2007

Numerical Methods for Portfolio Risk Estimation

Jianke Zhang
THE FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

NUMERICAL METHODS FOR PORTFOLIO RISK ESTIMATION

By

JIANKE ZHANG

A Dissertation submitted to the
Department of Mathematics
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Degree Awarded:
Spring Semester, 2007
The members of the Committee approve the Dissertation of Jianke Zhang defended on March 30, 2007.

Alec Kercheval
Professor Directing Dissertation

Fred Huffer
Outside Committee Member

Kyle Gallivan
Committee Member

Paul Beaumont
Committee Member

Warren Nichols
Committee Member

The Office of Graduate Studies has verified and approved the above named committee members.
I would like to gratefully and sincerely thank my advisor, Dr. Alec Kercheval for his guidance, understanding and patience during my graduate studies at Florida State University. His mentorship was paramount to providing a well rounded experience consistent with my long-term career goals. My sincerest thanks also go to Dr. Kyle Gallivan for his continuous support and working closely with me. Without his support and constant guidance this dissertation may not have been completed. I would like to thank to the rest of my dissertation committee, Dr. Warren Nichols, Dr. Fred Huffer and Dr. Paul Beaumont for their insightful comments. I have taken classes from all of them and gained valuable knowledge from those classes.

Dr. Christopher Hunter helped me when I took my qualifying exam, not only in the course he taught but also another course he did not teach! Dr. Eric Klassen also gave me very insightful instruction during my research. Dr. David Kopriva offered help in some algorithms. I have been sitting in several of Dr. Craig Nolder’s classes and I have learned a lot from him. I also want to thank to the Department of Mathematics for continuously supporting me as a teaching assistant. I also would like to thank my friends Andrew Culham, Juan Moreno, Emmanuel Salta, Chris Furner, Zheng Chen and Dr. Tianyu Zhang, for being always there for me throughout my dissertation writing experience. Many thanks also go to Prof. Deqian Pu at Yantai University. He taught me not only the knowledge of mathematics but also his life experience.

Most importantly, I would like to thank my parents for their unconditional faith and love in me and allowing me to be as ambitious as I wanted. It was under their watchful eye that I gained so much drive and an ability to tackle challenges head on.
# TABLE OF CONTENTS

List of Tables .................................................. vi
List of Figures ................................................. vii
Abstract ......................................................... viii

1. INTRODUCTION ................................................. 1

2. Literature Review .............................................. 5
   2.1 Parametrization Method ................................. 5
   2.2 Positive Semi-Definite Programming ............... 5
   2.3 Method of Linear Transformation ................. 6

3. Weighted Orthogonal Procrustes Problem ................. 8
   3.1 Orthogonal Procrustes Problem ..................... 11
   3.2 Polar Decomposition ................................. 12
   3.3 Geometric Analysis .................................. 13
   3.4 A necessary condition ............................. 16

4. Local Minimization Techniques ............................. 23
   4.1 An Algorithm based on geometric projection .... 23
   4.2 An Algorithm Based On Majorization .......... 25
   4.3 Comparison of Numerical Results ................. 32

5. Global Optimization ....................................... 36
   5.1 Global Optimization Review ....................... 36
   5.2 Tunneling Method .................................... 38
   5.3 Newton’s Method ..................................... 41

6. Comparison of New Algorithm and Previous Methods .... 47

7. CODE .......................................................... 53
   7.1 Majorization Algorithm Code ............... 53
   7.2 Pseudo-code of Gradient-flow Code base on Chu and Trendafilov’s Method 55
   7.3 Tunneling Method Code ............................. 57

8. CONCLUSION ................................................. 59
LIST OF TABLES

4.1 Convergence speed (in seconds) of two methods for different size of matrices. 32
4.2 Convergence of two methods for 2000 random staring matrices. 33
4.3 Convergence of two methods for 2000 random staring matrices. 34
4.4 Convergence of two methods for 2000 random starting matrices. 35
6.1 Our model comprises 730 factors in 23 markets. There is one block for each market and plus the currency block. 48
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The tangent and normal spaces of the orthogonal group</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>Parallel transport in the orthogonal group</td>
<td>16</td>
</tr>
<tr>
<td>3.3</td>
<td>A Necessary Condition of Being a Minimizer</td>
<td>17</td>
</tr>
<tr>
<td>4.1</td>
<td>Piecewise linear function for determining the least total absolute deviation</td>
<td>27</td>
</tr>
<tr>
<td>4.1</td>
<td>from the observations ([1, 3, 7, 8, 9, 11, 12])</td>
<td></td>
</tr>
<tr>
<td>5.1</td>
<td>Two alternating phases in the tunneling method for finding the global minimum.</td>
<td>39</td>
</tr>
<tr>
<td>5.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>Singular values of (B) and (C)</td>
<td>49</td>
</tr>
<tr>
<td>6.2</td>
<td>RMSE of cross-block correlations. Results are shown for gluing in a single</td>
<td>51</td>
</tr>
<tr>
<td>6.2</td>
<td>country block.</td>
<td></td>
</tr>
<tr>
<td>6.3</td>
<td>Singular values of (B) (blue cross) and (C) (red dot) from replacing the</td>
<td>52</td>
</tr>
<tr>
<td>6.3</td>
<td>Hong Kong market</td>
<td></td>
</tr>
</tbody>
</table>

vii
In portfolio risk management, a global covariance matrix forecast often needs to be adjusted by changing diagonal blocks corresponding to specific sub-markets. Unless certain constraints are obeyed, this can result in the loss of positive definiteness of the global matrix. Imposing the proper constraints while minimizing the disturbance of off-diagonal blocks leads to a non-convex optimization problem in numerical linear algebra called the Weighted Orthogonal Procrustes Problem. We analyze and compare two local minimizing algorithms and offer an algorithm for global minimization. Our methods are faster and more effective than current numerical methods for covariance matrix revision.

**Key words:** portfolio risk, total risk, optimization, positive definite, weighted orthogonal procrustes problem.
CHAPTER 1

INTRODUCTION

In 1952, Markowitz ([21]) developed a rigorously formulated, operational theory for portfolio selection under uncertainty, which laid the groundwork for Modern Portfolio Theory (MPT). MPT models the return of an asset as a random variable and a portfolio as a weighted combination of assets; the return of a portfolio is thus also a random variable and consequently has an expected value and a variance. Risk in this model is identified with the standard deviation of portfolio return.

There has been a heated debate regarding risk measures used for evaluating and optimizing investor’s risk exposure ([3]). In particular, many risk measures, other than the variance, have been proposed in portfolio theory. Correlation has been criticized as a dependence measure for the following reasons: 1. It can’t capture the non-linear dependency; 2. It is built on the assumption of Gaussian but real financial data is skewed and fat-tailed; and 3. It can only tell us a number but not the dependence structure ([7]).

Alternative risk measures, like semi-variance, VAR, CVAR, mean absolute deviation, coherent risk measures, etc., have been proposed to avoid the problems of correlation and variance theoretically. However, we do not see many empirical results that demonstrate the superiority of these measures in portfolio theory. (However this has been observed that in some other fields such as the extreme value theory in insurance risk.) Giacometti and Lozza ([8]) compared the performance of different risk measures for asset allocation models. Their results indicate that Markowitz’s mean-variance framework performs the best or very close to the best among all measures including CVAR, mean-MiniMax, mean-absolute deviation, mean-asymptotic and mean-Gini approach etc., for either historical data or simulated data with Gaussian or stable non-Gaussian returns. These results suggest that instead of giving up the variance as a risk measure we may be able to improve upon it since only mean-variance
has the elegant and convenient variance-covariance matrix tool available.

Correlations play an important role in Markowitz’s framework. Forecasts of correlation matrices are important for portfolio risk management, but they often suffer from estimation problems such as spurious outliers in the data, non-synchronous data, questionable relevance of past history, and varying levels of confidence bounds. Also, risk managers may want to alter some entries of an estimated correlation matrix to account for information not incorporated in the estimation algorithm itself. The result can be a proposed target correlation matrix that, while visibly symmetric with unit diagonal, is not in fact positive semi-definite (PSD) because of the presence of negative eigenvalues.

A risk manager of a financial institution always wants to know the total risk of multiple sources. This information is important since he needs to know if there is enough capital to cover the total risks of the institution. He also needs this information for asset allocation reasons from the point view of the firm. This formulates the problem: How can the firm evaluate the total risk of the combined portfolios of many managers? ([1])

Assume that we have \( K (> 1) \) managers, each responsible for a portfolio in one of \( K \) markets. We also assume that each of these managers has a factor covariance matrix \( V_k \) of size \( n_k \times n_k \), \( (k = 1, \ldots, K) \) of their own, which successfully describes the risk of portfolios in market \( k \).

To understand the firm-wide total risk of the union of the \( K \) managers’ portfolios, we need a large covariance matrix \( V \) of size \( N \times N \), where \( N = n_1 + \ldots + n_K \). The matrix \( V \) must agree on its diagonal blocks with the \( V_k \)’s, and have meaningful information on the off-diagonal blocks about correlations between factors in different markets. There may be enough historical data to estimate covariance matrices of size \( n_k \times n_k \), but there is almost certain not enough data to estimate an \( N \times N \) covariance matrix directly.

There are various ways to estimate this \( N \times N \) covariance matrix. For example, one could make a factor model for the factors, with individual market factors exposed to a smaller number of global factors. We henceforth assume that the firm can arrive at a (necessarily crude) estimate of the global \( N \times N \) covariance matrix which we call \( \Sigma \).

This matrix \( \Sigma \) reflects greater estimation error than the individual market covariance forecasts, but we suppose it contains the best information about the cross-market covariances, which is important for estimating total risk. We now face the following difficulty: the diagonal blocks of \( \Sigma \) corresponding to the individual markets are probably not equal to the
more refined covariance matrix forecasts ($V_k$’s) already developed by the individual managers. If there is a discrepancy, the global model and an individual market model will disagree about the risk of the portfolios within that market. This means the firm-wide view of risk is at odds with the views of the managers.

To solve this difficulty, a risk manager usually would do the following things([1]):

1. Develop a first-draft $N \times N$ covariance matrix $V$ using the available data, in whatever way seems to best capture cross-market correlations.

2. Replace the diagonal blocks $V_k$ of $V$ with the previously estimated market covariance matrices $V_k$ to form a new matrix $\tilde{V}$.

3. Since this replacement is likely to spoil positive semi-definiteness by creating negative eigenvalues, the off-diagonal blocks of $\tilde{V}$ must then be adjusted in some minimal way to restore positive definiteness.

This becomes the following optimization problem. We have a symmetric positive semi-definite matrix

$$V = \begin{pmatrix} V_1 & C \\ C^T & V_2 \end{pmatrix}$$

which is the first draft of the covariance matrix. Here $V_1$ is a diagonal block corresponding to covariances of the factors in one of the individual markets. We assume that $V_1$ is $n \times n$, $V$ is $N \times N$, and for simplicity we assume $n < N/2$.

Now suppose we have valid reason to believe that $\tilde{V}_1$ is a better estimate for the factor covariance matrix of that market. We wish to substitute $\tilde{V}_1$ for $V_1$ in $V$, without changing the other factors described by $V_2$. To avoid creating negative eigenvalues, this means we need to allow freedom to adjust $C$ to restore positive semi-definiteness. The problem then becomes finding $\tilde{C}$ such that

$$\tilde{V}(\tilde{C}) = \begin{pmatrix} \tilde{V}_1 & \tilde{C} \\ \tilde{C}^T & V_2 \end{pmatrix}$$

is positive semi-definite and as close as possible, in some suitable sense, to $V$. Solving this problem is the objective of this dissertation.

Let us simplify the problem here. In the most basic version of the problem, we are given a positive definite matrix $V$, expressed in block form as

$$V = \begin{pmatrix} A & B \\ B^T & C \end{pmatrix},$$
This is intended to be a first draft of a global correlation matrix. Here $A$ is a small diagonal block corresponding to correlations of the factors in one of the individual markets. We will say that $A$ is $n \times n$, $V$ is $N \times N$, and for simplicity assume $n < N/2$.

Independently, we are also given a better estimate $\tilde{A}$ for the factor correlation matrix of that market. We wish to substitute $\tilde{A}$ for $A$ in $V$, without changing the other factors described by $C$. To avoid creating negative eigenvalues, this means we need to allow freedom to adjust $B$ to restore positive semidefiniteness. The problem then becomes that of finding $\tilde{B}$ such that

$$\tilde{V}(\tilde{B}) = \begin{pmatrix} \tilde{A} & \tilde{B} \\ \tilde{B}^T & \tilde{C} \end{pmatrix}$$

is positive semidefinite and as close as possible to $V$.

This dissertation is organized as follows. In Chapter 2 we review the literature on this problem and reduce it to an orthogonal Procrustes type problem. In Chapter 3 we introduce the orthogonal Procrustes problem and weighted orthogonal Procrustes problem and its geometric properties. In Chapter 4 we discuss two local numerical optimization algorithms and compare them. In Chapter 5 we talk about the global optimization technique and application to weighted orthogonal procrustes problem. Numerical experiments are give in Chapter 6 and codes are listed in Chapter 7. We conclude in Chapter 8.
CHAPTER 2

Literature Review

We describe the following existing algorithms available in the literature for the minimization problem.

2.1 Parametrization Method

Rebonato and Jäckel ([24]) considered the problem of the nearest $n \times n$ correlation matrix to a given symmetric matrix with unit diagonal, and proposed a solution involving minimizing a norm over an $n(n - 1)$ dimensional parameter space while Anderson and Kercheval ([2]) improve the dimension by a factor of 2. In this method, each row vector of the $n \times n$ configuration matrix $X$ is parameterized by spherical coordinates. Subsequently, nonlinear programming algorithms such as Newton-Raphson or conjugate gradient are applied on the parameter or angle space. The parametrization by spherical coordinates implies that the objective function is given in terms of trigonometric sine and cosine functions. In turn, these yield a computational burden when calculating the derivative, which hinders an efficient implementation. For the example we will do later, a $65 \times 65$ matrix needs $\frac{65 \times 64}{2} = 2080$ parameters! This method also doesn’t avoid the problem that the solution will have zero eigenvalues.

2.2 Positive Semi-Definite Programming

An improvement on the technique of Rebonato and Jäckel ([24]) makes use of the concept of positive semi-definite programming, e.g. [13], [20]. Here, one notes that the space $S$ of positive semi-definite matrices is a convex cone in the set of all $n \times n$ matrices, and prescribing diagonal blocks represents a simple linear constraint. Hence the problem becomes a convex optimization problem, which can be solved for quite large $n$. While this approach
is very powerful, there are two difficulties in this context. Of necessity, if $\tilde{V}(B)$ has negative eigenvalues, then the optimum $\tilde{V}(\tilde{B})$ minimizing the norm $\|\tilde{V} - V\|$ will have zero eigenvalues, which is inconvenient for risk management applications. (One could add a further constraint that eigenvalues be larger than a certain chosen lower bound, but this is ad hoc.)

Things could get worse. The solution to the problem posed this way will always represent a change of the underlying variables that mixes factors across markets, which is financially undesirable. We explain this point in the next section.

### 2.3 Method of Linear Transformation

Anderson, Goldberg, Kercheval, Miller and Sorge ([1]) built the following framework. They observed that changing covariance matrices means changing variables.

For convenience, let $M_N$ denote the vector space of $N \times N$ real matrices, $GL(N, \mathbb{R})$ the subset of invertible matrices, and denote by $COV(N)$ the subset of all possible $N \times N$ covariance matrices of some $N$-dimensional random vector. (In our application, the random vector will be the vector of factor returns.) Equivalently, $COV(N)$ is the space of $N \times N$ positive semidefinite (symmetric) matrices. The subset of positive definite matrices will be denoted $COV^+(N)$. The following fact is elementary:

**Proposition 1** If $V \in COV^+(N)$, then

1. \( \{LVLT : L \in M_N\} = COV(N) \), and

2. \( \{LVLT : L \in GL(N, \mathbb{R})\} = COV^+(N) \).

Moreover, if $V$ is the covariance matrix of a random vector $s$, the matrix $LVLT$ is the covariance matrix induced by the linear change of variables $\tilde{s} = Ls$. Therefore we may think of changing the covariance matrix $V$ to a new matrix $\tilde{V}$ as equivalent to making a linear change of variables of the underlying factors $s$. Since factors are determined via linear regression on the asset returns, their identities are somewhat approximate in the first place - financially we can tolerate a small change $L$ close to the identity as a correction in light of the exogenous information in $\tilde{A}$.

However, we need to preserve the identities of the individual markets corresponding to the diagonal blocks of $V$, and furthermore we do not want to touch factors outside the $A$ block when making the above correction.
This means our change of variables should be constrained to the following block diagonal form: 

\[ L = \begin{pmatrix} L_1 & 0 \\ 0 & I \end{pmatrix} \]

It is now not difficult to show the following [1]:

**Proposition 2** With \( V, \tilde{V} \), and \( L \) as above,

\[ LV L^T = \tilde{V} \]

if and only if \( L_1 = \tilde{A}^{1/2}QA^{-1/2} \), where \( Q \) is orthogonal and the exponent \( 1/2 \) refers to the unique positive definite square root.

Equivalently, the block diagonal constraint on \( L \) implies that the rectangular block \( \tilde{B} \) is constrained to be of the form \( \tilde{B} = \tilde{A}^{1/2}QA^{-1/2}B \) for some orthogonal matrix \( Q \).

Since we have assumed that \( V, A \) and \( \tilde{A} \) are positive definite, then any admissible revised covariance matrix

\[ \tilde{V}(Q) = LV L^T = \begin{pmatrix} \tilde{A} & \tilde{A}^{1/2}QA^{-1/2}B \\ B^T A^{-1/2}Q^T \tilde{A}^{1/2} & C \end{pmatrix} \]

is necessarily invertible, because

\[ L = \begin{pmatrix} \tilde{A}^{1/2}QA^{-1/2} & 0 \\ 0 & I \end{pmatrix} \]

is invertible (and with condition number bounded uniformly in the choice of \( Q \)).

We now are searching for the admissible \( \tilde{V} \) closest to \( V \), in terms of some suitable norm.
CHAPTER 3

Weighted Orthogonal Procrustes Problem

We want to minimize \( \| \tilde{V} - V \| \). If we take the norm to be the usual Frobenius norm

\[
\|X\|_F = \sqrt{\text{tr}(XX^T)},
\]

this is equivalent to minimizing the off-diagonal block norm:

\[
\| \tilde{B} - B \|_F = \| \tilde{A}^{1/2} Q A^{-1/2} B - B \|_F \tag{3.1}
\]
as \( Q \) varies over the orthogonal group \( O(n) \). This is an unconstrained but nonconvex problem of dimension \( n(n - 1)/2 \).

This can be formulated as the following optimization problem which is called the Weighted Orthogonal Procrustes Problem ([17]) or Double Orthogonal Procrustes Problem ([1], [15]).

**Weighted Orthogonal Procrustes Problem (W OPP):** Given matrices \( A, B \) and \( C \) of compatible sizes, find an orthogonal matrix \( Q \) minimizing \( \| AQB - C \| \). i.e.

\[
\min_{Q} \| AQB - C \| \quad \text{s.t.} \quad Q^TQ = I
\]

**Proposition 3** The canonical form of W OPP Matrices \( A \) and \( B \) belonging to a W OPP

\[
\min_{Q} \| AQB - C \| \quad \text{s.t.} \quad Q^TQ = I
\]
can always be considered as diagonal matrices with nonnegative decreasing entries.

**Proof.** Let \( A = U_A \Sigma_A V_A^T \) and \( B = U_B \Sigma_B V_B^T \) be the singular value decomposition of \( A \) and \( B \). Then we have

\[
\| U_A \Sigma_A V_A^T Q U_B \Sigma_B V_B^T - C \|_F^2 = \| \Sigma_A \tilde{Q} \Sigma_B - U_A^T C V_B \|_F^2
\]
where $\tilde{Q} = V_A^T Q U_B$ is also an orthogonal matrix. Here we use the property of Frobenius form which is invariant under orthogonal transformation, which we will mention later in this chapter. So without loss of generality we may assume that $A$ and $B$ are diagonal matrices with nonnegative descending entries.

From now on we will assume each weighted orthogonal procrustes problem is in its canonical form.

This is highly related to the following classical Orthogonal Procrustes Problem (OPP) in numerical algebra ([9]). Procrustes was a host who adjusted his guests to their bed. Procrustes, whose name means ”he who stretches”, was arguably the most interesting of Theseus’s challenges on the way to becoming a hero. He kept a house by the side of the road where he offered hospitality to passing strangers, who were invited in for a pleasant meal and a night’s rest in his very special bed. Procrustes described it as having the unique property that its length exactly matched whoever laid down upon it. What Procrustes didn’t volunteer was the method by which this ”one-size-fits-all” was achieved, namely as soon as the guest laid down Procrustes went to work upon him, stretching him on the rack if he was too short for the bed and chopping off his legs if he was too long. Theseus turned the tables on Procrustes, fatally adjusting him to fit his own bed.

Before we start the OPP and WOPP, we will review the trace of a matrix. In linear algebra, the trace of an $n \times n$ square matrix $A$ is defined to be the sum of the elements on the main diagonal, i.e.,

$$\text{tr}(A) = \sum_{i=1}^{n} a_{ii},$$

where $a_{ij}$ represents the entry on the $i$th row and $j$th column of $A$.

The trace has the following properties:

1. It is a linear map and invariant under transposition: $\text{tr}(aA + bB) = a \text{tr}(A^T) + b \text{tr}(B^T)$ for scalars $a$ and $b$.

2. It is invariant under commutation and cyclic permutation:

$$\text{tr}(AB) = \text{tr}(BA),$$

$$\text{tr}(ABC) = \text{tr}(CAB) = \text{tr}(BCA).$$
3. It is similarity-invariant. That is, for any invertible $P$,

$$\text{tr}(A) = \text{tr}(P^{-1}AP).$$

For an $m \times n$ matrix $A$ with real entries, we have

$$\text{tr}(A^T A) \geq 0.$$

with equality only if $A = 0$. The assignment

$$< A, B > = \text{tr}(A^T B)$$

yields an inner product on the space of all real $m \times n$ matrices. The norm induced by the above inner product is called the Frobenius norm of a matrix. Indeed it is simply the Euclidean norm if the matrix is considered as a vector of length $n^2$.

In optimization the derivatives of the traces are important to obtain the optimal solution or more analysis. Let us define

$$\frac{\partial}{\partial X} = (\frac{\partial}{\partial X_{ij}}) \text{ where } X = (X_{ij})$$

The following derivatives are going to be used in later chapters ([22]).

$$\frac{\partial}{\partial X} \text{tr}(X) = I,$$

$$\frac{\partial}{\partial X} \text{tr}(AXB) = A^T B^T,$$

$$\frac{\partial}{\partial X} \text{tr}(AX^T B) = BA,$$

$$\frac{\partial}{\partial X} \text{tr}(AX^k) = \sum_{r=0}^{k-1} (X^r A X^{k-r-1})^T.$$

For any matrix $A \in \mathbb{R}^{m \times n}$, the Frobenius norm has the form

$$\|A\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}.$$

We will use the Frobenius in the following text because the following fact:

$$\|M\|_F^2 = \text{tr}(MM^T).$$
The Frobenius norm is also invariant with respect to orthogonal transformation. It is easy to verify that for all orthogonal $Q$ and $Z$ of appropriate dimensions we have

$$\|QAZ\|_F = \|A\|_F.$$ 

From now on, we will drop the subscript $F$ on the Frobenius norm $\| \cdot \|_F$.

We will use the following Singular Value Decomposition theorem without proof. For details, please see [9].

**Theorem 1 ([9]) (Singular Value Decomposition (SVD))** If $A$ is a real $m$-by-$n$ matrix, then there exist orthogonal matrices

$$U = [u_1, \ldots, u_m] \in \mathbb{R}^{m \times m} \text{ and } V = [v_1, \ldots, v_n] \in \mathbb{R}^{n \times n}$$

such that

$$U^T AV = \text{diag}(\sigma_1, \ldots, \sigma_p) \in \mathbb{R}^{m \times n} \quad p = \min\{m, n\}$$

where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p \geq 0$.

### 3.1 Orthogonal Procrustes Problem

Suppose $A, C \in \mathbb{R}^{m \times n}$. We need to find an orthogonal matrix $Q \in \mathbb{R}^{n \times n}$ to minimize $\|AQ - C\|(9)$. i.e.

$$\min_Q \|AQ - C\| \text{ s.t. } Q^TQ = I$$

$$\min_Q \|AQ - C\|^2 = \min_Q (\text{tr}(AQ - C)(AQ - C)^T)$$
$$= \min_Q (\text{tr}(AQQ^T A^T) + \text{tr}(CC^T) - 2 \text{tr}(AQC^T))$$
$$= \min_Q (\text{tr}(AA^T) + \text{tr}(CC^T) - 2 \text{tr}(AQC^T))$$

We need to maximize $\text{tr}(AQC^T)$ which is equal to $\text{tr}(C^T AQ)$. If we take the singular value decomposition (SVD) of $C^T A$, which is $C^T A = U \Sigma V^T$, and define the orthogonal matrix $Z$ by $Z = V^T Q U$, then

$$\text{tr}(C^T AQ) = \text{tr}(U \Sigma V^T Q) = \text{tr}(V^T Q U \Sigma) = \text{tr}(Z \Sigma) = \sum_{i=1}^p z_i \sigma_i \leq \Sigma \sigma_i.$$ 

Clearly, the upper bound is attained by setting $Q = VU^T$ in which case $Z = I_n$. 

11
3.2 Polar Decomposition

In the case of $A = I$, the closest orthogonal matrix to a square matrix is the orthogonal factor of its polar decomposition.

**Definition 1** The polar decomposition of real matrix $C$ is a matrix decomposition of the form

$$ C = SQ $$

where $Q$ is an orthogonal matrix and $S$ is a symmetric positive semi-definite matrix.

The following proposition shows that $Q$ is the closest orthogonal matrix to $C$ in the above definition.

**Proposition 4** The closest orthogonal matrix to a real matrix $C$ is any orthogonal matrix $U$ which occurs in a polar decomposition $C = (CC^T)^{\frac{1}{2}}U$ of $A$. The exponent $\frac{1}{2}$ refers to the nonnegative semi-definite square root. If $C$ is invertible then $U = (AA^T)^{-\frac{1}{2}}A$ is the unique closest orthogonal matrix to $A$.

Proof: Since $U$ is orthogonal, $UU^T = I$. Let $Q$ be any orthogonal matrix and $V = Q - U$. From the orthogonality of $Q$ and $U$ it follows that

$$ UV^T + VU^T + VV^T = 0 \quad (3.2) $$

Thus any orthogonal $Q$ can be written as $Q = U + V$ where $V$ satisfies (3.2). Now we can write

$$ \|C - Q\|^2 = \|C - U - V\|^2 \\
= \|C - U\|^2 - \text{tr}((C - U)V^T + V(C^T - U^T) - VV^T) \\
= \|C - U\|^2 - \text{tr}(CV^T + VC^T) \quad (3.3) $$

The last equality follows from (3.2). Next we use the polar decomposition of $C$ in the last term in (3.3), then use the invariance of the trace of a product under cyclic permutation of the factors and finally use (3.2) to obtain

$$ \text{tr}(CV^T + VC^T) = \text{tr}((CC^T)^{\frac{1}{2}}UV^T + UV^T(CC^T)^{\frac{1}{2}}) \\
= \text{tr}((CC^T)^{\frac{1}{2}}(UV^T + VU^T)) \\
= -\text{tr}((CC^T)^{\frac{1}{2}}VV^T). \quad (3.4) $$
Since $(CC^T)^{\frac{1}{2}}$ and $VV^T$ are both symmetric and nonnegative, the trace of their product is nonnegative.

By using (3.4) in (3.3) and noting that the last trace in (3.4) is nonnegative, we obtain

$$\|C - Q\|^2 = \|C - U\|^2 + \text{tr}((CC^T)^{\frac{1}{2}}VV^T) \geq \|C - U\|^2. \quad (3.5)$$

This proves the first part of the theorem. If $C$ is invertible then $CC^T$ is invertible, so the trace in (3.5) can vanish if and only if $V = 0$. Therefore the inequality holds in (3.5) unless $Q = U$. This proves the second part of the theorem.

Higham (1986) also proves this result in a complex field.

**Theorem 2** \([12]\) Let $A, B \in \mathbb{C}^{m \times n}$, let $B^H A \in \mathbb{C}^{n \times n}$ have the polar decomposition $B^H A = UH$ where $B^H$ represents the conjugate transpose of $B$, then for any unitary $Z \in \mathbb{C}^{n \times n}$,

$$\|A - BU\| \leq \|A - BZ\| \leq \|A + BU\|$$

When $m = n$ and $B = I$, we have

$$\|A - U\| \leq \|A - Z\| \leq \|A + U\|$$

i.e, the nearest unitary matrix to $A \in \mathbb{C}^{n \times n}$ is the unitary factor of the polar decomposition.

E. Klassen (16) also proves similar theorem. He also gives the number of critical points in his notes.

**Theorem 3** \([16]\) Let $Q \in \text{GL}(n, \mathbb{R})$ and let $Q = AS$ be the unique decomposition of $Q$ such that $A \in \text{O}(n, \mathbb{R})$ and $S$ is positive definite symmetric. Then $A$ is closer to $Q$ than any other element of $\text{O}(n, \mathbb{R})$.

In the proof of this theorem, Dr. Klassen also showed that the number of critical points of function $f(Q) = \|Q - A\|$ is $2^n$. Later we will see that even for a very special case of WOPP has much more critical points than this function.

### 3.3 Geometric Analysis

The $n \times n$ real matrices $Q$ such that $QQ^T = Q^TQ = I$ is the orthogonal group, denoted $\text{O}(n)$. We will make an intuitive introduction to differential geometry in Euclidean space, see ([6]) for details.
3.3.1 Tangent and Normal Spaces

Intuitively, the tangent space at a point is the plane tangent to the sub-manifold at that point, as shown in Figure (3.1). The normal space is the orthogonal complement. On the sphere, tangents are perpendicular to radii, and the normal space is radial. We will derive the equations for the tangent and normal spaces on the orthogonal group and compute the projection operators on this space.

Because the orthogonal group is a group and a manifold, it is a Lie group. $O(n)$ has a sub-manifold tangent space at the identity that is the Lie algebra of antisymmetric matrices $O(n)$. In fact, the orthogonal group is a compact Lie group. Suppose the entries of $Q$ are differentiable functions of $t$, and that $t = 0$ gives $Q = I$. Differentiating the orthogonality condition

$$Q^T Q = I$$

yields

$$\dot{Q}^T Q + Q^T \dot{Q} = 0$$

Evaluation at $t = 0$ ($Q = I$) then implies

$$\dot{Q}^T = -\dot{Q}.$$
In Lie group terms, this means that the Lie algebra of an orthogonal matrix group consists of skew-symmetric matrices. Going in the other direction, the matrix exponential of any skew-symmetric matrix is an orthogonal matrix.

The normal space is defined to be the orthogonal complement of the tangent space. Orthogonality depends upon the definition of an inner product. We will choose the standard inner product

$$< A, B >= \text{tr}(A^T B),$$

in \(n^2\)-dimensional Euclidean space, which is also the Frobenius inner product for \(n \times n\) matrices. The normal space at a point \(P\) consists of all matrices \(N\) satisfying

$$\text{tr} \Delta^T N = 0$$

for all \(\Delta\) in the tangent space. It follows that the normal space has dimension \(n(n + 1)/2\). It is easily verified that if \(N = QS\), where \(S\) is symmetric and \(Q\) is orthogonal, then \(N\) is in the normal space. Since the dimension of the space of such matrices is \(n(n+1)/2\), we see the normal space is exactly the set of matrices \(\{QS\}\), where \(S\) is any \(n\)-by-\(n\) symmetric matrix.

Let \(Z\) be any \(n\)-by-\(n\) matrix. We also let \(\text{sym}(A)\) denote \((A + A^T)/2\) and \(\text{skew}(A) = (A - A^T)/2\), it is easily verified that at \(Q\)

$$\pi_N(Z) = Q \text{sym}(Q^T Z)$$

defines a projection of \(Z\) onto the normal space. Similarly, at \(Q\),

$$\pi_T(Z) = Q \text{skew}(Q^T Z) \quad (3.6)$$

is a projection of \(Z\) onto the tangent space at \(Q\). This suggests that the tangent directions \(\Delta\) at \(Q\) then have the general form,

$$\Delta = QA$$

where \(A\) is \(n\)-by-\(n\) and skew-symmetric.

If we denote

$$S(n):= \{\text{all symmetric matrices in } R^{n \times n}\},$$

15
it follows that the tangent space $T_QO(n)$ and the normal space $N_QO(n)$ of $O(n)$ at any $Q \in O(n)$ are given, respectively, by:

$$T_QO(n) = QS(n)^\perp,$$
$$N_QO(n) = QS(n),$$

where $S(n)^\perp$ is the orthogonal complement of $S(n)$ with respect to the Frobenius inner product and hence consists of all skew-symmetric matrices in $\mathbb{R}^{n \times n}$.

### 3.3.2 Embedded Geodesics.

A geodesic is the curve of shortest length between two points on a Riemannian manifold. For the case of manifolds embedded in a Euclidean space the acceleration vector at each point along a geodesic is normal to the sub-manifold so long as the curve is traced with uniform speed. This condition is sufficient and necessary. In the case of a sphere, acceleration for uniform motion on a great circle is directed radially and therefore normal to the surface. So great circles are geodesics on the sphere.

In the case of an orthogonal group, the geodesic equation is obtained by $Q^T \dot{Q} = \text{constant}$, yielding the simple solution

$$Q(t) = Q(0)e^{At}. \quad (3.7)$$

### 3.4 A necessary condition

**Proposition 5** $Q$ is a critical point of $f(Q) = \|AQB - C\|^2$ if and only if $Q^T A^T (AQB - C) B^T$ is symmetric.
**Proof:** Any critical point $Q$ satisfies

$$(AQB - C) \perp AQ \cdot O(n) \cdot B.$$  

From the property of inner product, for any vectors $v, w$ and matrix $A$,

$$< Av, w > = < v, A^T w > .$$

Thus, we need to find a $Q$ such that

$$A^T (AQB - C) B^T \perp Q \cdot O(n),$$

i.e.

$$Q^T A^T (AQB - C) B^T \perp O(n).$$

From the expression of normal space of $O(n)$ we deduce in §3.3.1, any critical point $Q$ which satisfies $Q^T A^T (AQB - C) B^T$ is symmetric.

The reverse can be proven similarly.  

In the case of $C = 0$, the necessary condition becomes that $Q^T A^T AQBB^T$ is symmetric. We will work on this case in more detail to understand the number and the properties of critical points.
We assume $A^TA = S_1$ and $BB^T = S_2$. Obviously $S_1$ and $S_2$ are symmetric. We are looking for an orthogonal matrix $Q$ such that $Q^T S_1 Q S_2$ is symmetric. We need the following proposition to simplify our problem.

**Proposition 6** Two symmetric matrices $M_1$ and $M_2$ commute if and only if $M_1 M_2$ is symmetric.

**Proof:** $(\implies)$ We know

$$(M_1 M_2)^T = M_2^T M_1^T = M_2 M_1 = M_1 M_2,$$

so $M_1 M_2$ is symmetric. Here the third equality follows from the fact that $M_1$ and $M_2$ commute.

$$(\iff) \quad M_1 M_2 = (M_1 M_2)^T = M_2^T M_1^T = M_2 M_1,$$

so $M_1$ and $M_2$ commute. Here the first equality comes from $M_1 M_2$ symmetric. 

If $S_3 = Q^T S_1 Q$, then $S_3$ is symmetric. The necessary condition becomes that $S_3 S_2$ is symmetric. From Proposition 6, this is equivalent to saying that $S_2$ and $S_3$ commute. If we assume $S = Q^T S_1 Q S_2$, then we are looking for $Q$ such that $S$ is symmetric.

Let us determine what the $Q$ looks like. For simplicity, we will assume that $S_1$ and $S_2$ have distinct eigenvalues.

Now let us take the eigenvalue decomposition of $S_1$, say, $S_1 = Q_1 \Sigma_1 Q_1^T$ and the eigenvalue decomposition of $S_2$ is $S_2 = Q_2 \Sigma_2 Q_2^T$. When $Q = Q_1 Q_2^T$, we have

$$S = Q^T Q_1 \Sigma_1 Q_1^T Q_2 \Sigma_2 Q_2^T$$
$$= Q_2 Q_1^T Q_1 \Sigma_1 Q_1^T Q_1 Q_2^T Q_2 \Sigma_2 Q_2^T$$
$$= Q_2 \Sigma_1 \Sigma_2 Q_2^T$$

$\Sigma_1$ and $\Sigma_2$ are diagonal, so it is easy to verify that $S$ is symmetric.

We will show that there are no other matrices which can make $S$ symmetric. Before we do that, we need to show the following theorem for commuting matrices. Let us define a subset of orthogonal group $O(n)$. For a $n \times n$ matrix $M$, $D(M) = \{Q : Q^T M Q \text{ is diagonal}\}$. 

\[ \text{Page 18} \]
Proposition 7  Symmetric matrices $S_1, S_2 \in \mathbb{R}^{n \times n}$, with distinct eigenvalues commute if and only there exists a $Q \in O(n)$ which can simultaneously diagonalize $S_1$ and $S_2$, i.e. $Q \in D(S_1) \cap D(S_2)$.

Proof ($\implies$) Since $S_1$ is symmetric, we can find $Q \in O(n)$ s.t. $Q^T S_1 Q = \Lambda_1$ where $\Lambda_1$ is diagonal, which means $S_1 = Q \Lambda_1 Q^T$.

Because $S_1$ and $S_2$ commute, we have

$$S_1 S_2 = S_2 S_1 \iff Q \Lambda_1 Q^T S_2 = S_2 Q \Lambda_1 Q^T$$
$$\iff \Lambda_1 Q^T S_2 Q = Q^T S_2 \Lambda_1$$

Let $Q^T S_2 Q = Y$, then $Y$ is symmetric and $\Lambda_1 Y = Y \Lambda_1$. We write $\Lambda_1 = diag(\lambda_1, \lambda_2, \cdots, \lambda_n)$ and $Y = (y_{ij})$.

We know $(\Lambda_1 Y)_{ij} = \lambda_i y_{ij}$ and $(Y \Lambda_1)_{ij} = \lambda_j y_{ij}$. $\Lambda_1 Y = Y \Lambda_1$ gives $\lambda_i y_{ij} = \lambda_j y_{ij}$. For $i \neq j$, we have $y_{ij} = 0$, i.e. $\Lambda_2 = Q^T S_2 Q$ is diagonal.

($\iff$) From $Q^T S_1 Q = \Lambda_1$ and $Q^T S_2 Q = \Lambda_2$, we have $S_1 = Q \Lambda_1 Q^T$ and $S_2 = Q \Lambda_2 Q^T$.

$$S_1 S_2 = Q \Lambda_1 Q^T Q \Lambda_2 Q^T = Q \Lambda_1 \Lambda_2 Q^T$$
$$S_2 S_1 = Q \Lambda_2 Q^T Q \Lambda_1 Q^T = Q \Lambda_2 \Lambda_1 Q^T$$

Two diagonal matrices $\Lambda_1$ and $\Lambda_2$ always commute. i.e. $\Lambda_1 \Lambda_2 = \Lambda_2 \Lambda_1$, so we have $S_1 S_2 = S_2 S_1$.

Now we have

Proposition 8  If $Q$ satisfies that $Q^T S_1 Q S_2$ is symmetric, then $Q = Q_1 Q_2^T$ for some $Q_1 \in D(S_1)$ and $Q_2 \in D(S_2)$.

Proof: Let’s suppose there is a $Q = Q_s$ such that $Q_s^T S_1 Q_s S_2$ is symmetric, from Proposition 6, which means that $Q_s^T S_1 Q_s$ commutes with $S_2$. From Proposition 7, there always exists a $Q_2$ such that $Q_2 \in D(S_2) \cap D(Q_s^T S_1 Q_s)$. From the fact that $Q_2 \in D(Q_s^T S_1 Q_s)$, we know $Q_s Q_2 \in D(S_1)$. This means $Q_s = Q_1 Q_2^T$ for some $Q_1 \in D(S_1)$.
The theorem which gives the critical points of function \( f(Q) = \|AQB\|^2 \) follows.

Given that \( A^T A \) and \( BB^T \) have distinct eigenvalues, and there exists \( Q_1 \) and \( Q_2 \) such that \( Q_1^T A^T AQ_1 \) and \( Q_2^T BB^T Q_2 \) are diagonal matrices, the critical points of function \( f(Q) = \|AQB\|^2 \) has form \( Q_1 Q_2^T \).

**Proposition 9** Suppose that \( A^T A \) and \( BB^T \) have distinct eigenvalues, where \( A \) and \( B \) are \( n \times n \) matrices. Let \( f : O(n) \to \mathbb{R} \) be defined by \( f(Q) = \|AQB\|^2 \). Then \( Q \) is a critical point of \( f \) if and only if there exist \( Q_1 \in D(A^T A) \) and \( Q_2 \in D(BB^T) \) such that \( Q = Q_1 Q_2^T \).

**Proof:** The result follows from Proposition 5 and Proposition .

Let us investigate the number of critical points of \( f(Q) = \|AQB\|^2 \) now. We already know that the critical points have form \( Q_1 Q_2^T \) so it suffices to count the number of \( Q_1 \) and \( Q_2 \).

We know \( Q_1 \) and \( Q_2 \) are not unique. For any particular matrix \( Q_1^0 \in D(A^T A) \), \( Q_1 \) can be expressed as \( Q_1^0 PD \), where \( P \) is a permutation matrix and \( D \) is a signature matrix.

The permutation matrix \( P \) is a matrix obtained by permuting the rows of an \( n \times n \) identity matrix according to some permutation of the numbers 1 to \( n \). Every row and column therefore contains precisely a single 1 with 0s everywhere else, and every permutation corresponds to a unique permutation matrix. There are therefore \( n! \) permutation matrices of size \( n \), where \( n! \) is a factorial.

The signature matrix \( D \) is a diagonal matrix whose diagonal elements are positive or negative 1. Therefore there are \( 2^n \) signature matrices of size \( n \).

We also pick a matrix \( Q_2^0 \in D(BB^T) \), then \( Q_2 \) can be expressed as \( Q_2^0 P_2 D_2 \), and \( Q_1 Q_2^T \) can be written as \( Q_1^0 P_1 D_1 D_2^T P_2^T Q_2^0 \).

**Proposition 10** Assume that \( A^T A \) and \( BB^T \) have distinct eigenvalues, then \( f(Q) = \|AQB\|^2 \) has \( 2^n n! \) critical points.

**Proof:** Since \( Q_1 Q_2^T \) can be written as \( Q_1^0 P_1 D_1 D_2^T P_2^T Q_2^0 \). It suffices to count the number of \( P_1 D_1 D_2^T P_2^T \). \( D_2^T \) applied to \( D_1 \) is still a signature matrix \( \tilde{D} \). \( P_1 \tilde{D} \) is equivalent to \( \tilde{D} P_1 \) so \( P_1 \tilde{D} P_2^T \) may be written as \( \tilde{D} P_1 P_2^T \). \( P_1 P_2^T \) is a new permutation matrix \( \tilde{P} \) so we only need to count the number of \( \tilde{D} \tilde{P} \) which is \( 2^n n! \).
Unfortunately, this result does not help us count the number of critical points of $f(Q) = \|AQB - C\|^2$ for $C \neq 0$ since $f(Q)$ is not a linear projection. The problem of evaluating the number and positions of critical points of $f(Q) = \|AQB - C\|^2$ is still open.

When $A$ and $B$ have distinct singular values, we may find the location of all the minimum points.

**Proposition 11** Assume that $A$ and $B$ have distinct singular values, the singular value decompositions of $A$ and $B$ are $A = Q_A\Sigma_AQ_A^T$ and $B = Q_B\Sigma_BQ_B^T$, respectively, where $\Sigma_A = \text{diag}(a_1, a_2, \ldots, a_n)$ with $a_1 > a_2 > \cdots > a_n \geq 0$ and $\Sigma_B = \text{diag}(b_1, b_2, \ldots, b_n)$ with $b_1 > b_2 > \cdots > b_n \geq 0$. Then the $2^n$ global minimum points of $f(Q) = \|AQB\|^2$ are

$$Q = Q_A \begin{pmatrix} 0 & 0 & \cdots & 0 & \pm 1 \\ 0 & 0 & \cdots & \pm 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \pm 1 & \cdots & 0 & 0 \\ \pm 1 & 0 & \cdots & 0 & 0 \end{pmatrix} Q_B^T.$$ (3.8)

**Proof:**

$$f(Q) = \|AQB\|^2 = \|Q_A\Sigma_AQ_A^TQQ_B\Sigma_BQ_B^T\|^2 = \|\Sigma_AQ_A^TQQ_B\Sigma_B\|^2.$$

From above two propositions, we know that the critical points have form $Q_1Q_2^T$ where $Q_1 \in D(A^TA)$ and $Q_2 \in D(BB^T)$. It is easy to verify that $Q_A \in D(A^TA)$ and $Q_B \in D(BB^T)$. So the critical points have from $Q = Q_AQ_B^T$. Then at the critical points, $f(Q) = \|\Sigma_APD\Sigma_B\|^2$ for some permutation matrix $P$ and some signature matrix $D$.

Since $\Sigma_A$ and $\Sigma_B$ are diagonal matrices with nonnegative elements, $f(Q)$ equals $\sum_{i,j=1}^n a_ib_j$ where $i, j$ only appears once. Now let us prove that the minimum of $\sum_{i,j=1}^n a_ib_j$ is $a_1b_n + a_2b_{n-1} + \cdots + a_nb_1$.

First we will show this is true when $n = 2$.

\[
(a_1b_2 + a_2b_1) - (a_1b_1 + a_2b_2) = a_1(b_2 - b_1) + a_2(b_1 - b_2) = (a_1 - a_2)(b_2 - b_1) \leq 0
\]
We need to show that

\[ a_1 b_n + a_2 b_{n-1} + \cdots + a_n b_1 < a_1 b_i + a_2 b_{i-1} + \cdots + a_n b_i \]  \hspace{1cm} (3.9)

where \(i_1, i_2, \ldots, i_n\) are new orders of \(b_i\)'s.

If \(i_1 \neq 1\), \(b_1\) appears in term \(a_k b_1\) on the right hand side of 3.9, we have

\[
(a_k b_1 + a_n b_i) - (a_k b_i + a_n b_1) = a_k (b_1 - b_i) - a_n (b_1 - b_i) \\
= (a_k - a_n) (b_1 - b_i) \\
\geq 0.
\]

We may continue this process with the terms including \(b_2, b_3, \ldots, b_n\) and we find out that \(a_1 b_n + a_2 b_{n-1} + \cdots + a_n b_1\) is no more than the sum of any other combinations.

To get \(a_1 b_n + a_2 b_{n-1} + \cdots + a_n b_1\), we need to have

\[
PD = \begin{pmatrix}
0 & 0 & \cdots & 0 & \pm 1 \\
0 & 0 & \cdots & \pm 1 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & \pm 1 & \cdots & 0 & 0 \\
\pm 1 & 0 & \cdots & 0 & 0
\end{pmatrix}
\]

So 3.8 follows, and there are \(2^n\) of them.
CHAPTER 4

Local Minimization Techniques

Based on the geometric analysis in §3, we consider the following algorithm to minimize the objective function \( f(Q) = \| AQB - C \|^2 \).

4.1 An Algorithm based on geometric projection

We need to minimize the objective function

\[
  f(Q) = \| AQB - C \|^2 = \text{tr}((AQB - C)^T(AQB - C)) = \text{tr}(B^TQ^TA^TAQB - 2AQBCT + CTC) \tag{4.1}
\]

**Algorithm 1**

- Choose any orthogonal matrix \( Q \) as a starting point.

- Calculate the gradient of \( f \) at \( Q \). From the properties of \( \text{tr} \) function we introduced in Chapter 3, we have

  \[
  g_Q = 2A^TAQBB^T - 2A^TCB^T.
  \]

- Project \( g_Q \) onto the tangent space of \( O(n) \) at \( Q \). From (3.6), we have

  \[
  u = \pi_T(g_Q) = A^TAQBB^T - A^TCB^T - QBB^TQ^TA^TAQ + QBC^TAQ
  \]

- Move a small distance \( t \) from \( Q \) along a geodesic in \( O(n) \) in the direction of the negative of the projected gradient \( u \). From 3.7, we can get the destination point: \( Q \cdot \exp(-t \cdot Q^T \cdot u) \)

- Take the point arrived in the last step as the new starting point, continue the process until \( \| u \| \) is very small.
We tried this algorithm on some small size matrices and found that it worked well. When it comes to the large size matrices, the question becomes complicated. The choice of step length $t$ seems difficult. Before we investigate further, we read Chu and Trendafilov’s papers ([4], [5]) on this problem. It seems that our method is very similar to theirs but they explored a framework of differential equations which make their algorithm continuous.

Chu and Trendafilov ([4], [5]) consider the gradient of the objective function

$$f(Q) = \|AQB - C\|^2 = \frac{1}{2} < AQB - C, AQB - C >$$

to be

$$\nabla F(Q) = A^T(AQB - C)B^T.$$  

The projection $g(Q)$ of the gradient $\nabla F(Q)$ at a point $Q \in O(n)$ onto the tangent space $T_QO(n)$ can be computed explicitly. Then the differential equation

$$\frac{dQ}{dt} = -g(Q)$$

naturally defines a flow $Q(t)$ on the feasible manifold $O(n)$. Along the flow $Q(t)$ the function value $F(Q(t))$ is decreasing most rapidly relative to any other direction. Indeed, we have

$$\frac{dF(Q(t))}{dt} = < \nabla F(Q(t)), -g(Q(t)) > = -\|g(Q(t))\|^2 \quad (4.2)$$

Due to the fact that $g(Q(t))$ is the orthogonal projection of $\nabla F(Q(t))$ onto $T_Q(t)O(n)$, we may note from (4.2) that the value of $F(Q(t))$ decreases strictly until $g(Q(t))$ becomes zero, indicating a critical point has been reached. Most likely this critical point is a local minimum rather than a saddle point which will be shown in later chapters. The descent property is universal regardless of where the flow starts. The flow is guaranteed to converge globally.

Chu and Trendafilov deduce that the projection $g(Q)$ of $\nabla f(Q)$ onto the tangent space $T_QO(n)$ has the form:

$$g(Q) = \frac{Q}{2} (Q^T A^T (AQB - C) B^T - B(ABQ - C)^T ) A Q).$$

Therefore the differential equation
\[
\frac{dQ}{dt} = \frac{Q}{2} (B( AQB - C)^T AQ - Q^T A^T (AQB - C) B^T )
\]
defines a steepest descent flow on the manifold \( O(n) \) for the objective function \( f \). This can be solved by any available initial value problem solver.

We will report the numerical results of this algorithm in later chapters. Now let’s introduce another local minimization method.

4.2 An Algorithm Based On Majorization

4.2.1 Theory of Iterative Majorization

When the objective function is difficult to minimize, we adopt an iterative majorization method. The main feature of iterative majorization is that it converts the function to easy-to-minimize one. If there is a condition which can make the sequence converge, we usually end up at a stationary point that is a local minimum.

Let \( f(x) \) denote the objective function, we call the auxiliary function \( F(x, y) \) (we assume that \( x \) and \( y \) both vary in the same domain \( \Omega \)) a **majorizing function** if it satisfies\([23]\):

1. The auxiliary function \( F(x, y) \) is easier to minimize.
2. The objective function can’t exceed the auxiliary anywhere. That means \( F(x, y) \geq f(x) \) for all \( y \).
3. The auxiliary function should touch the objective function at the so-called **supporting point** \( y \). That is \( F(y, y) = f(y) \) for all \( y \).

According to our construction, the function \( F(x, y) \) is easier to simplify. The most common way to affect the iterative majorization method is to choose the majorizing function so that its minimum exists and is unique. Let us say

\[
x^* = \arg \min_{x \in \Omega} F(x, y).
\]

The last two requirements of the majorizing function imply the chain of inequalities

\[
f(x^*) \leq F(x^*, y) = \min_{x \in \Omega} F(x, y) \leq F(y, y) = f(y).
\]

Then it is possible to build an iterative majorization algorithm:
1. Start at $x^{(0)}$.

2. Set $x^{(k+1)}$ equal to the minimum argument of the function $F(x, x^{(k)})$.

3. If $f(x^{(k)}) - f(x^{(k+1)}) > \epsilon$ then go to step 2.

4. Otherwise stop with $x := x^{(k+1)}$.

**Illustrative Example: Iterative Calculation of the Median**

We will apply the theory of iterative majorization to a simple example: the calculation of the median. Given $n$ observations $v_i, i = 1, \ldots, n$, the median corresponds to the minimizing argument of

$$f(x) = \sum_i |v_i - x|. \quad (4.4)$$

This objective function is linear and convex (see Figure 4.1), and the derivative is not continuous. Heiser [11] proposed the family of quadratic functions

$$F(x, y) = \frac{1}{2} f(y) + \frac{1}{2} \sum_i |v_i - y|^{-1}(v_i - x)^2. \quad (4.5)$$

By using the inequality

$$(|v_i - x| - |v_i - y|)^2 \geq 0,$$

it may be verified easily that (4.5) satisfies the requirements 2 and 3 for the definition of a majorizing function. Starting from any initial estimate $y = x_0 \neq v_i$, we have to minimize the second part at the right side of (4.5), i.e. we have to compute a weighted mean with weights proportional to $|v_i - y|^{-1}$. According to (4.3), an improved value $f(x^*)$ is obtained and computation of iteratively re-weighted means may be continued until $x^* = y$. At this point, $y$ satisfies the stationary equation for the least square function (4.5)

$$\sum_i (v_i - y)/|v_i - y| = \sum_i \text{sign}(v_i - y) = 0, \quad (4.6)$$

which is identical to the stationary equation for (4.4). Thus the process stops at a desirable point. Note that (4.6) implies that when $n$ is odd, $x^*$ should be chosen as the element $v_{(n/2)}$, where $v_{(i)}$ denotes the $i$-th element of the ordered observations, and when $n$ is even, that any value in the interval between $v_{(n/2)}$ and $v_{((n+1)/2)}$ is equivalent, because it always yields two residuals with opposite signs.
4.2.2 Iterative Majorization on WOPP

We consider the square of our objective function (4.1) which is quadratic in $Q$. Generally speaking, quadratic functions are simple enough to minimize, but there are circumstances where it is useful to majorize them. In our case, the quadratic term makes the problem very hard to optimize. Heiser([11]) suggests that we use majorizing functions for general quadratic functions with positive curvature. For a function with $n$ variables of the form

$$\phi(x) = x^T A x + b^T x + c.$$ \hfill (4.7)

The function has positive curvature in all directions when $A$ is positive definite. $\phi(x)$ can be majorized by

$$\mu(x, \bar{x}) = x^T P x + (b + 2N \bar{x})^T x + (c - x^T N \bar{x}),$$ \hfill (4.8)

where $A$ has been decomposed as $A = P + N$, with $P$ some positive definite matrix of simpler form than $A$ itself, and $N$ some negative definite matrix. When $N$ is negative definite, we have

$$(x - \bar{x})^T N (x - \bar{x}) \leq 0.$$ \hfill (4.9)

Substituting the decomposition $A = P + N$ into (4.7) and using (4.9) yields the inequality $\phi(x) \leq \mu(x, \bar{x})$.

One way to decompose $A$ is to let $P = \rho^2 I$ for all choices of $\rho^2$ exceeding the largest
eigenvalue of $A$. In this situation, for all $x \in \mathbb{R}^n$ we have

\[
x^T A x \leq \rho^2 x^T x \Rightarrow x^T (A - \rho^2 I) x \leq 0
\]

\[
\Rightarrow x^T N x \leq 0
\]

\[
\Rightarrow (x-x)^T N (x-x) \leq 0
\]

The last inequality ensures that (4.8) is a majorizing function of $\phi(x)$ when $P = \rho^2 I$. From the above analysis, it is easy to write one majorizing function of (4.1) at $Q_{i+1}$. i.e.

\[
F(Q_{i+1}, Q_i) = \text{tr}(\rho^2 B^T B) - 2 \text{tr}(B^T Q_{i+1}^T (A^T C + (\rho^2 - A^T A) Q_i B)) + \text{tr}(C^T C + B^T Q_i^T (\rho^2 - A^T A) Q_i B)
\]

(4.10)

It is easy to verify that this function majorizes $f(Q_{i+1}) = \|AQ_i B - C\|^2$. It defines a sequence of $Q$'s and is linear in $Q_{i+1}$ so it is easy to minimize. We may construct an iterative algorithm easily from here.

This particular majorizing function method coincides the way Koschat and Swayne propose in ([17]), although they do not describe in that way. Let us introduce their work here.

They pick a matrix $A_r$ such that $A^T A + A_r^T A_r = r^2 I$ for some scalar $r$ where $r^2$ exceeds the largest eigenvalue of $A$. We have

\[
f(Q) = \|AQ B - C\| = \| \begin{pmatrix} A \\ A_r \end{pmatrix} Q B - \begin{pmatrix} C \\ A_r Q_i B \end{pmatrix} \|.
\]

They consider the sequence defined by

\[
f_{i+1}(Q_{i+1}) = \| \begin{pmatrix} A \\ A_r \end{pmatrix} Q_{i+1} B - \begin{pmatrix} C \\ A_r Q_i B \end{pmatrix} \|^2.
\]

where $Q_i$ is assumed to be known. The function $f_{i+1}(Q_{i+1})$ is linear in $Q_{i+1}$, so it can be optimized easily. This suggests an algorithm which starting from some initial point $Q_1$, will hopefully converge to some point $Q$. Convergence of such an algorithm is easily established. We will show this in the framework of iterative majorization. We write $f_{i+1}(Q_{i+1}) = F(Q_{i+1}, Q_i)$. 

28
Consider the function

\[ F^2(Q_{i+1}, Q_i) = \| (A A_r) Q_{i+1} B - (C A_r Q_i B) \|^2 \]

\[ = \text{tr}(B^T Q_{i+1} (A^T A_r^T) - (C^T B^T Q_i^T A_r^T)) \]

\[ = \text{tr}(B^T Q_{i+1} (A^T A + A_r^T A_r) Q_{i+1} B) - 2 \text{tr}(B^T Q_{i+1} (A^T C + A_r^T A_r Q_i B)) \]

\[ + \text{tr}(C^T C + B^T Q_i^T A_r^T A_r Q_i B) \]

\[ = \text{tr}(r^2 B^T B) - 2 \text{tr}(B^T Q_{i+1} (A^T C + A_r^T A_r Q_i B)) \]

\[ + \text{tr}(C^T C + B^T Q_i^T A_r^T A_r Q_i B) \]

Comparing above function with (4.10), we find the only difference is the integers \( r \) and \( \rho \).

It is easy to verify that \( F(Q_{i+1}, Q_i) \geq f(Q_{i+1}) \) and \( F(Q_i, Q_i) = f(Q_i) \). \( F(Q_{i+1}, Q_i) \) is easier to minimize since it is linear in \( Q_{i+1} \). These three conditions make \( F(Q_{i+1}, Q_i) \) a majorizing function of \( f(Q_{i+1}) \). This majorizing function is not unique. There might exist some other majoring functions which may make the convergence faster, but empirical results show that this is already good enough for our application.

We take \( Q_i \) as given. To minimize \( F(Q_{i+1}, Q_i) \) on \( Q_{i+1} \), we only need to maximize \( \text{tr}(B^T Q_{i+1} (A^T C + A_r^T A_r Q_i B)) \) which is equivalent to maximizing \( \text{tr}(B(C^T A + B^T Q_i^T (r^2 I - A^T A))Q_{i+1}) \).

This is actually an Orthogonal Procrustes Problem. As in Chapter 2.1, we may take the singular value decomposition, \( U \Sigma V^T = B(C^T A + B^T Q_i^T (r^2 I - A^T A)) \) and let \( Q_{i+1} = VU^T \) which is the optimal solution of \( \min F(Q_{i+1}, Q_i) \).

Kercheval expresses this in terms of a mapping \( T \) as follows([15]) .

**Definition 2** Choose \( r \) as above. Define \( T : O(n) \rightarrow O(n) \) by

\[ T(O) = \arg\min_Q (\| AQB - C \|^2 + \| A_r Q B - A_r OB \|^2) \].

That is, for any \( O \in O(n) \), \( T(O) \) is the unique global minimizer of the function \( g(\cdot; O) : O(n) \rightarrow R \) defined by

\[ g(Q; O) = \| AQB - C \|^2 + \| A_r Q B - A_r OB \|^2 \].

29
This $g$ is a majorizing function for $f$, and the mapping $T$ describes the iteration determined by the majorization algorithm in this case. Using the method described in Section 6, it is straightforward to justify the following formula for $T$.

**Proposition 12** For $T$ as defined above, and $O \in O(n)$, let $U\Sigma V^T$ be the singular value decomposition of
\[ B(C^T + B^T O^T (rI - A^2)). \]
Then
\[ T(O) = VU^T. \]

It is not difficult to verify the following property of $T$.

**Proposition 13** $T$ decreases the objective function $f(Q) = \|AQB - C\|$. Under the assumption of $A_r$ and $B$ invertible, away from fixed points, $T$ strictly decreases $f(Q)$.

Proof: For any $Q$,
\[
\|AT(Q)B - C\|^2 \leq \|AT(Q)B - C\|^2 + \|A_rT(Q)B - A_rQB\|^2 \\
\leq \|AQB - C\|^2 + \|A_rQB - A_rQB\|^2 \\
= \|AQB - C\|^2
\]
The inequality holds strictly when $T(Q) \neq Q$ since $A$ and $B$ are invertible.

The majorization iteration is then equivalent to iteration of this mapping $T$ until the objective function no longer decreases by more than a preselected tolerance. In fact, the following proposition establishes the convergence of the sequence in $O(n)$.

**Proposition 14** For any $Q \in O(n)$, the sequence $T^i(Q)$ converges to a limit in $O(n)$.

Proof: As in the proof of Lemma 7.3, we have
\[
\|AT(Q)B - C\|^2 + \|A_rT(Q)B - A_rQB\|^2 \leq \|AQB - C\|^2,
\]
and hence
\[
\|A_rT(Q)B - A_rQB\|^2 \leq \|AQB - C\|^2 - \|AT(Q)B - C\|^2.
\]
Therefore, for all $n$,
\[
\sum_{i=0}^{n} \|A_rT^{i+1}(Q)B - A_rT^i(O)B\|^2 \\
\leq \sum_{i=0}^{n} (\|AT^i(Q)B - C\|^2 - \|AT^{i+1}(O)B - C\|^2) \\
= \|AQ_B - C\|^2 - \|AT^{n+1}(Q)B - C\|^2 \\
\leq \|AQ_B - C\|^2.
\]

Therefore the infinite sum is convergent. Under the assumption that $A_r$ and $B$ have full rank, this also implies
\[
\sum_{i=0}^{\infty} \|T^{i+1}(Q) - T^i(Q)\|^2 < \infty \tag{4.11}
\]

By compactness, the sequence $T^i(O)$ must have a limit point $O^* \in O(n)$. By (8) and the triangle inequality, this limit point must be unique; hence $O^*$ must be the limit of the convergent sequence $T^i(O)$.

The mapping $T$ is not continuous because the singular value decomposition is not continuous in the data. However, it is continuous except on a set of codimension 1, and hence almost everywhere. Thus, typically the limit $O^*$ will be a point of continuity of $T$, in which case it must then be a fixed point of $T$.

Hence we have the following algorithm:

**Algorithm 2**

- Choose a random orthogonal matrix $Q$. Set $r^2 = 1.1 \cdot \max(eig(A))$.
  - Set $Q_1 = Q$ and $Z = B(C^T A + B^T Q_1^T (r^2 I - A^T A))$.
  - Take the SVD of $Z$, let $Z = U\Sigma V^T$ and $Q = VU^T$.
  - If $\|Q - Q_1\|$ is very small, stop; otherwise let $Q_1 = Q$ and repeat above steps.

These algorithms can guarantee the convergence to a local minimum almost surely but not necessarily a global one.
4.3 Comparison of Numerical Results

The comparison of different algorithms for optimizing the same criterion is a legitimate activity. We will focus on speed, accuracy and ability to find global minima rather than local minima here.

Table 4.1 lists the average convergence speed of two different methods. The programs are written in Matlab and ran on an laptop with AMD 3000+ 789M processor. Both methods were set with the $10^{-4}$ precision. We pick the starting points randomly and run each experiment 200 times then take the average time. It is obvious that the gradient method is significantly slower than the majorization method.

Table 4.1: Convergence speed (in seconds) of two methods for different size of matrices.

<table>
<thead>
<tr>
<th>Size of A</th>
<th>Size of B and C</th>
<th>Majorization Method</th>
<th>Gradient Flow Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>$3 \times 3$</td>
<td>$3 \times 3$</td>
<td>0.0195</td>
<td>0.142</td>
</tr>
<tr>
<td>$23 \times 23$</td>
<td>$23 \times 707$</td>
<td>11.7</td>
<td>82.4</td>
</tr>
<tr>
<td>$24 \times 24$</td>
<td>$24 \times 706$</td>
<td>32.4</td>
<td>154.3</td>
</tr>
<tr>
<td>$27 \times 27$</td>
<td>$27 \times 703$</td>
<td>47.2</td>
<td>184.5</td>
</tr>
<tr>
<td>$65 \times 65$</td>
<td>$65 \times 665$</td>
<td>30.1</td>
<td>226.9</td>
</tr>
</tbody>
</table>

Example 1 Let us set $A = \text{diag}[4 \ 2 \ 1], B = \text{diag}[0.7 \ 0.6 \ 0]$ and

$$C = \begin{pmatrix} 0.1746 & -0.5883 & 0.1139 \\ -0.1867 & 2.1832 & 1.0668 \\ 0.7258 & -0.1364 & 0.0593 \end{pmatrix}.$$ 

On $O(3)$, $f(Q) = \|AQB - C\|^2$ has a global minimum at

$$Q_{global} = \begin{pmatrix} 0.0577 & -0.2006 & -0.9780 \\ -0.0472 & 0.9780 & -0.2033 \\ 0.9972 & 0.0579 & 0.0470 \end{pmatrix}$$

which corresponds to the objective function value 1.4934 and a local minimum point at

$$Q_{local} = \begin{pmatrix} 0.0672 & -0.2003 & 0.9774 \\ -0.1288 & 0.9697 & 0.2075 \\ -0.9894 & -0.1398 & 0.0393 \end{pmatrix}.$$
which corresponds to the objective function value 2.0544.

We tried 2000 random starting points (Table 4.2). For majorization method, all starting points converge. 860 out of 2000 (43%) converge to the global minimum point $Q_{\text{global}}$. The other 57% starting points converge to the local minimum point $Q_{\text{local}}$. For the gradient flow method, all starting points converge too. 62.5% converge to the global minimum point $Q_{\text{global}}$ and 37.5% converge to the local minimum point $Q_{\text{local}}$.

Table 4.2: Convergence of two methods for 2000 random staring matrices.

<table>
<thead>
<tr>
<th>Minimized Function Value</th>
<th>Majorization Method</th>
<th>Gradient Flow Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.49</td>
<td>43%</td>
<td>62.5%</td>
</tr>
<tr>
<td>2.05</td>
<td>57%</td>
<td>37.5%</td>
</tr>
</tbody>
</table>

Example 2 Let us keep $A$ and $C$ unchanged and set a different $B$. i.e. $A = \text{diag}[4 \ 2 \ 1]$, $B = \text{diag}[3 \ 2 \ 0.8]$ and

$$C = \begin{pmatrix} 0.1746 & -0.5883 & 0.1139 \\ -0.1867 & 2.1832 & 1.0668 \\ 0.7258 & -0.1364 & 0.0593 \end{pmatrix}. $$

We run the program of majorization method 2000 times. Table 4.3 lists the record of objective function values $f(Q) = \|AQB - C\|^2$ at critical points. The minimum function value 4.32 corresponds to the unique global minimum point

$$Q_{\text{global}} = \begin{pmatrix} 0.0163 & -0.1295 & 0.9914 \\ -0.0347 & 0.9909 & 0.1300 \\ 0.9993 & 0.0365 & -0.0116 \end{pmatrix}. $$

From Table 4.3 we see the majorization method converges to 3 critical points while the project gradient method converges to 5 critical points. Both methods could converge to the global minimum point $Q_{\text{global}}$.

For the majorization method, 726 out of 2000 (36.3%) starting points converge to the global minimum $Q_{\text{min}}$. 590 out of 2000 starting points of gradient methods converge to that point. Both methods couldn’t guarantee convergence to all critical points.

One problem the needs our attention when we use the majorization method is the choice of $r$. Theoretically all $r$ no less than the square root of largest eigenvalue (i.e. 2-norm) of $A$. 

}\]
Table 4.3: Convergence of two methods for 2000 random staring matrices.

<table>
<thead>
<tr>
<th>minimized function value</th>
<th>Majorization Method</th>
<th>Gradient Flow Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.32</td>
<td>36.3%</td>
<td>29.5%</td>
</tr>
<tr>
<td>4.56</td>
<td>0</td>
<td>28.35%</td>
</tr>
<tr>
<td>5.23</td>
<td>30.8%</td>
<td>8.35%</td>
</tr>
<tr>
<td>5.42</td>
<td>0</td>
<td>7%</td>
</tr>
<tr>
<td>7.38</td>
<td>0</td>
<td>26.8%</td>
</tr>
<tr>
<td>7.95</td>
<td>32.9%</td>
<td>0</td>
</tr>
</tbody>
</table>

could work, but in practice, some $r$ would omit a lot of critical points. We pick $r$ 1.01 times the 2-norm of $A$ in our experiments and that works well, however 1.1 times would make the algorithm fail to converge to the global minimum in our second example. Taking $r$ too close to 2-norm of $A$ could make the algorithm fail too. We don’t know yet how to optimize this $r$ to make the algorithm most efficient.

The majorization method is significantly superior in speed to the gradient flow method, it also has very good accuracy as does the project gradient method.

**Example 3** We set $C = 0$ now and leave $A$ and $B$ unchanged from last example i.e. $A = diag[4 \ 2 \ 1], B = diag[3 \ 2 \ 0.8]$  

As we analyzed in Chapter 3, there are $2^3 3! = 48$ critical points and 8 among those are global minimum points. It is also easy to write the global minimum points from 11:

\[
Q = \begin{pmatrix} 0 & 0 & \pm 1 \\ 0 & \pm 1 & 0 \\ \pm 1 & 0 & 0 \end{pmatrix}.
\]

We run two local minimization methods 2000 times from random starting points. Table 4.4 lists the convergence results of two methods. All 2000 starting points lead to a global minimum.
Table 4.4: Convergence of two methods for 2000 random starting matrices.

<table>
<thead>
<tr>
<th>Global Minimum Point</th>
<th>Majorization Method</th>
<th>Gradient Flow Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Q_1$</td>
<td>1505</td>
<td>3</td>
</tr>
<tr>
<td>$Q_2$</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>$Q_3$</td>
<td>435</td>
<td>26</td>
</tr>
<tr>
<td>$Q_4$</td>
<td>60</td>
<td>501</td>
</tr>
<tr>
<td>$Q_5$</td>
<td>0</td>
<td>520</td>
</tr>
<tr>
<td>$Q_6$</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>$Q_7$</td>
<td>0</td>
<td>501</td>
</tr>
<tr>
<td>$Q_8$</td>
<td>0</td>
<td>430</td>
</tr>
<tr>
<td>time(s)</td>
<td>38</td>
<td>223</td>
</tr>
</tbody>
</table>
All the previous methods can only find local minima. Our goal is to find the global minimum of the objective function. To do this, let us review the global optimization techniques.

## 5.1 Global Optimization Review

The task of global optimization is to find a solution in the solution set for which the objective function obtains its smallest value, the global minimum. Global optimization thus aims at determining not just "a local minimum" but "the smallest local minimum" with respect to the solution set.

Several classifications of global optimization exist. We will follow the classification given by Törn and Žilinskas ([25]). They separate the techniques into those with a guaranteed accuracy in reaching the global minimum (also named the exact method), and those without. The techniques without guaranteed accuracy are separated into direct methods that use only local information and indirect methods that build a model of level sets of the objective function. This gives the following classification([25]):

- **Methods with guaranteed accuracy (Covering methods)**
  - Branch and Bound
  - Interval Analysis
  - Dynamic Programming
- **Direct Methods**
  - Random Search Methods
Among all these methods, we choose the tunneling method to do our global optimization. Let’s have a quick review why we don’t select other methods.

The methods with guaranteed accuracy (covering methods) are regarded as deterministic since they have guaranteed success. A simple example is complete enumeration in combinatorial problems. If we don’t know much information about the bounds of the function on each subinterval or the function is high dimensional, covering methods can’t handle our problem well because of their large size.

The methods without guaranteed accuracy are divided into two categories: (i) Direct methods utilizing only local information and (ii) Indirect methods where local information is used to build a global model of the level set or objective function.

Random search methods are easy to implement but most times are very inefficient. Clustering methods are relatively more efficient than random search methods but may suffer from bad initialization or difficulty of choice of clusters.

Generalized descent methods are applied to find a lower minimum. Trajectory methods try to model the stationary points (local minima, local maxima and saddle points) by a curve in an extra parametrization. Following the trajectory of the curve would lead to all
local minima, including the global one. These methods are commonly not regarded as being promising. Penalty methods prevent returning to local minima found in precious iterations. One such penalty method is the tunneling algorithm, that alternates a local search with a tunneling step. In the latter step we try to find a solution different from the precious local minimum, that has the same function value as the previous local minimum. An important and attractive feature of the tunneling algorithm is that successive local minima are always lower. We will return to this method in more detail in the next section.

The first type of indirect models aim at approximating the level sets which are not very promising. The last type is formed by methods approximating the objective function which are based on statistical model of the function. This class of methods can be used for functions that are expensive to evaluate.

5.2 Tunneling Method

Iterative methods generally use only local information to continue their trajectory of successive improvements. Therefore a major challenge is how to get away from a given stationary point since for those local methods, once we fall into the domain of attraction of a local minimum, all local information drives us back to the same point. To solve this challenge we introduce the concept of movable pole to destroy the local minimum. By the process of tunneling, we will get a descending series of local minima.

Let $f(x)$ be a twice continuously differentiable function on the set $\Omega = \{x \in \mathbb{R}^n : a \leq x \leq b\}$, with $a$ and $b \in \mathbb{R}^n$. We will assume that all the minima of $f(x)$ are isolated and there is a finite number of them.

The tunneling method for global optimization is due to Levy and Gomez ([18]) who suggested a procedure with two alternating phases:

1. **Minimization Phase.** Given a starting point $x_0$ we use any minimization algorithm to find a local minimizer of $f(x)$, say, $x^*$. We will assume that at the end of this phase a local minimizer has been found.

2. **Tunneling Phase.** We start this phase at $x^*$ which is the exit point of the minimization phase. The purpose is to find a point $x_0 \in \Omega$ such that

   $$f(x_0) \leq f(x^*), x_0 \neq x^*$$
This can be formally stated as follows: find \( x_0 \in Z = \{ x \in \Omega - \{ x^* \} : f(x) \leq f(x^*) \} \).

When \( Z \) is not empty, we have the following two cases:

(a) \( x_0 \) is a different local minimizer of \( f(x) \) from \( x^* \);

(b) \( x_0 \) is not a minimizer of \( f(x) \). In this case a further descent function value can be achieved. We may run our minimization phase again to get to a smaller minimum.

Both cases will lead us to a minimizer with a lower function value than \( f(x^*) \).

In order to move from \( x^* \), consider the tunneling function

\[
T(x) = \frac{f(x) - f(x^*)}{[(x - x^*)^T(x - x^*)]^{\eta}}.
\] (5.1)

which has a pole at \( x^* \) for \( \eta \) sufficiently large. Note that all \( x_0 \neq x^* \) satisfying \( T(x_0) \leq 0 \) are contained in \( Z \). Therefore, we will consider the problem of finding a nonpositive minimum of \( T(x) \).

In practice, after several applications of the two phases the function \( T(x) \) will have the general form

\[
T(x) = \frac{f(x) - f^*}{\prod_{i=1}^{I} [(x - x^*_i)^T(x - x^*_i)]^{\eta_i}[(x - x_m)^T(x - x_m)]^{\lambda_0}}.
\] (5.2)
The purpose of each of the terms in (5.2) is the following: The difference in the numerator eliminates as possible solutions of this phase all those points $x$ satisfying $f(x) > f^*$. The first term in the denominator prevents the algorithm from locating as solutions of this phase all previous minimizers found at $x_i^*, i = 1, 2, \ldots, l$, with a function value $f(x_1^*) = f(x_2^*) = \ldots = f(x_l^*) = f^*$; that is, the algorithm will not cycle between previous solutions with the same function value ($l$ is always taken as one whenever the new minimizer found produces a function value strictly lower than the previous one, and is increased by one if the new minimizer produces a function value equal to the previous one). The second term in the denominator of (5.2) is designed to smooth out, in an adaptive fashion, any irrelevant local minimizer of $T(x)$ that might attract any particular minimization algorithm during the search for $x_0$ (points where $T_x(x) = 0$ and $T(x) = 0$).

In short, the tunneling function $T(x)$, which controls this process, has a numerator that adds an intercept to the objective function $f(x)$ so that the current local minimum would obtain a value of zero, and a denominator that introduces a pole at stationary points. The pole is a multiplicative penalty function that should act locally: it serves to make selected points unattractive by lifting the function surface in their immediate neighborhood.

The difficulty we need to face in this tunneling phase is the problem of pole strength. The idea of a pole is actually used for two purposes: first, it is used to cancel out the previous local minimum; second, to cancel out irrelevant local minima of the tunneling function so that the radius of convergence of the zero finder is extended. However, to actually achieve cancellation of these points of attraction, the pole has to be strong enough. This can be accomplished either by raising the power of denominator to some $r > 1$ or taking the $r$th root of the numerator. But if the pole is made too strong, the effect of the denominator is no long local, meaning that $T(x) \to 0$ regardless of what the numerator is when $x$ is far from $x^*$.

We will not go deep into the construction of tunneling function. Our function is quadratic. We may just choose $r = 1.5$. For detail, please see [10] and [18].

The tunneling phase needs a root-finding algorithm. Our objective function $f(Q) = \|AQB - C\|^2$ is quadratic in $Q$. We will adopt Newton’s method here.
5.3 Newton’s Method

Newton’s root-finding method for a function of one variable $f(x)$ consists of choosing the approximation of the root by the iteration $x_1 = x_0 + \frac{f(x)}{f'(x)}$ where $x_0$ is the old estimate. The step length $\Delta x = x_1 - x_0$ and the direction $f'(x)$ satisfy the formula $f'(x) \cdot \Delta x = -f(x)$.

Let us generalize Newton’s method to the case of a function of multi-variables $f(X)$, where the shape of $X$ can be disregarded. We may call it a point or a vector since the space of $m \times n$ matrices is isomorphic to a vector space $\mathbb{R}^{mn}$. Newton’s method works with a displacement vector $X = X_1 - X_0$ that is chosen concurrent with the direction of greatest change, $\nabla f(X_0)$, the vector of partial derivative evaluated at $X_0$.

Let $M(n, m)$ denote the space of real $n \times m$ matrices with $n$ rows and $m$ columns. An element of $M(n, 1)$, that is, a column vector, is denoted with a boldface lowercase letter $\mathbf{x}$, while $\mathbf{x}^T$ denotes its transpose row vector. An element of $M(1, 1)$ is a scalar. All functions are assumed to be of differentiability class $C^1$ unless otherwise noted.

Because the space $M(n, 1)$ is identified with the Euclidean space $\mathbb{R}^n$ and $M(1, 1)$ is identified with $\mathbb{R}$, the notation developed here can accommodate the usual operations of vector calculus.

The gradient of a scalar function $f : \mathbb{R}^n \to \mathbb{R}$

$$\nabla f(x) = \left[ \frac{\partial f}{\partial x_1} \cdots \frac{\partial f}{\partial x_m} \right].$$

The directional derivative of $f$ in the direction of $\mathbf{v}$ is then

$$\nabla_{\mathbf{v}} f = \frac{\partial f}{\partial \mathbf{x}} \mathbf{v}.$$

The differential of a function $g : \mathbb{R}^m \to \mathbb{R}^n$ is described by the Jacobian matrix

$$\frac{\partial g}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \cdots & \frac{\partial g_1}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_n}{\partial x_1} & \cdots & \frac{\partial g_n}{\partial x_m} \end{bmatrix}.$$  

The differential of $g$ along a vector $\mathbf{v}$ in $\mathbb{R}^m$ is

$$d g(\mathbf{v}) = \frac{\partial g}{\partial \mathbf{x}} \mathbf{v}.$$

For the purposes of defining derivatives of simple functions, not many changes with matrix spaces; the space of $n \times m$ matrices is after all isomorphic as a vector space to $\mathbb{R}^{nm}$. The
three derivatives familiar from vector calculus have close analogues here, though beware of the complications that arise in the identities below.

The gradient of a scalar function \( f : M(n, m) \to \mathbb{R} \)

\[
\frac{\partial f}{\partial X} = \left[ \begin{array}{ccc}
\frac{\partial f}{\partial X_{1,1}} & \cdots & \frac{\partial f}{\partial X_{1,m}} \\
\vdots & \ddots & \vdots \\
\frac{\partial f}{\partial X_{n,1}} & \cdots & \frac{\partial f}{\partial X_{n,m}}
\end{array} \right].
\]

Notice that the indexing of the gradient with respect to \( X \) is transposed as compared with the indexing of \( X \). The directional derivative of \( f \) in the direction of matrix \( Y \) is given by ([19])

\[
\nabla_Y f = \text{tr} \left( \frac{\partial f}{\partial X} Y \right),
\]

where \( \text{tr} \) denotes the trace.

The differential or the matrix derivative of a function \( F : M(n, m) \to M(p, q) \) is an element of \( M(p, q) \otimes M(m, n) \), a fourth rank tensor (the reversal of \( m \) and \( n \) here indicates the dual space of \( M(n, m) \)). In short it is an \( m \times n \) matrix each of whose entries is a \( p \times q \) matrix.

\[
\frac{\partial F}{\partial X} = \left[ \begin{array}{ccc}
\frac{\partial F}{\partial X_{1,1}} & \cdots & \frac{\partial F}{\partial X_{1,m}} \\
\vdots & \ddots & \vdots \\
\frac{\partial F}{\partial X_{n,1}} & \cdots & \frac{\partial F}{\partial X_{n,m}}
\end{array} \right],
\]

and note that each \( \frac{\partial F}{\partial X_{i,j}} \) is a \( p \times q \) matrix defined as above. Note also that this matrix has its indexing transposed; \( m \) rows and \( n \) columns. The derivative of \( F \) along an \( n \times m \) matrix \( Y \) in \( M(n, m) \) is then

\[
dF(Y) = \text{tr} \left( \frac{\partial F}{\partial X} Y \right).
\]

Note that this definition encompasses all of the preceding definitions as special cases.

Analogously to the condition on the step size in the one variable case, the step length \( \Delta X_0 \) is determined by the matrix equation

\[
\text{tr}((\nabla f(X_0))^T(\Delta X_0)) = -f(X_0).
\]

From the fact that \( \text{tr}(A^T A) = \|A\|^2 \), we know

\[
\text{tr}((\nabla f(X_0))^T \nabla f(X_0)(-f(X_0)) / \|\nabla f(X_0)\|^2) = -f(X_0).
\]
If we define the iteration as
\[ X_1 = X_0 - \frac{f(X_0)}{\|\nabla f(X_0)\|^2} \nabla f(X_0). \]  \quad (5.4)

(5.3) would be satisfied.

At first sight, iteration formula (5.4) looks like a gradient method rather than a Newton’s method. We will deduce (5.4) another way and we will prove the quadratic converge rate of this iteration. To use Newton’s method on a scalar function \( f \) with a matrix variable \( X \), we want to find the root of \( f \) along the direction of greatest change \( \nabla f(X) \) which is the gradient of \( f \). Let
\[ X_{i+1} = X_i + t \cdot l. \]
where \( l \) is the searching direction.

In the direction of the gradient \( \nabla f(X) \), we have \( l = \frac{\nabla f(X)}{\|\nabla f(X)\|} \). We define \( g(t) = f(X_i + t \cdot l) \). Our objective is to find \( t \) such that \( g(t) = 0 \).

The first order Taylor’s expansion of \( g(t) \) at 0 is
\[ g(t) = g(0) + g'(0) \cdot t + o(t^2) \]

We know that \( g(0) = f(X_i) \) and \( g'(0) = \frac{\partial f}{\partial l} = \|\nabla f(X_i)\| \). Plugging these into the above equation we have
\[ 0 = f(X_i) + \|\nabla f(X_i)\| \cdot t \]
That means \( t = -\frac{f(X_i)}{\|\nabla f(X_i)\|} \), and therefore
\[ X_{i+1} = X_i + t \cdot l = X_i - \frac{f(X_i)}{\|\nabla f(X_i)\|^2} \nabla f(X_i) \] \quad (5.5)

Convergence Analysis

**Theorem 4** (5.5) converges quadratically.
Proof: The first-order Taylor expansion of \( f(X_{i+1}) \) at \( f(X_i) \) is

\[
\begin{align*}
    f(X_{i+1}) &= f(X_i) - \frac{f(X_i)}{\|\nabla f(X_i)\|^2} \nabla f(X_i) \\
    &= f(X_i) - \nabla f(X_i) \cdot \frac{f(X_i)}{\|\nabla f(X_i)\|^2} \nabla f(X_i) \\
    &= f(X_i) - \frac{f(X_i)}{\|\nabla f(X_i)\|^2} \nabla f(X_i) \cdot \nabla f(X_i) \\
    &= f(X_i) - \frac{f(X_i)}{\|\nabla f(X_i)\|^2} \cdot \|\nabla f(X_i)\|^2 \\
    &= 0.
\end{align*}
\]

so (5.5) has a quadratic convergence rate.

### 5.3.1 Numerical Algorithm

When we use Newton’s method (5.5) on root-finding of function (4.1), \( Q \) is restricted to be orthogonal. After each iteration, \( Q \) is not guaranteed be in orthogonal group \( O(n) \). We will project the new \( Q_i \) onto \( O(n) \). From what we show in §3.2, the closest orthogonal matrix to a given matrix is the orthogonal factor of its polar decomposition.

We have the following algorithm.

**Algorithm 3**

- *Take the Newton’s iteration*

\[
Q_{i+1} = Q_i - \frac{f(Q_i)}{\|\nabla f(Q_i)\|^2} \nabla f(Q_i)
\]

- *Take the polar decomposition of \( Q_{i+1} \), \( Q_{i+1} = \tilde{Q}S \)*

- *Set \( Q_{i+1} = \tilde{Q} \)*

- *Stop when \( \|Q_{i+1} - Q_i\| < \epsilon \)*

### 5.3.2 Algorithm Analysis

We applied the tunneling method to the example in Chapter 4 where we set

\[
A = \begin{pmatrix} 4 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0.8 \end{pmatrix}
\]

44
and

\[
C = \begin{pmatrix}
    0.1746 & -0.5883 & 0.1139 \\
    -0.1867 & 2.1832 & 1.0668 \\
    0.7258 & -0.1364 & 0.0593
\end{pmatrix}.
\]

To obtain the global minimum of \( f(Q) = \|AQ - C\|^2 \), first we run the majorization method to get a local minimum. By randomly selecting a starting point

\[
Q_0 = \begin{pmatrix}
    -0.5837 & 0.5841 & -0.5640 \\
    -0.1379 & -0.7559 & -0.6400 \\
    -0.8002 & -0.2958 & 0.5218
\end{pmatrix},
\]

with objective function value 11.1759, our local minimization algorithm converges to the local minimum point

\[
Q_{local} = \begin{pmatrix}
    0.0177 & -0.0916 & 0.9956 \\
    -0.2271 & -0.9701 & -0.0852 \\
    -0.9737 & 0.2246 & 0.0379
\end{pmatrix},
\]

corresponding to objective function value 7.9186.

Next we run our Newton’s algorithm to look for a new point with the objective function value no larger than 7.9186. Our tunneling function converges to point

\[
Q_{new} = \begin{pmatrix}
    -0.0372 & 0.6151 & 0.7876 \\
    -0.0332 & 0.7869 & -0.6162 \\
    0.9988 & 0.0490 & 0.0089
\end{pmatrix},
\]

with function value 6.8472.

Now take this \( Q_{new} \) as our new starting point and run the minimization algorithm again, we will arrive to the next minimum point

\[
Q_* = \begin{pmatrix}
    0.0163 & -0.1295 & 0.9914 \\
    -0.0338 & 0.9909 & 0.1300 \\
    0.9993 & 0.0356 & -0.0117
\end{pmatrix},
\]

with function value 4.3215 which is the global minimum as we discuss in §4. We run the tunneling phase again, Newton’s algorithm leads us nowhere this time.

Since we have already had an effective local minimization algorithm, what decides if the tunneling method is successful or not is the tunneling step, i.e. the efficiency of Newton’s method. It is well-known that Newton’s method works best if near the root the system is close to linear and the derivatives are bounded away from zero. Its convergence is also guaranteed, but only if it is started close enough to the root, or within an interval that satisfies certain conditions.

The drawbacks of Newton’s method are ([14]):
1. Divergence at inflection points: If the selection of a guess or an iterated value turns out to be close to the inflection point of \( f(x) \), that is, near where \( f(x) = 0 \), the roots may start to diverge away or converge slowly from the root.

2. Division of zero or near zero.

3. Root jumping: In some case where the function \( f(x) \) is oscillating and has a number of roots, one may choose an initial guess close to a root. However, the guesses may jump and converge to some other root.

4. Oscillations near local maximum and minimum: Results obtained from the Newton-Raphson method may oscillate about the local maximum or minimum without converging on a root but converging on the local maximum or minimum. Eventually, it may lead to division to a number close to zero and may diverge.

For these cases, the tunneling methods would fail due to the failure of the tunneling step. The use of tunneling functions also influences root-finding of Newton’s method.
We tested the algorithms using the same data used in [1] which was from actual equity risk factors, taken from the MSCIBarra equity risk model as of April 2001. Table 6.1 reports the number of factors in each of 22 single-country equity risk models. Combined with a 40-factor currency model, the resulting global model contains 730 factors total. The markets range from highly developed, such as UK and Japan, to emerging, such as Mexico and Taiwan. The largest market is the US, comprising 65 factors, and the smallest is the 11-factor New Zealand block. Our test data comes from factor covariances forecasts as estimated by MSCIBarra for the month of April 2001, based on a historical monthly time series of prices.

Let us look at the input of the objective function $f = \|AQB - C\|^2$. Here $A$ is the square root matrix of a new market block, the largest problems corresponds the $65 \times 65$ US block, in a $730 \times 730$ global correlation matrix. $B$ is the product of the inverse square root matrix of the corresponding old market block and an off-diagonal block, while $C$ is the off-diagonal block to be changed (Both $B$ and $C$ have size $65 \times 665$).

We have observed the same or similar objective function values from our data wherever the starting point is. This looks surprising and suspicious at first sight.

Figure 6.1 lists the singular values of $B$ and $C$ when we set $A$ as the US block. We notice that there are a lot of near zero singular values in $B$ and $C$. To understand the structure of the objective function clearly, we partition $B$ into two parts: the part with positive singular values and the part with zero or near zero singular values.

Let us take the singular value decomposition of $B$. We have $B = U_B \Sigma_B V_B^T$. Now our
The objective function is

\[ f(Q) = \|AQB - C\|^2 = \|AQU_B\Sigma_B V_B^T - C\|^2 = \|A\tilde{Q}\Sigma_B - CV_B\|^2 = \|A\tilde{Q}\Sigma_B - \tilde{C}\|^2 \]

where \(\tilde{Q} = QU_B\) and \(\tilde{C} = CV_B\).

Table 6.1: Our model comprises 730 factors in 23 markets. There is one block for each market and plus the currency block.

<table>
<thead>
<tr>
<th>Country</th>
<th>Block Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Currency</td>
<td>40</td>
</tr>
<tr>
<td>Australia</td>
<td>33</td>
</tr>
<tr>
<td>Brazil</td>
<td>29</td>
</tr>
<tr>
<td>Canada</td>
<td>30</td>
</tr>
<tr>
<td>France</td>
<td>21</td>
</tr>
<tr>
<td>Greece</td>
<td>23</td>
</tr>
<tr>
<td>Germany</td>
<td>27</td>
</tr>
<tr>
<td>Hong Kong</td>
<td>23</td>
</tr>
<tr>
<td>Indonesia</td>
<td>31</td>
</tr>
<tr>
<td>Japan</td>
<td>52</td>
</tr>
<tr>
<td>Korea</td>
<td>35</td>
</tr>
<tr>
<td>Malaysia</td>
<td>24</td>
</tr>
<tr>
<td>Mexico</td>
<td>15</td>
</tr>
<tr>
<td>Netherlands</td>
<td>15</td>
</tr>
<tr>
<td>New Zealand</td>
<td>11</td>
</tr>
<tr>
<td>South Africa</td>
<td>53</td>
</tr>
<tr>
<td>Singapore</td>
<td>27</td>
</tr>
<tr>
<td>Sweden</td>
<td>30</td>
</tr>
<tr>
<td>Switzerland</td>
<td>20</td>
</tr>
<tr>
<td>Thailand</td>
<td>41</td>
</tr>
<tr>
<td>Taiwan</td>
<td>35</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>50</td>
</tr>
<tr>
<td>United States</td>
<td>65</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>730</strong></td>
</tr>
</tbody>
</table>
Let

$$\Sigma_B = \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix}$$

where $B_1$ is the sub-matrix of $\Sigma_B$ with positive diagonals and $B_2$ is the sub-matrix with near zero diagonals.

Let us partition $\tilde{Q}$ into $\tilde{Q} = (Q_1|Q_2)$ such that the row of $Q_1$ has same size with the column of $B_1$. We also split $\tilde{C}$ into $\tilde{C} = (C_1|C_2)$ with compatible sizes. Thus we have

$$f(Q) = \|A\tilde{Q}\Sigma_B - \tilde{C}\|^2$$

$$= \|A(Q_1|Q_2) \begin{pmatrix} B_1 \\ 0 \end{pmatrix} - (C_1|C_2)\|^2$$

$$= \|(AQ_1B_1|AQ_2B_2) - (C_1|C_2)\|^2$$

$$= \|AQ_1B_1 - C_1\|^2 + \|AQ_2B_2 - C_2\|^2$$

From the split of $B$, we know $AQ_2B_2$ is very close to zero, so $\|AQ_2B_2 - C_2\|^2$ which is close to $\|C_2\|^2$. $Q_1$, whose number of columns is the number of non-zero singular values of $B$ determines the minimum of $f(Q)$.

This structure can explain the similarity of the objective function values from our data. We tested the matrices $Q_1$ of all those local maxima and found out the difference of the
objective function value of different $Q_1$s is negligible! That means any local minimum point $Q$ has a free $Q_2$ sub-matrix which leads to the same objective function values. In our case, 38 out of 65 singular values of $B$ are close to 0, that means any local minimum $Q$ has a degree of freedom $\frac{65 \times 64}{2} - \frac{38 \times 37}{2} = 1729!$

Anderson et al. ([1]) select a map $g$ from $\mathbb{R}^N$ to $O(n)$ and then to minimize the composition of the objective function $f(Q) = \|AQB - C\|^2$ with this map $g$. They denote by $R_{i,j}(\theta)$ the rotation by angle $\theta$ in the $(i,j)$ plane for $1 \leq i < j \leq n$. This rotation transforms the $s_i$ and $s_j$ factors according to $s_i \rightarrow s_i \cos \theta + s_j \sin \theta$ and $s_j \rightarrow s_j \cos \theta - s_i \sin \theta$, and leaves other factors unchanged. Based on the fact that every element of $O(n)$ with determinant 1 may be expressed as a product of $N$ rotations,

$$g(\theta_1, \ldots, \theta_N) \equiv R_{1,2}(\theta_1)R_{1,3}(\theta_2)\ldots R_{1,n}(\theta_n)\ldots R_{n-1,n}(\theta_N).$$

for some angles $\theta_1, \ldots, \theta_N$. Here $N = n(n - 1)/2$.

With $g$ in hand, the objective function can be expressed as a sum of squares of sine and cosine functions. This lends itself to the Levenberg-Marquardt numerical method because the partial derivatives of the function of $g(\theta_1, \ldots, \theta_N)$ are easily computed. Their experiments showed that implementation in $C++$ on a unix work station, $20 \times 20$ blocks took over an hour and the $65 \times 65$ problem did not converge before the experiments gave up after several hours (Kercheval([15])).

The majorization method works much better than the above two methods. A simple MATLAB implementation of the iteration on an inexpensive laptop took about 3 minutes to converge to within $1e-6$ for each starting value of $Q$.

The tunneling method works poorly on this problem. That is not surprising because of the distribution structure of the local minima — there are too many minimum points with close objective function values. For this type of problem, Newton’s method does not work well. In one of our experiments, it took about 90 hours for the tunneling step to find the next starting points. In our other example of same size $B$ with all positive singular values, it only took several minutes to converge.

For the purpose of comparison, we redo the experiments done in Anderson et al. ([1]). We still choose the Barra global equity risk model, essentially as it was constituted in 2001. This model covers 23 markets and has 730 factors, as described in Table 6.1. The markets range from highly developed, such as UK and Japan, to emerging, such as Mexico and Taiwan.
The largest market is the US, comprising 65 factors, and the smallest is the 11-factor New Zealand block. Our test data comes from factor correlation forecasts as estimated by Barra for the month of April 2001, based on a historical monthly time series of prices.

The inputs to our study are a ‘first draft’ of a \( 730 \times 730 \) factor covariance matrix, estimated by a Barra method using global factors, and a collection of preferred single market blocks that have been estimated with separate models tuned to their individual markets.

We examine the result of revising a single market block, comparing

1. the default simple revising transformation using \( Q = I \).

2. the optimized transformation in which the choice of \( Q \) is obtained by minimizing the objective function \( \|AQB - C\| \) through tunneling method.

3. the optimized solution obtained by Anderson et al.[1]

We can see from Figure 6.2 that there is a significant improvement in the use of an optimized revising transformation as compared to the default, generally by a factor of two to four. Typical average correlations change by around 0.05 with the default transformation \( (O = I) \), while the optimized transformation leads to average correlation changes of around 0.02.
We run the tunneling method on those blocks in Figure 6.2. We may observe that the apparent improvement on the Greece block which has lower RMSE than the optimized solution based on Anderson Et al.’s method. We did not see much improvement on other blocks but certainly we don’t see worse results. Based on our analysis about the many close objective function values we observe from the US block, this is not surprising. We calculate the corresponding $B$ and $C$ for each different and we see similar pattern — some near zero eigenvalues. Figure 6.3 gives a typical pattern of $B$ and $C$ from replacing Hong Kong market matrix.
7.1 Majorization Algorithm Code

% function majorization, returns optimized Q.

function Q = majorize(A,B,C)

clear;

format long;

load A B C

[Q, R0]=qr(rand(size(A,1)));

%Q0=Q;
Q1=Q;

%norm(A*Q*B-C,'fro')

r = 1.01*norm(A, 2)^2;

A0 = (r*eye(size(A))-A'*A)^(1/2);

Z =B*(C'*A + B'*Q1'*(r*eye(size(A))-A'*A));
$$[U, S, V] = \text{svd}(Z);$$

$$Q = V*U'; \quad \%\text{optimized solution based on OPP}$$

$$i=1;$$

$$\text{tic}$$

$$\text{while norm}(A0*Q*B-A0*Q1*B,'fro')>1e-4$$

$$\quad Q1=Q;$$

$$\quad Z = B*(C'*A + B'*Q1'*(r*eye(size(A))-A'*A));$$

$$\quad \%\text{norm}(Z,'fro');$$

$$\quad [U,S,V] = \text{svd}(Z);$$

$$\quad Q = V*U';$$

$$\quad i = i+1;$$

$$\quad \text{norm}(Q1-Q,'fro');$$

$$\quad \text{objectivePrev} = \text{norm}(A*Q1*B - C,'fro');$$

$$\quad \text{objectiveNext} = \text{norm}(A*Q*B-C, 'fro');$$

$$\quad \text{objectiveDiff} = \text{objectivePrev} - \text{objectiveNext};$$

$$\quad \text{normQDiff} = \text{norm}(Q-Q1);$$

$$\text{if(mod(i,500) == 0)$$

$$\quad \text{objectivePrev}$$

$$\quad \text{objectiveNext}$$

$$\quad \text{objectiveDiff}$$

$$\quad \text{normQDiff}$$

$$\quad i$$

$$\text{end}$$

$$\text{end}$$

$$\text{toc}$$

54
\[ \text{normAQB_AQ1B} = \text{norm}(A*Q*B-A*Q1*B,'fro') \]

\[ m = \text{norm}(A*Q*B-C,'fro'); \]
\[ \text{RMS} = m/((\text{size}(C,1) \times \text{size}(C,2)))^{(1/2)} \]

\[ \text{RMSId} = \text{norm}(A*B-C,'fro')/((\text{size}(C,1) \times \text{size}(C,2)))^{(1/2)} \%Q=I \]

\[ [UA,SA,VA] = \text{svd}(A); \text{sa} = \text{diag}(SA); \]
\[ [UB,SB,VB] = \text{svd}(B); \text{sb} = \text{diag}(SB); \]
\[ [UC,SC,VC] = \text{svd}(C); \text{sc} = \text{diag}(SC); \]

### 7.2 Pseudo-code of Gradient-flow Code base on Chu and Trendafilov’s Method

```plaintext
global md n global A B C In
load A, B, C
n = size(A);
[n,md]=size(C);
In=eye(n);
```

55
\[ [Q_0, R] = \text{qr}(\text{rand}(n)) \];

\% B = A * Q_0 * C;

\[ [Q_{\text{in}}, R] = \text{qr}(\text{rand}(n), 0) \];

Q_{\text{in}}

\[ [Q_{\text{out}}, \text{time, } \text{mm}] = \text{solver}(Q_{\text{in}}), A * Q_{\text{out}} * C; \]

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function \[ [Q_{\text{last}}, \text{timeout}] = \text{solver}(Q); \]

global md n
global A B C Ip Iq
tstart = 0; tstep = 50; tfinal = 2000000; tol = 1.e-06;
options = \text{odeset}('AbsTol', tol, 'RelTol', 1.e-04);
if tstop < tfinal
    timeout = 0; counter = 0;
tstop = tstart + tstep;
tspan = tstart:(tstop - tstart)/10:tstop;
timei = \text{cputime};
    [tout, yout] = \text{ode15s}('flow', tspan, y0, options);
else
    return
end

timelocal = \text{cputime} - timei;
timeout = timeout + timelocal;
Q_{\text{last}} = Q;
7.3 Tunneling Method Code

global A B C global norm_localmin Q_local

Q_local = majorize(A,B,C)

[Q3, R3] = qr(rand(size(A,1)));

Q2 = zeros(size(A,1),size(A,1));

[g, gprime] = tunneling(Q3);

j=1;

Q2 = Q_local - g*gprime/norm(gprime,'fro')^2;

fQdiff = norm(A*Q3*B-C,'fro')-norm_localmin;

tic

while fQdiff > 1e-2
    Q3 = Q3 - g*gprime/norm(gprime,'fro')^2;
    [U2,S2,V2] = svd(Q2);
    Q2 = U2*V2';
    Q3 = Q2;
    [g, gprime] = tunneling(Q3);
    fQdiff = norm(A*Q3*B-C,'fro')-norm_localmin
    j = j+1;
    if ( mod(j,500) == 0)
        fQdiff = norm(A*Q3*B-C,'fro')-norm_localmin
        j
    end
function [tunfn, tunfn_grad] = tunneling(Q)

global A B C global norm_localmin Q_local

tunfn=(norm(A*Q*B-C,'fro')^2-norm_localmin^2)*exp(1.5/norm(Q-Q_local,'fro'));

tunfn_grad =
[2*(A'*A*Q*B*B'-A'*C*B')*norm(Q-Q_local,'fro')^2+2*(norm(A*Q*B-C,'fro')^2-norm_localmin^2)
58
}]
CHAPTER 8

CONCLUSION

We have developed a numerically efficient way to adjust a multi-market covariance matrix forecast by correcting one or more diagonal blocks without loss of positive semi-definiteness, and to do so in a way that does not disturb the factors in the other blocks and approximately minimizes the disturbance to the cross-block covariances. In practice, this will allow a global risk forecast to be made consistent with potentially more refined individual market forecasts corresponding to diagonal blocks of the global covariance matrix. There remains a need for better theoretical understanding of the structure of our objective function. The proposed method still only finds local minima, though in practice we observe that the lowest local minima have the largest basins, and so are the most likely to be found with this method. We feel the special structure of the problem has not yet been fully exploited, leaving open the possibility of a fast method that guarantees a fast approach to the global minimum. Progress on the Double Orthogonal Procrustes Problem would be of interest in a variety of fields where this data-fitting context arises.

To forecast the total risk, we incorporate one or more refined sub-market risk models into a larger but coarser total risk model, without violating positive semi-definiteness of the resulting covariance matrix. Based on the model suggested in ([1]), we need to solve a weighted orthogonal procrustes problem. We have addressed two questions in this paper. First, what is an efficient way to solve the weighted orthogonal procrustes problem and what is the structure of the objective function? Given that the global matrix has an order of several hundreds. Second, can we find the global minimum solution to the problem? The first question is addressed by means of geometric analysis of the objective function and comparison of numerical solution of several methods. We answer the second question by using a penalty method in optimization. The optimization problem, as we have framed it,
is unconstrained, but is typically large and non-convex. However, the true global minimum has no special properties. In our experiments, 80% could lead us to a global minimum.

We still need to have a better theoretical understanding of the objective function. We have known the critical number of a special case, but what is this number for the general case? Could we get a bound of the global minimum for given $A, B$ and $C$?
REFERENCES


BIOGRAPHICAL SKETCH

Jianke Zhang

Jianke Zhang was born in Qingdao, China in 1975. He completed his Bachelor of Science degree in Applied Mathematics and Information Science at Yantai University in 1996. He obtained his Master of Science degree in Computational Mathematics and Applied Software in spring 1999 from Peking University, China. He obtained his second Master of Science degree in financial mathematics from Florida State University in spring 2004. He earned his Doctorate of Philosophy degree in financial Mathematics in spring 2007 at the Florida State University. His research interests include portfolio management, risk management and copula methods in finance. Jianke currently lives at Juno Beach, FL.