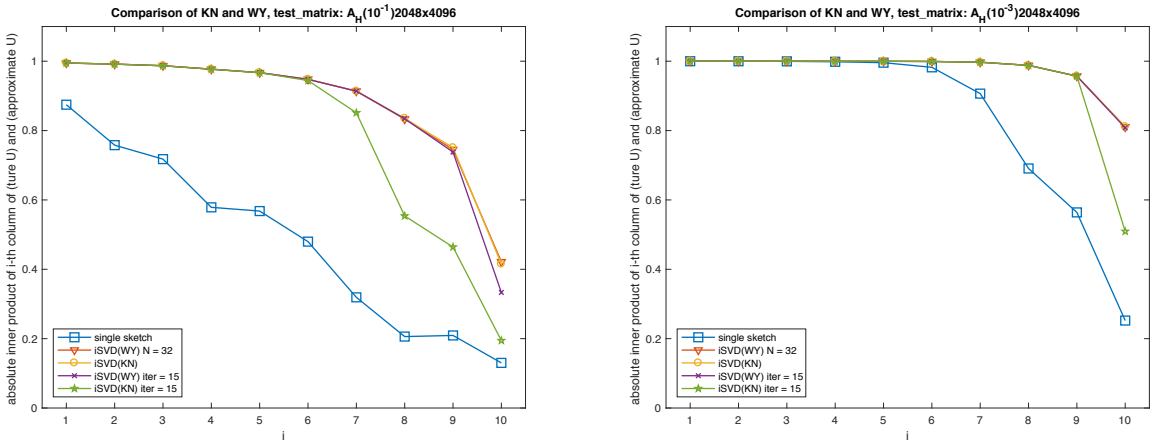
6.2 Comparison of KN and WY

The purpose of this test is to compare the results of iSVD by using WY and KN. WY is derived from the viewpoint of line search and KN is derived from the viewpoint of average on Stiefel manifold. Although they are derived from different viewpoints, both of them contain the idea of gradient ascent. Therefore, it is interesting to observe the difference between these two algorithms.

Figure 6.3 shows the accuracy for the approximate singular vectors by using WY and KN. Both WY and KN can capture the approximate singular vectors with the same accuracy when they both converge. This result is coherent to the convergent theory in Section 4 that both of WY and KN converge to the same $\overline{\boldsymbol{Q}}$, and hence generate the same $\widetilde{\boldsymbol{U}}$. However, KN needs more iteration to converge. To eliminate the effect of different convergence criteria, tow more results for the accuracy for WY and KN with same iteration number are added to the same figure. The approximation from WY is more accurate than KN when the iteration number is fixed as 15. This difference is not surprising since WY use BB step size to accelerate the convergence while KN does not use any technique to accelerate the convergence.

We do more test on the iteration number for WY and KN. Figure 6.4 and Figure 6.5 show the accuracy from WY and KN respectively, with the iteration number $5, 10, 15$ and 20. As these tow figure shown, both WY and KN get a better accurate result as the iteration number goes large. WY converge faster than KN. For the test matrix $\boldsymbol{A}_H(10^{-1})$, WY can get almost same accurate as the converged result with only 20 iteration. For the test matrix $\boldsymbol{A}_H(10^{-3})$, WY only uses 15 iteration to get the almost same accurate result. In contrast,

doi:10.6342/NTU201702973

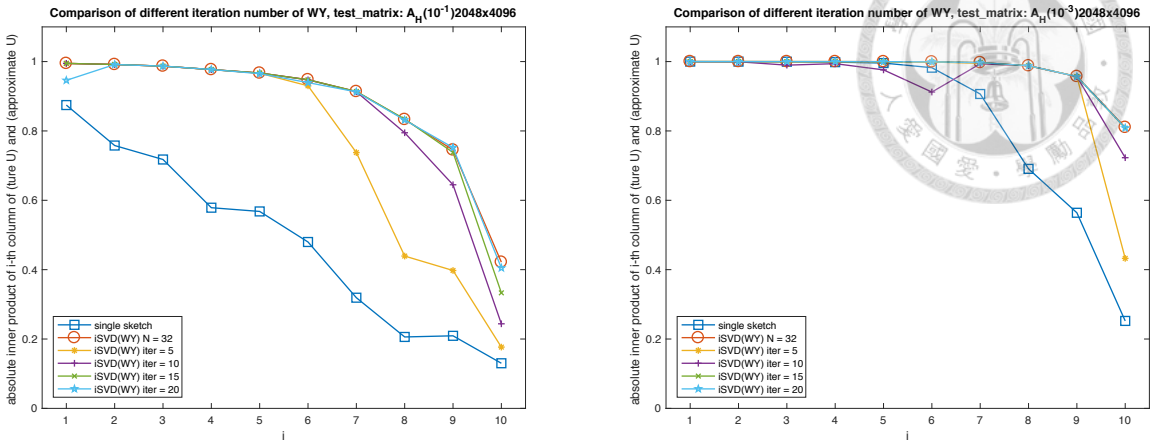


(a) Test matrix: $\boldsymbol{A}_H(10^{-1})$ (b) Test matrix: $\boldsymbol{A}_H(10^{-3})$

time(sec) iteration number	iSVD(WY)	iSVD(KN)	iSVD(WY, fix iter)	iSVD(KN, fix iter)
$\boldsymbol{A}_H(10^{-1})$	0.964061 61	1.634885 242	0.266471 15	0.113637 15
$\boldsymbol{A}_H(10^{-3})$	1.363975 84	1.538313 218	0.243105 15	0.142008 15

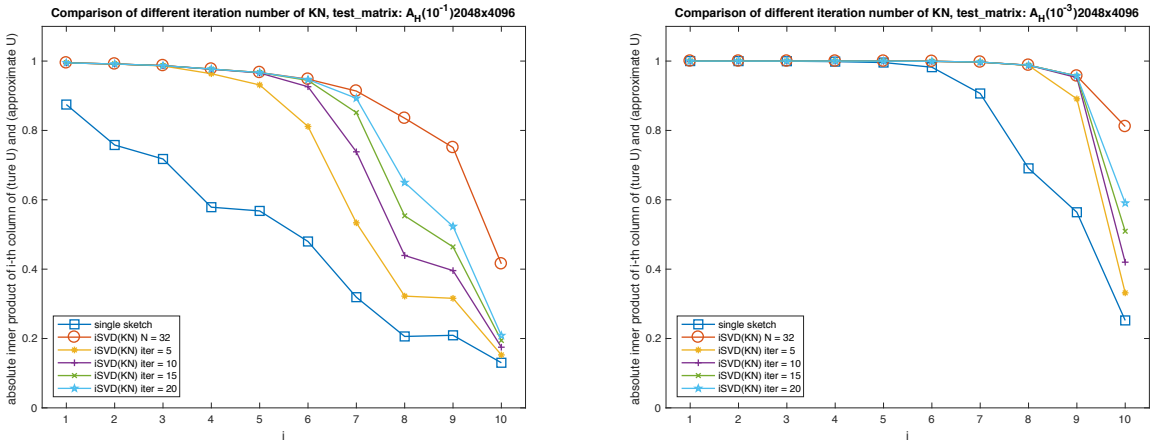
Figure 6.3: Comparison of the approximate singular vectors by using WY and KN. The size of test matrix is $m = 2^{11}$, $n = 2^{12}$ and the sampling number $N = 32$. All of these test use the same 32 sketched subspaces $\boldsymbol{Q}_{[i]}$. The first line in the legend represents the similarity of $\overline{\boldsymbol{Q}} = \boldsymbol{Q}_{[1]}$. The second and third lines are the result from WY and KN respectively. The forth and fifth lines are from WY and KN respectively with fixed iteration number 15.

doi:10.6342/NTU201702973



(a) Test matrix: $\boldsymbol{A}_H(10^{-1})$ (b) Test matrix: $\boldsymbol{A}_H(10^{-3})$

Figure 6.4: WY with different iteration numbers. The sketched subspaces $\boldsymbol{Q}_{[i]}$ are same in Figure 6.3. The first line in the legend represents the similarity for the case $\overline{\boldsymbol{Q}} = \boldsymbol{Q}_{[1]}$. The second line is the similarity from WY (with 61 iteration for $\boldsymbol{A}_H(10^{-1})$ and 84 iteration for $\boldsymbol{A}_H(10^{-3})$ to converge). The third to sixth lines are from WY with iteration number $5, 10, 15, 20$ respectively.



(a) Test matrix: $\boldsymbol{A}_H(10^{-1})$ (b) Test matrix: $\boldsymbol{A}_H(10^{-3})$

Figure 6.5: KN with different iteration numbers. The sketched subspaces $\boldsymbol{Q}_{[i]}$ are same in Figure 6.3. The first line in the legend represents the similarity for the case $\boldsymbol{Q}_{[1]}$. The second line is the similarity from KN. The third to sixth lines are from KN with iteration number $5, 10, 15, 20$ respectively.

doi:10.6342/NTU201702973

KN does not get the same accurate result in both cases by using 20 iteration.

However, the convergence of KN is more 'smooth' than WY. For the test matrix $\boldsymbol{A}_H(10^{-1})$, the similarity of the first column at 20 iteration is lower than the previous iteration. The same phenomenon also shows up in the sixth column of 10 iteration for the test matrix $\boldsymbol{A}_H(10^{-3})$. This could be a risk if one wants to use early stop technique in iSVD for WY. However, the fast convergence of WY could also be an advantage to use early stop technique.

Also, these two figures point out that the stopping criterion for WY may not be suitable. For the test matrix $\boldsymbol{A}_H(10^{-3})$, WY can get almost same accurate as the converged result with only 20 iteration. However, the stopping criterion is not satisfied until 84 iteration. This phenomenon also happens in the test matrix $\boldsymbol{A}_H(10^{-1})$. The reason why this phenomenon happens may be the unsuitable choice of stopping criteria, since the stopping criteria measure whether $\boldsymbol{Q}_c$ is convergent in WY, not measure whether $\boldsymbol{Q}_c$ is an enough accurate subspace for generating the low-rank approximation of $\boldsymbol{A}$.

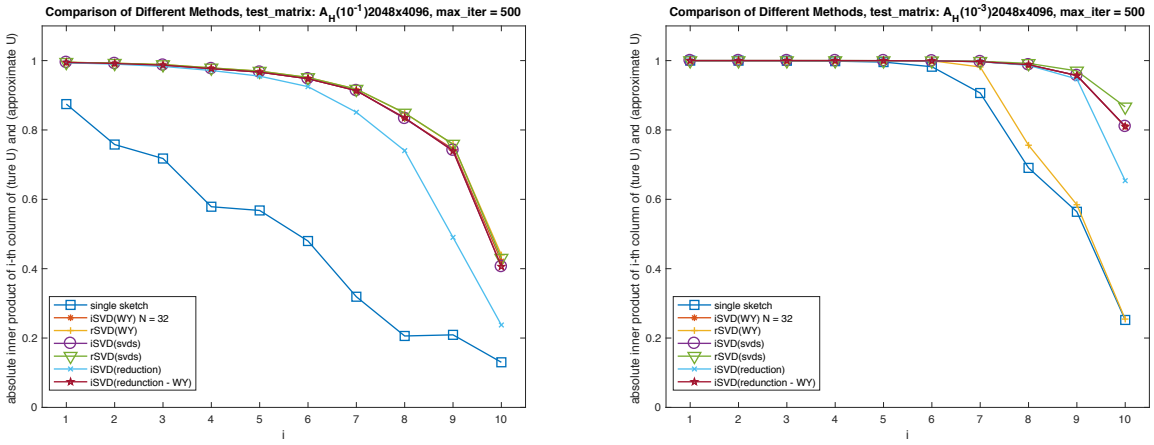
Remark. The timing results show that for a single iteration, WY uses about twice time than KN. However, the computation of complexity shows WY and KN have same computational complexity. This difference could be caused by the small size of the matrix and the lack of optimization for codes implemented WY.

6.3 Comparison of iSVD, rSVD and Reduction

The purpose of this test is to observe the difference between iSVD and rSVD numerically. As mentioned in the Chapter 5, iSVD and rSVD is very similar if the total sketching number is same. Also, we will do the numerical test of reduction in the same time to compare with rSVD more easily.

Figure 6.6 shows the result for iSVD and rSVD with WY and svds. We may treat the result from svds as the ideal result for iSVD and rSVD and compare it with the result from WY. For the test matrix $\boldsymbol{A}_H(10^{-1})$, the accuracy of rSVD is slightly better than iSVD. Also, both rSVD and iSVD with WY can capture the same accurate approximation as svds does. However, the iteration number of rSVD is more than the iteration number of

doi:10.6342/NTU201702973

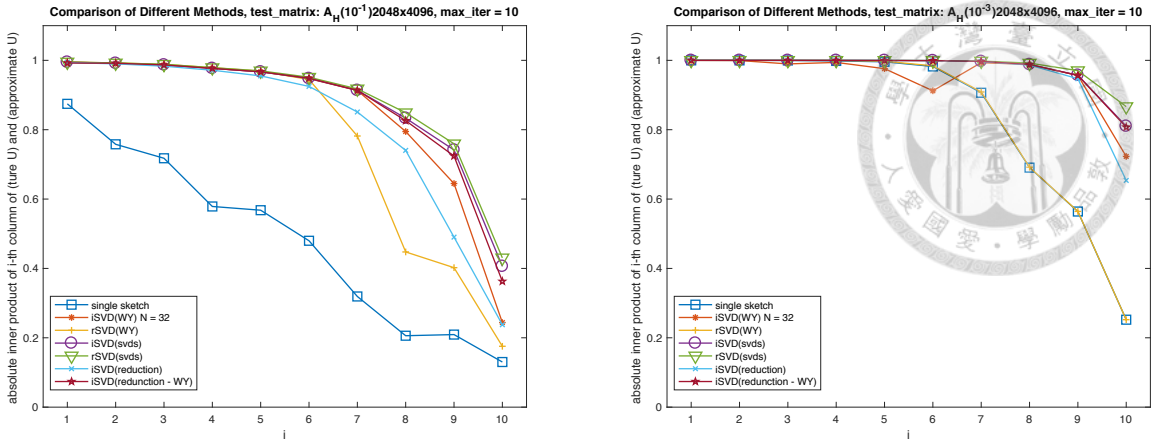


(a) Test matrix: $\boldsymbol{A}_H(10^{-1})$ (b) Test matrix: $\boldsymbol{A}_H(10^{-3})$

time(sec) iteration number	iSVD(WY)	rSVD(WY)	iSVD(svds)	rSVD(svds)	iSVD(red.)	iSVD (red.+WY)
$\boldsymbol{A}_H(10^{-1})$	0.746769 61	1.839638 122	0.775641	0.692867	0.052223	0.901197 66
$\boldsymbol{A}_H(10^{-3})$	1.057405 84	0.865924 63	0.640089	0.597390	0.039458	1.017564 79

Figure 6.6: Similarity for WY with iSVD and rSVD, reduction, and svds. The number of sketched subspaces in iSVD is $N = 32$. The number of sketching in rSVD is $32 * 22$, which is same as the total number of sketching in iSVD. The algorithm red.+WY uses the result of reduction as the initial value of WY.

doi:10.6342/NTU201702973



(a) Test matrix: $\boldsymbol{A}_H(10^{-1})$ (b) Test matrix: $\boldsymbol{A}_H(10^{-3})$

time(sec) iteration number	iSVD(WY)	rSVD(WY)	iSVD(svds)	rSVD(svds)	iSVD(red.)	iSVD (red.+WY)
$\boldsymbol{A}_H(10^{-1})$	0.158318 10	0.153800 10	0.823445	0.694192	0.039138	0.131893 10
$\boldsymbol{A}_H(10^{-3})$	0.149794 10	0.166209 10	0.637057	0.617643	0.039255	0.133753 10

Figure 6.7: Similarity for the methods in 6.6 with the fixed iteration number 10 for WY.

iSVD. For the test matrix $\boldsymbol{A}_H(10^{-3})$, rSVD with WY even fails to find the approximation with the same accuracy as svds. To eliminate the difference of convergent criteria, Figure 6.7 shows the results for the same setting with fixed iteration number 10. As shown in the figure, the accuracy for iSVD with WY is better than the accuracy for rSVD with WY when the iteration number is fixed. Although svds capture slightly better result than WY for iSVD, the computing time of WY is much faster than the svds.

For the reduction part, Figure 6.6 shows that reduction fails to capture the same accurate approximate as WY and svds. However, the computational time only takes about 0.04 second, which is very fast compare to WY and svds. Figure 6.7 shows that the accuracy for reduction is even better than the rSVD when the iteration number is 10.

Also, Figure 6.7 shows that using the result from reduction as the initial value of WY with only 10 iteration gives an almost accurate approximation same as svds does. However, the computational time for reduction+WY is much lower than svds. Although there is no theoretical guarantee for the error bound of the reduction so far, the numerical experiment shows that reduction can get an integrated subspace with better accuracy than

doi:10.6342/NTU201702973

the single sketched subspaces, which is the original initial value for WY. It is not surprising that reduction+WY can get a better result than WY with same iteration number since reduction+WY start from a better initial guess.

doi:10.6342/NTU201702973

Chapter 7

Discussion and Conclusion

In this thesis, some theoretical and performance analysis are shown. The integrated subspace $\overline{\boldsymbol{Q}}$ is defined by the solution of the optimization problem (2.1) or (3.1). The solution of these problems is the leading ℓ singular vectors of the matrix $\overline{\boldsymbol{P}}$. Also, this is the only local maximizer of the problem (3.1).

Line search type algorithm (WY), KN type average (KN), and reduction are introduced to compute the integrated subspace. WY and KN are convergent to the integrated subspace. The leading term of computational complexity for WY and KN is $O(INm\ell^2)$, where I is the iteration number to converge. The computational complexity for canonical SVD for integration is $O(N^2m\ell^2)$. The numerical results show that N is independent of I. Also, WY can get nearly accurate approximation same as the convergent result by only few iteration number. These results provide that WY is an efficient algorithm for integration.

The computational complexity of reduction is $O(Nm\ell^2)$, which is faster than canonical SVD, WY, and KN. In theory, it does not capture the exact integrated subspace defined in (2.1). In the numerical experiment, the reduction can find an approximate integrated subspace. Also, it shows the potential that reduction can be used as a preprocess for WY to speed up the convergent with just a little extra computational cost.

If the total sketching number is same, iSVD is similar to rSVD. The difference between them is that the spectrum of the expected value in Theorem 3.2.1 and 5.0.1. Also, WY can directly apply to rSVD as the method to find the leading singular vectors of the sketching.

doi:10.6342/NTU201702973

In the numerical experiment, WY converges faster for iSVD than rSVD. Sometimes WY also fails to find the leading singular vectors of the sketching in rSVD. However, there is no theoretical proof or explanation so far.

To sum up, iSVD gives an idea to integrate the subspaces generated from random sketching of a matrix. Although some phenomenons shown in numerical experiment lack theoretical explanation and proof, iSVD still shows the potential for approximate low-rank SVD with higher quality than rSVD in the same computing time. As more theoretical results showing up, iSVD could be an option for computing dimension reduction and feature extraction of large scale data faster but still accurate in the future.

doi:10.6342/NTU201702973

Bibliography

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[2] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[3] J. Barzilai and J. M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

[4] T.-L. Chen, D. D. Chang, S.-Y. Huang, H. Chen, C. Lin, and W. Wang. Integrating multiple random sketches for singular value decomposition. *arXiv preprint arXiv:1608.08285*, 2016.

[5] S. Fiori, T. Kaneko, and T. Tanaka. Mixed maps for learning a kolmogoroff-nagumo-type average element on the compact Stiefel manifold. *IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, pages 4518– 4522, 2014.

[6] I. Griva, S. G. Nash, and A. Sofer. *Linear and nonlinear optimization*. Siam, 2009.

[7] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[8] T. Kaneko, S. Fiori, and T. Tanaka. Empirical arithmetic averaging over the compact Stiefel manifold. *IEEE Transations on Signal Processing*, 61(4):883–894, 2013.

doi:10.6342/NTU201702973

[9] J. R. Magnus and H. Neudecker. The commutation matrix: some properties and applications. *The Annals of Statistics*, pages 381–394, 1979.

[10] V. Rokhlin, A. Szlam, and M. Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.

[11] Z. Wen and W. Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1-2):397–434, 2013.

[12] H. Zhang and W. W. Hager. A nonmonotone line search technique and its application to unconstrained optimization. *SIAM Journal on Optimization*, 14(4):1043–1056, 2004.

doi:10.6342/NTU201702973