# ABSTRACT

LIN, MATTHEW MIN-HSIUNG. Inverse Problems of Matrix Data Reconstruction. (Under the direction of Moody Chu.)

Mathematical modeling is an indispensable task in almost every discipline of sciences. If a model for a specific phenomenon can be correctly established, then it empowers the practitioners to analyze, predict, and delegate an onward decision which may have important applications and consequences. However, since most of the information gathering devices or methods, including our best intellectual endeavor for understanding, have only finite bandwidth, we cannot avoid the fact that the models employed often are not exact. A constant update or modification of an existing model based on the newest information therefore is in demand. Such a task is generally referred to as an inverse problem. While in a forward problem the concern usually is to express the behavior of a certain physical system in terms of its system parameters, in the inverse problem the concern is to express the parameters in term of the behavior. This thesis addresses a small portion of the mass domain of inverse problems. The specific focus has been on matrix data reconstruction subject to some intrinsic or prescribed constraints.

The purpose of this investigation is to develop theoretic understanding and numerical algorithms for model reconstruction so that the inexactness and uncertainty are reduced while certain specific conditions are satisfied. Explained and illustrated in this thesis are some most frequently used methodologies of matrix data reconstruction so that for a given dataset, these techniques can be employed to construct or update various (known) structural properties, or to classify or purify certain (unknown) embedded characteristics. Areas of applications include, for example, the applied mechanics where systems of bodies move in response to the values of their known endogenous parameters and the medical or social sciences where the causes (variables) of the observed incidences neither are known a priori nor can be precisely quantified. All of these could be considered as an inverse problem of matrix data reconstruction. This research revolves around two specific topics – quadratic inverse eigenvalue problems and low rank approximations – and some other related problems, both in theory and in computation.

An immediate and the most straightforward application of the quadratic inverse eigenvalue problem would be the construction of a vibration system from its observed or desirable dynamical behavior. Its mathematical model is associated to the quadratic matrix polynomial $\mathfrak{Q}(\lambda) = M\lambda^2 + C\lambda + K$ whose eigenvalues and eigenvectors govern the vibrational behavoir. Tremendous complexities and difficulties in recovering coefficient matrices $M$, $C$, $K$ arise when the predetermined inner-connectivity among its elements and the mandatory nonnegativity of its parameters must be taken into account. Considerable efforts have been taken to derive theory and numerical methods for solving inverse eigenvalue problems, but techniques developed thus

far can handle the inverse problems only on a case by case basis. The first contribution in this investigation is an efficient, reliable semi-definite programming technique for inverse eigenvalues problems subject to specified spectral and structural constraints. Of particular concern is the issue of inexactness of the prescribed or measured eigeninformation, which is almost inevitable in practice, since inaccurate data will affect the solvability of this inverse eigenvalue problem. To address this issue, a second contribution in this investigation is a methodical approach by using the notion of truncated QR decomposition to first determine whether a nearby inverse problem is solvable and, if it is so, the method computes the approximate coefficient matrices while providing an estimate of the residual error. Both methods enjoy the advantages of preserving inter-connectivity structures and other important properties embedded in the original problems. More importantly, both approaches allow more flexibilities and robustness in handling highly structured problems than other special-purpose algorithms.

Low rank approximations have become increasingly important and ubiquitous in this era of information. Generally, there is no unified approach because the technique often is data type dependent. This research studies and proposes new factorization techniques for three different type of data. The first algorithm aims to perform a nonnegative matrix factorization of a nonnegative data matrix by recasting the problem geometrically as the approximation of a given polytope on the probability simplex by a simpler polytope with fewer facets. This view leads to a convenient way of decomposing the data by computing the proximity map which, in contrast to most existing algorithm where only an approximate map is used, finds the unique and global minimum per iteration. The second algorithm investigates the factorization of integer matrices which is more realistic and important in informatics. Searching through the literature, it appears that there does not exist a suitable algorithm which can handle this type of problem well owing to its discrete nature. Two effective approaches for computing integer matrix factorization are proposed in this investigation — one is based on hamming distance and the other on Euclidean distance. A lower rank approximation of a matrix $A \in \mathbb{Z}^{m\times n} \approx UV$ with factors $U \in \mathbb{Z}_2^{m\times k}$, $V \in \mathbb{Z}^{k\times n}$, where columns of $U$ are mutually exclusive and integer $k < \min\{m, n\}$ is given. The third algorithm concerns expressing a nonnegative matrix as the shortest sum of nonnegative rank one matrices, the so called nonnegative rank factorization. Till now, only a few abstract results which are somewhat too conceptual for numerical implementation have been developed in the literature. Employing the Wedderburn rank reduction formula, a numerical procedure detecting whether a nonnegative rank factorization exists is presented. In the event that such a factorization does not exist, it is able to compute the maximum nonnegative rank splitting.

This thesis includes a detailed analysis of inverse eigenvalue problems and low rank factorizations. Some of the theories are classical, but new insights are obtained and their implementation for numerical computation are developed. On the other hand, this investigation leads to quite

a few innovative algorithms which are effective and robust in tackling the otherwise very difficult inverse problems. The research is ongoing and several interesting research problems are identified in this thesis.

# Inverse Problems of Matrix Data Reconstruction

by
Matthew Min-Hsiung Lin

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Mathematics

Raleigh, North Carolina

2010

APPROVED BY:

| | |
|---|---|
| Pierre Gremaud | Alina Chertock |
| Dmitry Zenkov | Moody Chu<br>Chair of Advisory Committee |

## DEDICATION

To my dear family

for their support and love

# BIOGRAPHY

Min-Hsiung Lin received his Bachelor of Science degree in mathematics from the National Taiwan Normal University in 2003 and his Master of Science degree in applied mathematics from the National Tsing Hua University in 2005. Then, his life has turned to a new chapter. He went to the United States to pursue his Ph.D. degree in applied mathematics at the North Carolina State University. During the next four years, he received his second Master's degree in computer science in 2009, was baptised on March 29, 2009, and completed his PhD dissertation in 2010.

## ACKNOWLEDGEMENTS

Thank you God for all your many blessings.

Thank my advisor Dr. Moody Chu and his wife Mrs. Joyce Chu

- They like parents. Fill my heart with endless care and support.
- They like angels. Open my eyes. Let me have the chance to see the God.
- They like mentors. Lead me through the difficulty with profound knowledge and wisdom.
- They are interceders. Bless my road with sincere daily prayers.

Thank the members of my committee: Dr. Pierre Gremaud, Dr. Alina Chertock, Dr. Min Kang, and Dr. Dmitry Zenkov, for their informative and wonderful instruction in their courses, for their valuable comments, insights, and suggestions on my work .

Thank Dr. Dennis R. Bahler for his perpetual kind and gentle nature allowing me to finish my master degree in Computer Science.

Thank Dr. Wen-Wei Lin, Dr. Tsung-Min Hwang, Dr. Tsan-Yao Chen, Dr. Shu-Ming Chang, Dr. Bo Dong, Dr. Shih-Feng Shieh, Dr. Yuen-Cheng Kuo, and Dr. Chung-Yueh Chiang for their invaluable advice and suggestions on my work.

Thank my friends who offered me great support and precious friendships. I would like to share my happiness with them.

Thank my parents, two brothers, and Siao-Yong, who have never failed to give me their full support, whenever I need them. Without their understanding and encouragement, I would not be able to understand what I insist on, nor could I have made my way so far.

## TABLE OF CONTENTS

**List of Tables** . . . . . **viii**

**List of Figures** . . . . . **ix**

**List of Notations** . . . . . **x**

**Chapter 1 Introduction** . . . . . **1**
1.1 Overview . . . . . 1
1.2 Quadratic Inverse Eigenvalue Problems . . . . . 3
1.3 Low Rank Factorization . . . . . 5
1.4 Research Outline . . . . . 9

**I Quadratic Inverse Eigenvalue Problems** **11**

**Chapter 2 Inverse Problems with Basic Real Symmetric Models** . . . . . **12**
2.1 Overview . . . . . 12
2.2 Fundamental theorem on spectral decomposition . . . . . 14
2.2.1 Role of the parametric $\mathscr{H}$ matrix . . . . . 14
2.2.2 Structure of the parametric $\mathscr{H}$ matrix . . . . . 16
2.3 Eigenvector completion . . . . . 18
2.3.1 Issues related to numerical computation . . . . . 23
2.4 Model updating with no spill-over . . . . . 27
2.5 Role of eigenvalues . . . . . 29
2.6 Conclusion . . . . . 33

**Chapter 3 Inverse Problems with Partial Eigenpair Information** . . . . . **34**
3.1 Overview . . . . . 34
3.2 Handling exact eigenpairs . . . . . 35
3.2.1 Inherent structure . . . . . 39
3.2.2 Linear equality system . . . . . 40
3.2.3 Linear inequality system . . . . . 42
3.3 Handling inexact eigenpairs . . . . . 44
3.4 Numerical experiments . . . . . 47
3.5 Conclusion . . . . . 51

**Chapter 4 Inverse Problems with Structured Models** . . . . . **52**
4.1 Overview . . . . . 52
4.2 Semidefinite programming . . . . . 53
4.3 Models with symmetric and positive semi-definite structures . . . . . 57
4.4 Models with gyroscopic structures . . . . . 60
4.5 Models with sparsity patterns . . . . . 61

4.6 Models with prescribed entries . . . 63
4.7 Model updating problems . . . 64
4.8 Conclusion . . . 69

**II Low Rank Factorizations 71**

**Chapter 5 Nonnegative Matrix Factorization via Polytope Approximation . . 72**
5.1 Overview . . . 72
5.2 Pull-back to the probability simplex . . . 74
5.3 Polytope fitting problem . . . 77
5.3.1 Proximity map on the probability simplex . . . 79
5.3.2 Alternating direction iteration . . . 84
5.4 Application to the nonnegative matrix factorization . . . 88
5.4.1 Proximity map on the the simplicial cone . . . 90
5.4.2 Alternating direction search . . . 93
5.5 Numerical experiments . . . 97
5.6 Conclusion . . . 102

**Chapter 6 Integer Matrix Factorization via Rank-one Approximation . . . . 104**
6.1 Overview . . . 104
6.2 Basic rank-one approximation . . . 105
6.2.1 Approximation with Hamming metric . . . 106
6.2.2 Approximation with Euclidean metric . . . 110
6.3 Algorithms of integer matrix factorization . . . 111
6.3.1 Communal rule . . . 112
6.3.2 Recursive decomposition . . . 113
6.3.3 Optimal low rank approximation . . . 114
6.4 Numerical experiments . . . 118
6.4.1 Association analysis . . . 118
6.4.2 Cluster analysis . . . 120
6.4.3 Pattern discovery . . . 123
6.4.4 Random performance test . . . 125
6.5 Conclusions . . . 127

**Chapter 7 Nonnegative Rank Factorization via Rank Reduction . . . . . 129**
7.1 Overview . . . 129
7.2 Wedderburn rank reduction formula . . . 132
7.3 Nonnegative rank factorization . . . 133
7.4 Completely positive matrices . . . 138
7.5 Maximal nonnegative rank splitting . . . 139
7.6 Geometric meaning of nonnegative rank . . . 141
7.7 Numerical experiments . . . 144
7.8 Conclusion . . . 146

**Chapter 8 On the Nonnegative Rank of Euclidean Distance Matrices . . . . . 147**
8.1 Overview . . . . . 147
8.2 Rank condition and standard form . . . . . 147
8.3 Nonnegative rank factorization for linear Euclidean distance matrices . . . . . 150
8.4 A conjecture for higher dimensional Euclidean distance matrices . . . . . 156

**Chapter 9 Future Work . . . . . 158**

**References . . . . . 161**

**Appendix . . . . . 173**
Appendix A QR Factorization with Column Pivoting . . . . . 174

## LIST OF TABLES


Table 4.1 An SDP code calling routines from YALMIP to solve QIEPs for positive semi-definite $(M, C, K)$. . . . . 58
Table 4.2 Existence of positive definite $M$, $C$ and $K$ to the symmetric QIEP with random eigenstructure. . . . . 60
Table 4.3 Existence of positive definite $M$, $\mathscr{C}$, $\mathscr{K}$, and skew-symmetric $\mathscr{G}$ to the QIEP with random eigenstructure. . . . . 61

Table 6.1 An transaction example with 5 transactions and 6 items . . . . . 119
Table 6.2 A binary representation of the transaction example . . . . . 119
Table 6.3 Two representative transactions $T_1'$ and $T_2'$ . . . . . 119
Table 6.4 Precision and Recall of edible or poisonous mushrooms . . . . . 123
Table 6.5 Precision and Recall of benign or recall patients . . . . . 125
Table 6.6 Global convergence rate and average time per factorzation (in seconds) on randomly generated data sets. . . . . 127

Table 8.1 Standard nonnegative factorizations of $Q_4$. . . . . 153

## LIST OF FIGURES


Figure 2.1 Curves where the principle minors of $XJ^2H^{-1}X^\top$ vanish. . . . . 32

Figure 3.1 A four-degrees-of-freedom mass-spring system. . . . . 36
Figure 3.2 An RLC electronic network. . . . . 37

Figure 4.1 The $48 \times 48$ sparse matrices $M_0$ and $K_0$. . . . . 67
Figure 4.2 Performance of model updating for Harwell-Boeing test data BCSST*01 (n = 48). . . . . 68
Figure 4.3 Performance of model updating for Harwell-Boeing test data BCSST*02 (n = 66). . . . . 69

Figure 5.1 Convex hull of $\vartheta(A) \in \mathbb{R}^{3\times n}$ with $n = 11$. . . . . 76
Figure 5.2 Convex hull of $\vartheta(Y)$ and $U$ in $\mathcal{D}_3$. . . . . 78
Figure 5.3 NNMF in $\mathbb{R}^2$ when $p = 1$. . . . . 90
Figure 5.4 Simplicial cone of $U$ and convex hull of $\vartheta(Y) \in \mathbb{R}^{3\times n}$ with $p = 2$ and $n = 11$. . . . . 91
Figure 5.5 Triangle (polytope) enclosing a prescribed set of points on $\mathcal{D}_3$. . . . . 98
Figure 5.6 Objective values $f(U, V)$ per iteration by the Chu-Lin algorithm versus the Lee-Seung algorithm. . . . . 99
Figure 5.7 Objective values $f(U, V)$ in 20 iterations per specified rank by the Chu-Lin and the Lee-Seung algorithms. . . . . 100
Figure 5.8 Complexity of the Chu-Lin algorithm measured in CPU time per sweep of the alternating optimization. . . . . 101
Figure 5.9 80 sample images from the swimmer database. . . . . 102
Figure 5.10 17 "parts" decomposed from the swimmer database by Algorithm 5 in 10 iterations. . . . . 102

Figure 6.1 Performance of algorithms on the mushroom data set . . . . . 122
Figure 6.2 Performance of algorithms on Wisconsin breast cancer data set . . . . . 124
Figure 6.3 Basic elements recovered from the swimmer data set by the IMF. . . . . 126
Figure 6.4 Bases elements recovered from block matrix data set by the IMF. . . . . 127

Figure 7.1 A geometric representation of the matrix $\vartheta(\mathscr{C})$. . . . . 142

Figure 8.1 Euclidean matrix of three points in $\mathbb{R}^2$. . . . . 148
Figure 8.2 A geometric representation of the matrix $\vartheta(Q_4)$ when $r = 1$. . . . . 152
Figure 8.3 A geometric representation of the matrix $\vartheta(Q_4)$ when $r > 1$. . . . . 157

## LIST OF NOTATIONS

| Notation | Description |
|---|---|
| Calligraphic capitals, e.g., $\mathcal{X}$ or $\mathcal{J}$ | matrices associated to Jordan canonical forms |
| German capitals, e.g., $\mathfrak{X}$ or $\mathfrak{J}$ | complex-valued matrices of general sizes |
| Roman capitals, e.g., $M$ or $H$ | real-valued matrices of general sizes |
| Script capitals, e.g., $\mathscr{B}$ or $\mathscr{H}$ | some specially defined $2n \times 2n$ matrices |
| $\mathbb{R}$ | set of real numbers |
| $\mathbb{R}_+$ | set of nonnegative real numbers |
| $\mathbb{Z}$ | set of integer numbers |
| $\mathbb{R}^n$ | space of real column vectors of dimension $n$ |
| $\mathbb{R}_+^n$ | space of nonnegative real column vectors of dimension $n$ |
| $\mathbb{Z}^n$ | space of integer column vectors of dimension $n$ |
| $\mathbb{C}^{m\times n}$ | space of complex matrices of dimension $m \times n$ |
| $\mathbb{R}^{m\times n}$ | space of real matrices of dimension $m \times n$ |
| $\mathbb{R}_+^{m\times n}$ | space of nonnegative real matrices of dimension $m \times n$ |
| $\mathbb{Z}^{m\times n}$ | space of integer matrices of dimension $m \times n$ |
| $S\mathbb{R}^{n\times n}$ | space of symmetric real matrices of dimension $m \times n$ |
| $A \succeq 0$ | positive semidefinite matrix $A$ |
| $A \succ 0$ | positive definite matrix $A$ |
| $A^\top$ | the transpose of a matrix $A$ |
| $A^*$ | the conjugate transpose of a matrix $A$ |
| $A^{-1}$ | the inverse of a matrix $A$ |
| $\mathbf{x}^*$ | the conjugate transpose of a vector $\mathbf{x}$ |
| $\mathbf{x}^\top$ | the transpose of a vector $\mathbf{x}$ |
| $0_{m\times n}$ | zero matrix of dimension $m \times n$ |
| $0_n$ | zero vector in $\mathbb{R}^n$ |
| $0$ | zero matrix/vector of size depending on the context |
| $I_n$ | identity matrix of dimension $n \times n$ |

# Chapter 1

# Introduction

## 1.1 Overview

Experimental sciences are built upon the conviction that, despite imperfection due to noises, device limitation, and other adulterate factors, empirical data should still be laden with certain degrees of reliability or embedded with some essential characteristics of the original problems. In order to improve the reliability or extract the essence, it is critical to update the underlying models or refine the datasets for the purpose of reducing or even eliminating errors. Conventional error correction involves the reconstruction of models by imposing new updated information to postulate the existence of exact data or by screening existent data with the hope of inferring some innate particulars included in the true but cloaked data. These procedures of constituting or refining original models play an important role in many areas of sciences and engineering, including applied mechanics, circuit analysis, electrical oscillation, vibro-acoustics, information retrieval, bioinformatics, digital image processing, and so on. In the area of vibro-acustics, for example, we can get rid of the noise and access a suitable vibration control for air handling system by fine-tuning through some real-time feedback results from sensors. In the area of image processing, the presence of atmosphere turbulence alone has frustrated astronomers ever since telescopes were invented. Without the correction to the turbulence, it is known that no design or optical quantity of a telescope can improve the degraded image. A remedy comes only recently when a mechanical means, generally referred to as the adaptive optics, has been developed to perform turbulence compensation with promising effect. Along these lines, this research is devoted to matrix data reconstruction with primary focus on *quadratic inverse eigenvalue problems* (QIEPs) and *low rank factorization* (LRF) where data in the underlying models are represented by matrices of numerical attributes.

In contrast to the well developed theory and computation for the classical eigenvalue problem, the inverse eigenvalue problem (IEP) concerns the reconstruction of a structured matrix

from prescribed spectral data. It is clear that an IEP is trivial if the desirable matrix is not restricted to a certain structure. As IEPs often arise from some physical systems, we indeed need to ensure that the output of reconstruction not only follows the underlying natural frequencies or normal models but also satisfies certain feasibility constraints. For example, the matrix representation to the Sturn-Liouville operator is necessarily of the Jacobi structure. In the inverse reconstruction, we want to guarantee that the recovered data still preserve the same structure. The IEP for Jacobi matrices is now considered classical. Major results on this subject can be found in [19, 68, 71]. See also [37, Section 4.2] for a comprehensive review of its variants. Quadratic inverse eigenvalues related to practical applications inherit more complicated geometric structures and are notoriously more difficult. To make the matter worse, in many cases it is not enough to just consider geometric structures. Often we need to take other properties of parameters or eigeninformation, such as nonnegativity, into consideration simultaneously. Our main contribution in this investigation is to establish a general framework for solving QIEPs with general intrinsic structures. To set forth our consideration, four fundamental issues should be addressed in any type of QIEPs [37].

- Solvability. Not all vibration behavior is achievable. The most fundamental question for a QIEP is to determine conditions under which a solution exists. It is also sometimes desirable to determine whether a solution is unique.

- Computability. Suppose that a given set of eigeninformation is feasible. It is natural to seek a numerical algorithm for computing a solution. It is further desirable to improve the algorithm to make its computation more effective and efficient.

- Sensitivity. By sensitivity analysis we refer to the study of how reconstructed results will be affected, qualitatively or quantitatively, by perturbations in the input data. Such an analysis helps to qualify the numerical computation and provides the capacity of further refining a model by seeking the errors hidden in the model.

- Feasibility. The feasibility is closely linked to the exactness or completeness of a given data, or the suitability of the output result while testing it on the physical system.

The problem of LRF arises in a large variety of disciplines in modern science and engineering. Its main goal is to to compress the overly excess data in terms of fewer and more concise representatives and to assess the loss or the retrieval of important information. Most conventional approaches for solving LRF can be categorically cast as the so-called *vector space model* [170]. Roughly speaking, each model in the dataset is reshaped into a vector and the collection of those vectors becomes a single data matrix. Depending on the data types, the assessment of loss (or retrivial) is based on different measures in computing the similarity among different

models. For example, for continuous data, we might compute the heterogeneities via the Euclidean metric whereas for discrete data, we might employ the Hamming metric to count the cardinality of distinct elements, instead of real difference, as a means of representing dissimilarity. This way of representing data as vectors and searching for its best low rank approximation is stimulated with the aim of reaching the following goals [29].

- Compression. In real-world data, it often involves a large number of attributes related to diverse characteristics. A natural question is whether a set of representative patterns involving relatively smaller amount of parameters is sufficient to characterize the original model.
- Concision. We hope that the original data could be clarified by enhancing its characteristics via much more representative patterns derived from the data.
- Celerity. It is always desirable to obtain the best low rank approximation fast and effectively.

Our objectives in this investigation are to provide an informative and wide-scope overview of these interesting problems, to introduce some of their many applications, to explore mathematical properties, and to formulate numerical schemes so that: (i) a decision about the solvability of a QIEP can always be made either by theory or by computation; and (ii) an LRA could be achieved through the notion of "factorizing" data into a product of representative objects while maintaining its specific data type.

## 1.2 Quadratic Inverse Eigenvalue Problems

When attempting to characterize vibration phenomenon arising in solid mechanics, circuit analysis, electrical oscillation, acoustics, or finite element models of some PDEs, we often will come to a second-order dynamical system

$$M\ddot{\mathbf{y}} + C\dot{\mathbf{y}} + K\mathbf{y} = f(t), \tag{1.1}$$

where $\mathbf{y}(t)$ stands for an appropriate state variable and $M$, $C$, $K$ are time-invariant, real $n \times n$ matrices representing embedded physical parameters. It is well known that if $\mathbf{y}(t) = \mathbf{u}e^{\lambda t}$ represents a fundamental solution of (1.1), then the scalar $\lambda$ and the vector $\mathbf{u}$ must satisfy the algebraic equation

$$(\lambda^2 M + \lambda C + K)\mathbf{u} = 0. \tag{1.2}$$

In the forward setting, known as the quadratic eigenvalue problem (QEP) [70, 157], the eigenvalues and eigenvectors of the quadratic matrix polynomial (1.2) are used to describe the

dynamical behavior of the underlying physical system with coefficient matrices $M$, $C$ and $K$ already determined from the specified physical parameters. In the inverse setting, which we refer to as the quadratic inverse eigenvalue problem (QIEP), the task is to validate, determine, or estimate the parameters of the system according to its observed or expected behavior. That is, the aim of a QIEP is to determine physical parameters from its complete or partial eigeninformation.

In general, a QIEP can be defined as follows [33]

(**QIEP**). Construct a nontrivial quadratic matrix polynomial $Q(\lambda) = \lambda^2 M + \lambda C + K$ so that its matrix coefficients $(M, C, K)$ are of a specified structure and $Q(\lambda)$ has a specified set $\{(\lambda_i, \mathbf{u}_i)\}_{i=1}^k$ as its eigenpairs.

There are two main mandates embedded in the above definition: structural preservation and eigeninformation compliance. Indeed, as we will see in the subsequent discussion, it is these specified constraints embedded in the structure that make QIEPs difficult and challenging. A partial list illustrating different approaches to the QIEPs in the literature is briefed as follows. Ram and Elhay [144] determine a damped oscillatory system with symmetric tridiagonal coefficient matrices from two sets of prescribed eigenvalues. Starek and Inman [153] propose a numerical algorithm for determining a nonproportional damped system with respect to symmetric coefficient matrices. Lancaster and Prells [117] investigate a solution with symmetric and positive definite $M$ (denoted henceforth by $M \succ 0$), symmetric and positive semi-definite $C$ (denoted by $C \succeq 0$), and $K \succ 0$ from complete spectral information about eigenvalues and eigenvectors. Later, Lancaster [114] advances the notion of solving symmetric coefficient matrices $M$, $C$, and $K$, given all eigenvalues and partial eigenvectors corresponding to only real eigenvalues. Chu, Kuo and Lin [38] characterize a sufficient condition for the solvability of QIEPs with partially prescribed eigenpairs while $M \succ 0$ and $K \succeq 0$. Kuo, Lin and Xu [113] later put forward the parameterization of symmetric coefficient matrices ($M \succ 0$, $C$, $K$) with $k \leq n$ by using QR decomposition. This result soon is further generalized in [24]. More general representation of symmetric ($M \succ 0$, $C$, $K$) in terms of eigenvalues and eigenvectors is discussed in [40, 70] . See also [49, 51, 135] for the related work on applying the notions of feedback control to reassign the eigenstructure, but the loss of symmetry. Despite the numerous endeavors, research results advanced thus far for QIEP are incomplete, fragmental, and indeed quite inadequate. Most disappointing is that no structures other than symmetry or positive definiteness on some coefficients can be addressed by these techniques.

Developing effective techniques for solving structured inverse eigenvalue problems has been in quite desperate need but the advance thus far is limited to special cases. The most common discussed structure is symmetry in each of the coefficient matrices $(M, C, K)$. When the condition of positive definiteness is imposed upon the coefficient matrices, current theory requires

the complete information on eigenvalues and eigenvectors [117]. This is far from being sufficient to embrace the subtle properties such as inner-connectivity among elements of the underlying physical system or the requisite nonnegativity of the physical parameters for the sake of feasibility. For other types of structures, such as the gyroscopic systems where matrix coefficients are composed of positive definite and skew-symmetric matrices or local model updating problems where some entries in the coefficient matrices are prescribed and fixed, there is simply no theory or techniques available at all. At present, effective numeral algorithms for QIEP are few and those already developed are geared toward some specific and non-refined problems which cannot be generalized to other types of quadratic systems with different structure.

The notion of QIEP is of fundamental importance, because its ultimate goal of constructing or updating a vibration system from some observed or desirable dynamical behaviors while respecting some inherent feasibility constraints suits well many engineering applications in a variety of industries, including aerospace, automobile, manufacturing, and national defense. Thus far, however, QIEPs have largely remained challenging both theoretically and computationally due to the great variations of structural constraints that must be addressed. Of notable interest and significance in our investigation is to provide a general theoretical framework for the basic real and symmetric models and several numerical procedures for solving QIEPs with almost all kinds of specified structures.

## 1.3 Low Rank Factorization

Computing matrix factorization is an important task in numerical linear algebra. Theories and numerical applications for matrix factorization have received great attention in modern sciences and techniques. The general purposes of matrix factorization could be enlightened through an interesting appraisal by Hubert, Meulman, and Heiser [97],

> "Matrix factorization in the context of numerical linear algebra (NLA) generally serves the purpose of rephrasing through a series of easier subproblems a task that may be relatively difficult to solve in its original form."

Taking the problem of solving a linear system $A\mathbf{x} = \mathbf{b}$ for $A \in \mathbb{R}^{n\times n}$, $\mathbf{x}$ and $\mathbf{b} \in \mathbb{R}^n$, for example, we all know that it is impracticable to formulate $A^{-1}$ by any means. Instead, we formulate the LU factorization, which transforms the originally complex problem into a two easier subproblems of triangular systems. The purpose of this process is merely to decompose the square matrix $A$ into a lower triangular matrix $L$ and an upper triangular matrix $U$. This matrix factorization is only as a computationally convenient tool which has no real importance in and of itself.

In an entirely different context, we can intepret the matrix $A \in \mathbb{R}^{m\times n}$ as a data matrix such that each row corresponds to an object (subject) over $n$ attributes (variables). Under this model, there are a total of $m$ objects individually corresponding to $n$ attributes. Such a collection in the matrix form is for the purpose of classifying the given data by comparing the similarity between the objects with respect to a certain metric. The notion of matrix factorization is not so much directed at solving a linear equation or finding eigenvalues, but more so at revealing critical information within observations and their features. In the context of applied statistics/psychometrics (AS/P), it is said that "matrix factorizations are again directed toward the issue of simplicity, but now the actual components making up a factorization are of prime concern and not solely as a mechanism for solving another problem [97]." We prefer to be able to "interpret the matrix factorization geometrically and actually present the results spatially through coordinates obtained from the components of the factorization". If such a factorization can be achieved, we might better understand the structure or retrieve the information hidden in the original data matrix. That is,

> "The major purpose of a matrix factorization in this context is to obtain some form of lower-rank (and therefore simplified) approximation to $A$ for understanding the structure of the data matrix, particularly the relationship within the objects and within the attributes, and how the objects relate to the attributes [97]."

The notion of LRF plays an important role in machine learning, statistics, signal processing, and other informatic related fields. It aims to retrieve a sensible low-dimensional approximation from the original data matrix. We can regard LRF as a special kind of matrix factorization. Similar to eigenvalue problems, the LRF problem could also be considered from both a direct or an inverse points of view. The direct problem of an LRF is to formulate the original data matrix by computing the product of two low rank matrices. Clearly, we do not have trouble in doing such a calculation. On the other hand, the inverse problem of LRF is to minimize the difference between the original target matrix and the product of its low rank factorizations. Depending on the types of constraints imposed on the factors, there are at least three different ways to formulate a matrix factorization, all of which are studied in this research. Unless otherwise mentioned, the subsequent discussion of the LRF is about solving the inverse problem.

The traditional approach to the LRF is to first express the matrix $A$ as the product of two or more factors and then perform suitable truncations. In the ultimate sense, the factorization of the matrix $A$ is equal to the sum of rank-one matrices. The goal of breaking down the matrix into rank-one matrices could be achieved via the well known Wedderburn rank-one reduction formula. This formula is capable of unifying fundamental factorization processes under one framework. Indeed, Chu, Funderlic and Golub in the review paper [35] have shown that almost all matrix decompositions known in NLA could be obtained via this mechanism.

The mathematical approach of rank one reduction might not be deservedly recognized in the NLA community, but its concept as well as applications have been prevalent for decades in the AS/P community. See, for example, discussions in [45, 80, 81, 82, 85, 93] and also the timeline about the appearance of rank reduction results in [97]. In this approach, the data types involved in the calculation usually are real or complex numbers.

Another important variant of the LRF is the *nonnegative matrix factorization* (NMF) where the data to be analyzed are nonnegative. There are numerous applications of the NMF, including text mining [54, 169], image articulation [119, 140], bioinformatics [30], micro-array analysis [57], cheminformatics [138, 139], air quality research [138], spectral data analysis [16], and cancer pattern discovery [23, 64, 107]. The idea behind the NMF is to represent original model by a sequence of more compact representations whereas entries are nonnegative values for the purpose of satisfying physical realities. The mathematical formulation for an NMF can be defined as follows [139]:

**(NMF)** *Given a nonnegative matrix* $A \in \mathbb{R}^{m\times n}$ *and a positive integer* $p < \min\{m, n\}$, *find nonnegative matrices* $U \in \mathbb{R}^{m\times p}$ *and* $V \in \mathbb{R}^{p\times n}$ *so as to minimize the functional*

$$f(U, V) := \frac{1}{2}\|A - UV\|_F^2. \tag{1.3}$$

Note that the equality $A = UV$ may never be true, but the two matrices $U, V$ are still termed as low rank "factors" of $A$ in the literature. Classical tools cannot guarantee to maintain the nonnegativity. Considerable research efforts have been devoted to develop special techniques for NMF. See, for example, the multiplicative update algorithm by Lee and Seung [119, 120], the gradient methods [34, 96], and alternating least square methods [17, 22, 108, 118, 138]. Additional discussions can be found in [39, 56, 91, 95, 105, 152], and the many references contained therein. In this research, we offer yet another geometric approach by taking advantage of the powerful Hanh-Banach theory.

The two types of factorizations and their approaches stated above are distinct in nature. The feature of the data they intend to handle, however, are all from a continuum domain. There is yet another important class of information, namely, discrete data, that we have to consider in this investigation. To motivate the significance, envisage the scenario of switch manufacturing in the telecommunications industry [132]. Suppose that each switch cabinet consists of n slots which can be fitted with only one type of specified board options. Assume that there are $t_i$ different board types for each slot. In all, there are $\prod_{i=1}^{n} t_i$ different models. It is desirable to build a few basic models at the beginning so that corresponding to different custom orders, we might meet a specific configuration more efficiently. The question is how many and what basic models should be built. One plausible approach is to "learn" from past experiences, that is, we try to obtain the basic model information from the matrix of past sales. Labeling each board

type in a slot by an integer (or any token) while the same integer for different slots (columns) may refer to different attributes, the data representing past $m$ customer orders form an integer matrix of size $m \times n$. We must emphasize that in this setting, entries in the data matrix are of discrete values and that these values may or may not reflect any ordinal scale. A factorization of such a data matrix therefore must be subject to certain specifically defined arithmetic rules, which constitutes the main thrust of our work on this subject.

Without causing ambiguity, let $\mathbb{Z}$ denote either the conventional set (or subset) of integers when ordinal scale matters, or the set of all possible tokens which are not totally preordered, for the system under consideration. For the simplicity of discussion, we shall assume that the entries of the given matrix $A$ are from the same set $\mathbb{Z}$. In practice, different rows of $A$ may be composed of elements from different subsets of $\mathbb{Z}$. Even the same element in $\mathbb{Z}$ may have different meaning in different columns. If a comparison between two observations (rows) is needed, entries at different locations in the rows might need to be measured differently. Before we can perform the factorization, a properly defined metric for the row vectors, such as the metric for the product space (of individual entries), therefore must be in place, which then induces a metric $d$ to measure the "nearness" between two matrices. With this in mind, we define our integer matrix factorization (IMF) as follows.

> **(IMF)** Given an integer matrix $A \in \mathbb{Z}^{m\times n}$, an induced metric $d$, and a positive integer[1] $k < \min\{m, n\}$, find a binary matrix $U$ of size $m \times k$ with mutually orthogonal columns and $V \in \mathbb{Z}^{k\times n}$ so that the functional
>
> $$f(U, V) := d(A, UV), \tag{1.4}$$
>
> is minimized.

Each entry $a_{ij}$ in $A$ denotes, in a broad sense, the *score* obtained by entity $i$ on variable $j$; the value of $u_{i\ell}$, being dichotomous, indicates whether entity $i$ is influenced by "factor" $\ell$; and $v_{\ell j}$ marks the *attribute* of variable $j$ to the factor $\ell$. Take the sale record of a certain electronic device as the matrix $A$, for example. Then rows of $A$ denote the specifications by $m$ customers with respect to the $n$ different slots in the device; rows of $V$ represent the basis models to be built; and the mutual exclusion of columns of $U$ implies that each customer order corresponds to exactly one possible basic model. The factors in this case represent some abstract concepts by which the customer orders are divided into $k$ clusters. Clearly, this is a typical classification problem written in factorization form. We also see that the number $k$ plays an important role in the low rank approximation. In practice, we prefer to have as few factors as possible while keeping the objective value $f(U, V)$ low.

[1]The determination of such a rank $k$ is itself an important question which thus far has no satisfactory answer yet.

## 1.4 Research Outline

Inverse problems of matrix data reconstruction continue to be a fertile and flourishing field for modern sciences and technology because it involves a rich vein of undiscovered knowledge that cannot be manifested in direct problems. For instance, when solving inverse problems, we often encounter a series of conditions that may not present in the direct problem: the given information is incomplete; the original model satisfies certain constraints; the solution does not exist; the solution is not unique; the solution does not depends continuously on the given data, especially, the problem might not be well posed [68]. These questions are challenging but interesting, and may have impact on many important applications.

This research aims at providing a comprehensive view of two important topics, QIEPs and LRFs. We present our research results in eight chapters. We outline our organization as follows. Part I deals with QIEPs. In Chapter 2 we set forth a mathematical framework for determining real, symmetric coefficient matrices $(M, C, K)$ in a basic QIEP from merely a prescribed or observed subset of natural modes. This approach is in contrast to the classical approach to IEPs where the model is to be constructed from natural frequencies corresponding to various boundary conditions. In particular, merely given the cardinalities of real or complex eigenvalues but not the actual eigenvalues, the set of eigenvectors can be completed by solving an underdetermined nonlinear system of equations. This completion shows that the construction of symmetric coefficient matrices $(M, C, K)$ is possible even if the underlying system possesses arbitrary eigenvalues. Generic conditions under which the real symmetric quadratic inverse mode problem is solvable are discussed. Applications to important tasks such as updating models without spill-over or constructing models with positive semi-definite coefficient matrices are investigated. In Chapter 3 we propose a numerical approach via truncated QR decomposition to solve the inverse problems for arbitrary generally linked systems with inexact eigeninformation. This QR approach provides us with the capacity of determining the solvability of inverse problems and, if the inexact data are still feasible, computes the coefficient matrices with an estimate of residual error. In other words, we offer an approach that is general and robust for any kind of physical configuration. In Chapter 4, we introduce a powerful semi-definite programming (SDP) approach, to handle QIEPs under almost all kinds of structured constraints. Because of the great variations of structured constraints inherent in diverse engineering applications, the QIEPs thus far have remained challenging both theoretically and computationally. Of notable interest and significance in this chapter are the uniformity and the simplicity in the SDP formulation that solves effectively many otherwise very difficult QIEPs. Part II deals with LRFs. In Chapter 5, we compute the factorization of nonnegative matrix factorization by formulating the problem as a low dimensional polytope approximation. We approximate such a polytope on the probability simplex by another simpler polytope with fewer facets. Theoretical

assertion by the Hahn-Banach theorem guarantees that the difference between two polytopes can be reduced by computing the supporting hyperplane for a given point and a disjoint polytope. Since our transformation allows us to work on a compact set with known boundary, it is easier to trace the approximation procedure. In contrast to most existing algorithm where only an approximate map is used, our approach finds the unique and global minimum per sweep. Numerical results show that our method is more effective than the popular Lee-Seung algorithm in reaching the optimal approximation. In Chapter 6, we study the factorization of discrete datasets, namely integer or binary data types. It has been shown that factorization of discrete datasets generally leads to NP-hard problems. Compared with traditional techniques such as k-means methods [127], k-modes algorithm [98], and vector-quantization [75], our methods, which could be viewed as a generalized algorithm discussed in [110, 111], has no constraint on the cluster number and the cluster size. More importantly, based on a sequence of the "best" rank-one approximation, we can guarantee that each approximation is global minimization at each sweep. Even under the additional condition that the cluster number or cluster size is fixed, our algorithm enjoys a surprising but pleasant feature that is similar to the truncated singular value decomposition. That is, we are able to obtain the best possible truncated low rank factorization for discrete data. Numerical testing experiments with popular collections such as swimmer, Wisconsin breast cancer, and mushroom date sets seem to support our algorithms and theories very well. In Chapter 7 we discuss the factorization of a given nonnegative matrix $A$ into a sequence of rank one matrices each of which successively downdates the rank of $A$ by one. Again, computing the exact nonnegative rank and the corresponding factorization are known to be NP-hard. Available NMF techniques, which mostly utilize the notion of approximation, can hardly guarantee the required equality in a complete factorization. Alternately, our work employs the Wedderburn rank reduction formula to recursively extract, whenever possible, a rank-one nonnegative portion from the previous matrix while keeping the residual nonnegative and lowering its rank by one. The idea can equally be applied to completely positive matrices with a slight modification for symmetry. Numerical testing seems to suggest that this numerical approach, though heuristic in nature, performs reasonably well in factoring a given nonnegative matrix via rank-one downdate. Finally, in Chapter 8 we investigate the rank properties of the Euclidean distance matrix with $n$ distinct points in $\mathbb{R}^r$ and show that, while algebraic rank is generically $r + 2$, its nonnegative rank is generically $n$ for 1-dimensional Euclidean distance matrices.

# Part I

# Quadratic Inverse Eigenvalue Problems

# Chapter 2

# Inverse Problems with Basic Real Symmetric Models

## 2.1 Overview

We begin our investigation with a synopsis of the spectral decomposition in real, symmetric coefficient matrices $(M, C, K)$. Spectral decomposition of a real, symmetric matrix polynomials has been well developed in literature [41, 114]. However, for our application, we find that it is critical to decipher the underlying structures of spectral decomposition in more details. Understanding the underlying structure enables us to address the solvability of real and symmetric QIEPs from merely partial information about eigenvectors. This is the so called inverse mode problem (IMP) which is yet another special type of QIEPs. The notion of IMP perhaps can be exemplified by this famous paper entitled, "Can one hear the shape of a drum?" [104]. The question was, "if one has perfect pitch (to hear the natural frequencies), could one find the shape of a drum?". The IMP asks a counter analogy, "Can one see the sound of a string?", which means, "If one has perfect vision (to see the natural modes), could one tell the tone of the string?".

The IMP is first considered by Gladwell [66] for finite difference models of a vibrating rod. It is shown that for only two sets of eigenvalues and corresponding eigenvectors, the system could be uniquely constructed apart from a scale factor. Other related works on IMPs include Ram and Gladwell's construction of tri-diagonal mass and stiffness matrices from a single eigenvalue and two eigenvectors [146], Gladwell's general discussion about the formulations of simple chain-like structures which naturally involves tridiagonal structure [67], and Ram and Elishakoof's explication that the cross-sectional area of an axially vibrating non-uniform rod can be generated uniquely, up to a scale factor, from one eigenvector [145]. We offer a mathematical framework for the general IMPs subject to the mild constraint of coefficients being merely real

and symmetric.

One key advance in our approach is to rediscover the important role played by a specific block diagonal matrix $\mathscr{H}$ which is embedded in the spectral decomposition and whose structure depends only on the cardinalities of real and complex eigenvalues. It is a known fact that columns in the adjoint of the eigenvector matrix are $\mathscr{H}$-orthogonal[1] to themselves (see (2.5)). It is thus typically assumed that, after properly transformation, this $\mathscr{H}$ matrix is normalized to a canonical form known as the sip matrix [41, 70]. Such an assumption of $\mathscr{H}$ being a constant matrix, however, is not applicable for QIEPs because we usually do not have a complete list of eigenvectors. Partial eigeninformation is not adequate for us to perform the normalization of $\mathscr{H}$. To remedy this deficiency, the part of $\mathscr{H}$ matrix corresponding to the prescribed eigenvectors must be treated as unknown to be solved simultaneously with the IMP itself. This way of "relaxing" some designated entries of the matrix $\mathscr{H}$ as free variables offers a framework that unifies different approaches in the literatures and resolves some difficult issues encountered in [41] and [114]. In this chapter, we discuss two new applications of our general framework.

First, we propose a mechanism for updating an existent quadratic model with no spill-over. Updating a given quadratic model with newly measured eigeninformation has been an important task in engineering applications [27, 41, 52, 115]. The idea is to replace the portion of eigenvalues or eigenvectors which might be unwanted, uncontronlled, or inaccurate. In pole assignment problem, for example, the goal is to adjust the eigenvalues (poles) only [133]. An added challenge, which thus far has no satisfactory solution, is to avoid spill-over after updating. That is, it is desirable to keep the portion of eigenstructure in the original system, which is either unmeasured or already acceptable, invariant during the updating process.

Second, we separate the roles of eigenvalues and eigenvectors in a QIEP and show that the role of eigenvalues must take into account only when the completed set of eigenvectors is not sufficient for addressing additional structural constraint such as positive definiteness. We have numerical evidence showing that the eigenvectors for the QIEP with symmetric and positive definite coefficient matrices cannot be arbitrary.

This chapter is organized as follows. We begin in Section 2.2 with the classical theory of parameterization of real and symmetric coefficient matrices. The most general setting over the complex field is developed in the seminar book by Gohberg, Lancaster, and Rodman [69], which later is modified for the real field by Chu and Xu [41]. We then investigate the structure of a critical matrix $\mathscr{H}$ which provides a basis for achieving the completion of eigenvectors in Sections 2.2.2 and 2.3. Unlike the conventional approach in the literature where $\mathscr{H}$ is assumed to be a fixed, constant matrix for the sake of simplicity, the matrix $\mathscr{H}$ is characterized by the block diagonal structure through the information of the cardinalities of real and complex eigenvalues. Then, this matrix $\mathscr{H}$ with its undetermined nature is used as a parameter in

[1] A column vector $\mathbf{x}$ is $\mathscr{H}$-orthogonal to a column vector $\mathbf{y}$ if and only if $\mathbf{x}^*\mathscr{H}\mathbf{y} = 0$.

describing a solution for the IMPs. In Section 2.4, we apply this parameterization to model updating problems with no spill-over. In Section 2.5, the problem of positive definite or positive semi-definite property of the coefficient matrices will be put into consideration.

## 2.2 Fundamental theorem on spectral decomposition

In this section we briefly review the spectral representation of a real symmetric coefficient matrices $(M, C, K)$, Of particular importance is a special matrix $\mathscr{H}$ which plays a role like tuning the representation. For this discussion, the complete spectral information of the matrix polynomial $\mathfrak{Q}(\lambda)$ must be known.

### 2.2.1 Role of the parametric $\mathscr{H}$ matrix

To begin with, we need to introduce the notion of a standard pair [69, 70] which elegantly encapsulate spectral information in the most general form as we shall see below.

**Definition 2.2.1** *A pair of matrices $(\mathfrak{X}, \mathfrak{J}) \in \mathbb{C}^{n\times 2n} \times \mathbb{C}^{2n\times 2n}$ is called a standard pair for the quadratic matrix polynomial $\mathfrak{Q}(\lambda)$ if and only if the pair $(\mathfrak{X}, \mathfrak{J})$ satisfies the equation*

$$M\mathfrak{X}\mathfrak{J}^2 + C\mathfrak{X}\mathfrak{J} + K\mathfrak{X} = 0 \tag{2.1}$$

*and the square matrix $\begin{bmatrix} \mathfrak{X} \\ \mathfrak{X}\mathfrak{J} \end{bmatrix} \in \mathbb{C}^{2n\times 2n}$ is nonsingular.*

Given a quadratic matrix polynomial $\mathfrak{Q}(\lambda)$ with nonsingular leading coefficient matrix $M$, define two matrices $\mathscr{B}$ and $\mathscr{C}$ in $\mathbb{R}^{2n\times 2n}$ such that

$$\mathscr{B} := \begin{bmatrix} C & M \\ M & 0 \end{bmatrix}, \quad \mathscr{C} := \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}C \end{bmatrix}. \tag{2.2}$$

It is not difficult to check that the matrix $\mathscr{C}$, an equivalence of the companion matrix for a scalar polynomial has the same eigenstructure as that of $\mathfrak{Q}(\lambda)$. Corresponding to any given standard pair $(\mathfrak{X}, \mathfrak{J})$ for $\mathfrak{Q}(\lambda)$, define a special matrix $\mathscr{H}$

$$\mathscr{H}(\mathfrak{X}, \mathfrak{J}) := \begin{bmatrix} \mathfrak{X} \\ \mathfrak{X}\mathfrak{J} \end{bmatrix}^* \mathscr{B} \begin{bmatrix} \mathfrak{X} \\ \mathfrak{X}\mathfrak{J} \end{bmatrix}. \tag{2.3}$$

This matrix $\mathscr{H}(\mathfrak{X}, \mathfrak{J}) \in \mathbb{C}^{2n\times 2n}$ plays an important role in expressing the spectral decomposition of matrix coefficients $(M, C, K)$ in the following way [41, 69].

**Theorem 2.2.1** *Let $(\mathfrak{X},\mathfrak{J})$ be a standard pair for the quadratic matrix polynomial $\mathfrak{Q}(\lambda)$ and $\mathscr{H}=\mathscr{H}(\mathfrak{X},\mathfrak{J})$. Then*

$$\begin{cases} M=(\mathfrak{X}\mathfrak{J}\mathscr{H}^{-1}\mathfrak{X}^*)^{-1}, \\ C=-M\mathfrak{X}\mathfrak{J}^2\mathscr{H}^{-1}\mathfrak{X}^*M, \\ K=-M\mathfrak{X}\mathfrak{J}^3\mathscr{H}^{-1}\mathfrak{X}^*M+CM^{-1}C. \end{cases} \tag{2.4}$$

The converse is more important to us, namely, the existence of a matrix $\mathscr{H}\in\mathbb{C}^{2n\times 2n}$ can qualify a given a pair of matrices $(\mathfrak{X},\mathfrak{J})$ as a standard pair for some real symmetric quadratic matrix polynomial in the following way [41].

**Theorem 2.2.2** *Consider a pair of matrices $(\mathfrak{X},\mathfrak{J})\in\mathbb{C}^{n\times 2n}\times\mathbb{C}^{2n\times 2n}$. If there exists a nonsingular matrix $\mathscr{H}\in\mathbb{C}^{2n\times 2n}$ such that $\mathfrak{X}\mathfrak{J}\mathscr{H}^{-1}\mathfrak{X}^*$ is nonsingular and $\mathscr{H}$ satisfies the three equalities*

$$\begin{cases} \mathfrak{X}\mathscr{H}^{-1}\mathfrak{X}^* &= 0, \\ \mathscr{H}\mathfrak{J} &= (\mathscr{H}\mathfrak{J})^*, \\ \mathscr{H} &= \mathscr{H}^*, \end{cases} \tag{2.5}$$

*then $(\mathfrak{X},\mathfrak{J})$ is a standard pair for the real symmetric quadratic matrix polynomial $\mathfrak{Q}(\lambda)$ whose matrix coefficients $(M,C,K)$ are defined according to (2.4). Furthermore, the nonsingular matrix $\mathscr{H}$ is related to $(\mathfrak{X},\mathfrak{J})$ via (2.3) and also the relationship*

$$\mathscr{C}\begin{bmatrix}\mathfrak{X}\\ \mathfrak{X}\mathfrak{J}\end{bmatrix}=\begin{bmatrix}\mathfrak{X}\\ \mathfrak{X}\mathfrak{J}\end{bmatrix}\mathfrak{J}. \tag{2.6}$$

*holds with $\mathscr{B}$ and $\mathscr{C}$ defined by (2.2) in terms of the newly constructed $(M,C,K)$.*

The relationship (2.6) strongly suggest that the construction of $(M,C,K)$ based on (2.4) can be achieved by using *complete* eigeninformation $(\mathfrak{X},\mathfrak{J})$ which, in an QIEP setting, is not immediately available. Nonetheless, the relationship (2.5) gives rise to the structure of $\mathscr{H}$ and is particularly useful for studying the inverse problem later.

Let the Jordan canonical decomposition of $\mathscr{C}$ be denoted by

$$\mathscr{C}=\mathcal{Q}\mathcal{J}\mathcal{Q}^{-1}, \tag{2.7}$$

where $\mathcal{J}\in\mathbb{C}^{2n\times 2n}$ represents a block diagonal matrix consisting of Jordan blocks corresponding to eigenvalues of $\mathscr{C}$ and columns of $\mathcal{Q}\in\mathbb{C}^{2n\times 2n}$ are comprised of the corresponding generalized eigenvectors. Partition the matrix $\mathcal{Q}$ across rows into two blocks of size $n\times 2n$,

$$\mathcal{Q}:=\begin{bmatrix}\mathcal{X}\\ \mathcal{Z}\end{bmatrix}.$$

Rewriting (2.7) as

$$\begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}C \end{bmatrix} \begin{bmatrix} \mathcal{X} \\ \mathcal{Z} \end{bmatrix} = \begin{bmatrix} \mathcal{X} \\ \mathcal{Z} \end{bmatrix} \mathcal{J}, \tag{2.8}$$

we find the two relationships

$$\mathcal{Z} = \mathcal{X}\mathcal{J}, \tag{2.9}$$

$$M\mathcal{X}\mathcal{J}^2 + C\mathcal{X}\mathcal{J} + K\mathcal{X} = 0. \tag{2.10}$$

This implies that the Jordan pair $(\mathcal{X}, \mathcal{J})$ is indeed a *standard pair* for the quadratic matrix polynomial $\mathfrak{Q}(\lambda)$. It follows that the corresponding matrix

$$\mathcal{H} := \mathscr{H}(\mathcal{X}, \mathcal{J}) = \mathcal{Q}^* \mathscr{B} \mathcal{Q} \tag{2.11}$$

must satisfy the relationships $\mathcal{X}\mathcal{J}\mathcal{H}^{-1}\mathcal{X}^* = M^{-1}$, $\mathcal{X}\mathcal{H}^{-1}\mathcal{X}^* = 0$, $\mathcal{H} = \mathcal{H}^*$, and $\mathcal{H}\mathcal{J} = (\mathcal{H}\mathcal{J})^*$, from which the structure of $\mathcal{H}$ can be characterized.

### 2.2.2 Structure of the parametric $\mathscr{H}$ matrix

For simplicity, we shall assume henceforth that all eigenvalues of $\mathfrak{Q}(\lambda)$ are simple. Suppose that the Jordan matrix $\mathcal{J}$ along with the corresponding matrix $\mathcal{X}$ of eigenvectors are expressed as

$$\begin{cases} \mathcal{J} = \mathrm{diag}\{\lambda_1, \overline{\lambda}_1, \lambda_2, \overline{\lambda}_2, \ldots, \lambda_t, \overline{\lambda}_t, \lambda_{2t+1}, \ldots, \lambda_{2n}\}, \\ \mathcal{X} = [\mathbf{x}_1, \overline{\mathbf{x}}_1, \mathbf{x}_2, \overline{\mathbf{x}}_2, \ldots, \mathbf{x}_t, \overline{\mathbf{x}}_t, \mathbf{x}_{2t+1}, \ldots, \mathbf{x}_{2n}], \end{cases} \tag{2.12}$$

where $t$ is the number of distinct complex-conjugate pairs of eigenvalues. The congruence relationship

$$\begin{bmatrix} I_n & -\frac{1}{2}CM^{-1} \\ 0 & I_n \end{bmatrix} \mathscr{B} \begin{bmatrix} I_n & -\frac{1}{2}CM^{-1} \\ 0 & I_n \end{bmatrix}^T = \begin{bmatrix} 0 & M \\ M & 0 \end{bmatrix} \tag{2.13}$$

asserts that the matrix $\mathscr{B}$ must have equal numbers of positive and negative eigenvalues. As a consequence, the matrix $\mathscr{H}(\mathfrak{X}, \mathfrak{J})$ in general and $\mathcal{H}$ in particular should also have equal number of positive and negative eigenvalues. From the facts that $\mathcal{H} = \mathcal{H}^*$, $\mathcal{H}\mathcal{J} = \mathcal{J}^*\mathcal{H}$ and rearrange the ordering of real eigenvalues in $\mathcal{J}$ if necessary, we know that $\mathcal{H}$ must be a block diagonal of

the form [63, Chapter VIII, Theorem 1]

$$\mathcal{H} = \text{diag}\left\{ \begin{bmatrix} 0 & \overline{h}_1 \\ h_1 & 0 \end{bmatrix}, \ldots, \begin{bmatrix} 0 & \overline{h}_t \\ h_t & 0 \end{bmatrix}, h_{2t+1}, \ldots h_{2t+r}, -h_{2t+r+1}, \ldots, -h_{2n} \right\} \tag{2.14}$$

where $r := n - t$; for $j = 1, \ldots, t$, $h_j$ is a complex number; and for $j = 2t+1, \ldots, 2n$, $h_j$ is a positive real number. By definition, the values of $h_i$'s depend on the pair $(\mathcal{X}, \mathcal{J})$. Those real eigenvalues in $\mathcal{J}$ whose corresponding diagonal entries in $\mathcal{H}$ are positive (or negative) are said to have a positive (or negative) *sign characteristic*[2].

Now we can convert the standard pair $(\mathcal{X}, \mathcal{J})$ into a real-valued standard pair $(X, J)$ by defining

$$\begin{cases} J &:= \mathcal{R}\mathcal{J}\mathcal{R}^* = \text{diag}\{\lambda_1^{[2]}, \ldots, \lambda_t^{[2]}, \lambda_{2t+1}, \ldots, \lambda_{2n}\} \in \mathbb{R}^{2n \times 2n}, \\ X &:= \mathcal{X}\mathcal{R}^* = [\sqrt{2}\mathbf{x}_{1R}, \sqrt{2}\mathbf{x}_{1I}, \ldots, \sqrt{2}\mathbf{x}_{tR}, \sqrt{2}\mathbf{x}_{tI}, \mathbf{x}_{2t+1}, \ldots, \mathbf{x}_{2n}] \in \mathbb{R}^{n \times 2n}, \end{cases} \tag{2.15}$$

with the unitary transformation

$$\mathcal{R} := \text{diag}\left\{ \underbrace{\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix}, \ldots, \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ i & -i \end{bmatrix}}_{t \text{ copies}}, I_{2r} \right\}, \tag{2.16}$$

where $i = \sqrt{-1}$ and for $j = 1, \ldots, t$,

$$\begin{cases} \lambda_j^{[2]} &:= \begin{bmatrix} \alpha_j & \beta_j \\ -\beta_j & \alpha_j \end{bmatrix} \in \mathbb{R}^{2\times 2}, \quad \text{if } \lambda_j = \alpha_j + i\beta_j, \\ \mathbf{x}_j &:= \mathbf{x}_{jR} + i\mathbf{x}_{jI}. \end{cases}$$

The corresponding real matrix $H = \mathscr{H}(X, J)$ should have a similar block structure as in $\mathcal{H}$,

$$\begin{aligned} H &= \mathcal{R}\mathcal{H}\mathcal{R}^* \\ &= \text{diag}\left\{ \begin{bmatrix} a_1 & b_1 \\ b_1 & -a_1 \end{bmatrix}, \ldots, \begin{bmatrix} a_t & b_t \\ b_t & -a_t \end{bmatrix}, h_{2t+1}, \ldots h_{2t+r}, -h_{2t+r+1}, \ldots, -h_{2n} \right\} \end{aligned} \tag{2.17}$$

with $a_j$, $b_j \in \mathbb{R}$.

In the conventional setting for the forward problem, with appropriate scaling and rotations

[2]More details about the concept of sign characteristics can be found in the book [70] and their usages for QIEPs in the two recent articles [114, 117]. We only need the fact that real eigenvalues of $\mathcal{J}$ are divided into two mutually exclusive groups in our discussion.

of the eigenvectors, we can derive the following canonical form for the matrix $H$ even for the case that eigenvalues are semi-simple [41, Corollary 3.5] .

**Theorem 2.2.3** *Suppose that all eigenvalues of a given real symmetric quadratic pencil $\mathfrak{Q}(\lambda)$ are semi-simple but not necessarily distinct. Then there exists a real standard pair $(\mathfrak{X}, \mathfrak{J})$ such that*

$$\begin{cases} \begin{bmatrix} \mathfrak{X} \\ \mathfrak{X}\mathfrak{J} \end{bmatrix}^{\top} \begin{bmatrix} C & M \\ M & 0 \end{bmatrix} \begin{bmatrix} \mathfrak{X} \\ \mathfrak{X}\mathfrak{J} \end{bmatrix} = \Gamma := \operatorname{diag}\left\{ \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \ldots, \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \right\}, \\ \begin{bmatrix} \mathfrak{X} \\ \mathfrak{X}\mathfrak{J} \end{bmatrix}^{\top} \begin{bmatrix} -K & 0 \\ 0 & M \end{bmatrix} \begin{bmatrix} \mathfrak{X} \\ \mathfrak{X}\mathfrak{J} \end{bmatrix} = \Gamma\mathfrak{J}. \end{cases} \tag{2.18}$$

The matrix $\Gamma$ is precisely the so called *sip* (standard involutory permutation) matrix repeatedly referred to by Lancaster [114, 117]. In the setting for IMPs, nevertheless, only a partial list of eigenvectors is accessible in the beginning. We do not have enough information to know how these given eigenvectors should be scaled or rotated. All we can do is to use the given partial eigenvectors to construct an appropriate matrix $H$, which might not be the same as this particular sip form as in (2.18). In other words, the corresponding blocks in $H$ to the prescribed eigenvectors should be considered as part of the unknowns to be determined. This standpoint is a fundamentally different approach from, and is perhaps more correct than, that considered in [114, 117] where the sip form is implicitly but automatically assumed and thus significantly delimits the solvability because some rank conditions may not be satisfied.

## 2.3 Eigenvector completion

In Theorem 2.2.1, we have seen that the coefficient matrices $(M, C, K)$ could be constructed, provided a standard pair $(X, J)$ is given. However, we do not know $(X, J)$ in its entirety in the IMPs. The missing eigeninformation could be expiated by using Theorem 2.2.2, so long as a nonsingular matrix $H$ satisfying (2.5) could be found. Thus, our strategy is to first determine the structure of $H = \mathscr{H}(X, J)$ from the guessed or assigned structure, but not values, of $J$. This would automatically satisfy the last two conditions in (2.5). Finally, we use the first necessary condition

$$XH^{-1}X^{T} = 0. \tag{2.19}$$

to build $H$ and to complete $X$ simultaneously. The values of $J$, which actually can be assigned almost arbitrarily, are needed only to ensure the invertibility of the matrix $XJH^{-1}X^{\top}$, which is generically true.

To illustrate this idea, suppose that we have prescribed a subset of $k$ eigenpairs, which are closed under complex conjugation and have assumed a desirable number $t$ of complex conjugate pairs of eigenvalues or, equivalently, the desirable number $r = n - t$ of real eigenvalues with positive (or negative) sign characteristic in the constructed quadratic polynomial[3]. For convenience, we partition the columns of $X$ as

$$X = [\ \underbrace{C_0, \quad C_1}_{2t \text{ columns}}\ ,\ \underbrace{P_0, \quad P_1}_{r \text{ columns}}\ ,\ \underbrace{N_0, \quad N_1}_{r \text{ columns}}\ ] \tag{2.20}$$

where $[C_0, P_0, N_0]$ is a submatrix of size $n \times k$ whose columns represent the $k$ prescribed eigenvectors. Suppose further that among the prescribed eigeninformation there are $2k_C$ complex eigenvalues closed under conjugation, $k_P$ real eigenpairs with positive characteristics, and another $k_N$ real eigenpairs with negative characteristics[4]. The known $C_0$, $P_0$, and $N_0$, therefore, are of sizes $n \times 2k_C$, $n \times k_P$, and $n \times k_N$, respectively, with $k = 2k_C + k_P + k_N$. Columns of $[C_1, P_1, N_1]$ denote the unknown eigenvectors that are to be completed.

Clearly, $H^{-1}$ has exactly the same block structure as $H$. We might be as well working on $H^{-1}$ directly. Partition the inverse of the matrix $H$ in (2.17) into blocks of sizes conformal to those in (2.20) and denote,

$$H^{-1} = \operatorname{diag}\left\{H_0^C, H_1^C, H_0^P, H_1^P, -H_0^N, -H_1^N\right\}, \tag{2.21}$$

where each block has its own structure, e.g., $H_0^C$ is a $k_C \times k_C$ block diagonal matrix consisting of $2 \times 2$ submatrices, $H_0^P$ and $H_0^N$ respectively are $k_P \times k_P$ and $k_N \times k_N$ diagonal matrices with positive diagonal entries, and so on. We rewrite (2.19) as

$$C_0 H_0^C C_0^\top + P_0 H_0^P P_0^\top - N_0 H_0^N N_0^\top = N_1 H_1^N N_1^\top - P_1 H_1^P P_1^\top - C_1 H_1^C C_1^\top. \tag{2.22}$$

Since $H_1^P$ and $H_1^N$ are diagonal matrices with positive entries and $P_1$ and $N_1$ are indeterminate, we can redefine the products $P_1(H_1^P)^{1/2}$ and $N_1(H_1^N)^{1/2}$ as the new variables $P_1$ and $N_1$, respectively. Likewise, by the identity

$$\begin{bmatrix} a & b \\ b & -a \end{bmatrix} = U \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} U^\top,$$

[3]This pair of nonnegative integers $(t, r)$ is what we refer to as the structure of $J$.

[4]In practice, it appears that the choice of $k_P$ and $k_N$ is immaterial for IMPs so long as $k_P + k_N = k - 2k_C$. See the discussion in Section 2.3.1.

with

$$U := \underbrace{\begin{bmatrix} a+\sqrt{a^2+b^2} & -b \\ b & a+\sqrt{a^2+b^2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{b^2+(a+\sqrt{a^2+b^2})^2}} & 0 \\ 0 & \frac{1}{\sqrt{b^2+(a+\sqrt{a^2+b^2})^2}} \end{bmatrix}}_{\text{orthogonal}} \underbrace{\sqrt[4]{a^2+b^2}}_{\text{scaling}},$$

we can properly rotate and scale the columns of $C_1$ in (2.22) and redefine the variable $C_1$ such that the system (2.22) is reduced to

$$\Omega := C_0 H_0^C C_0^\top + P_0 H_0^P P_0^\top - N_0 H_0^N N_0^\top + C_1 \Upsilon C_1^\top + P_1 P_1^\top - N_1 N_1^\top = 0, \tag{2.23}$$

where the constant matrix is denoted to be

$$\Upsilon := \operatorname{diag}\left\{ \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \ldots, \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}}_{t-k_C \text{ copies}} \right\}.$$

It is important to point out that this reduction process cannot take place at the left hand side of (2.22) because $[C_0, P_0, N_0]$ are prescribed matrices which cannot assimilate the unknown scalings or rotations. The eigenvector completion problem amounts to finding a real matrix $[C_1, P_1, N_1]$ of size $n \times (2n-k)$, $2k_C$ real numbers for the block diagonal matrix $H_0^C$, and $k_P + k_N$ positive numbers for the diagonal matrices $H_0^P$ and $H_0^N$ so that the equation (2.23) is satisfied. Totally there are $n(2n-k)+k$ unknowns in $\frac{n(n+1)}{2}$ equations. If

$$k < \frac{n(3n-1)}{2(n-1)}, \tag{2.24}$$

then the system (2.23) is under-determined. Generically, the algebraic solutions to (2.23) form a nontrivial smooth manifold [79], but for our IMPs we also need positive $H_0^P$ and $H_0^N$ from this solution manifold. The following count on the cardinality of prescribed eigenpairs to guarantee solvability can easily be verified from (2.24).

**Theorem 2.3.1** *The maximal allowable number of prescribed eigenvectors so that the system (2.23) is generically solvable is given by*

$$k_{max} = \begin{cases} 3\ell+1, & \text{if } n = 2\ell, \\ 3\ell+2, & \text{if } n = 2\ell+1. \end{cases} \tag{2.25}$$

Interestingly enough, Theorem 2.3.1 offers exactly the same solvability condition (2.25)

which is proved in [32, Theorem 3.5] by using an entirely different approach. That is, suppose we are given $k$ eigenpairs $\{(\sigma_j, \mathbf{y}_j)\}_{j=1}^{k}$ which are closed under complex conjugation. Convert this eigenpair information into real-valued matrices $(\Sigma, Y)$ in the same way as we did in (2.15). Then if $k \leq k_{max}$ the coefficient matrices $(M, C, K)$ for the QIEP with eigenpair $(\Sigma, Y)$ are solutions to the linear system

$$[M, C, K] \begin{bmatrix} \Sigma Y^2 \\ \Sigma Y \\ \Sigma \end{bmatrix} = 0. \tag{2.26}$$

In this setting, we see that the eigenvalue information $\Sigma$ does come into play, no concern about finding positive $H_0^P$ and $H_0^N$ is needed, and all possible solutions $(M, C, K)$ to the QIEP forms a *linear* subspace. The trade-off, however, is that no iinformation about the remaining eigenstructure in the reconstructed matrix polynomial is revealed at all. In contrast, our current approach tackles the QIEP *without* the eigenvalue information. Given only eigenvalues structure in $J$, we seek to solve the nonlinear system (2.23) for the remaining eigenvectors $[C_1, P_1, N_1]$ and positive $H_0^P$ and $H_0^N$. Once the partial eigenvectors $Y$ is fully extended to a complete set $X$ of eigenvectors, the remaining eigenvalues can be almost arbitrarily assigned and the coefficient matrices $(M, C, K)$ are obtainable from the formula (2.4). Compared to the linear subspace formed by all feasible coefficient matrices $(M, C, K)$, all possible remaining eigenvectors in the polynomial system (2.23) form a *nonlinear* algebraic variety. In short, it is quite intriguing that two different approaches using different sets of information end up with the same bound on allowable number of prescribed eigenvectors.

An example might be more informative to demonstrate our point.

**Example 2.3.1** *Consider the simple case when $n = 2$ and $k = 2k_C = 2$. Suppose*

$$C_0 = \begin{bmatrix} 0 & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 \end{bmatrix}$$

*is given. There are only two ways to complete the eigenstructure.*

*Assuming first that the remaining two eigenvectors are real, that is, $r = 1$, then we need to determine $H_0^C = \begin{bmatrix} a_1 & b_1 \\ b_1 & -a_1 \end{bmatrix}$, $P_1 = [p_1, p_2]^\top$ and $N_1 = [n_1, n_2]^\top$ from the equation*

$$C_0 H_0^C C_0^\top + P_1 P_1^\top - N_1 N_1^\top = \begin{bmatrix} -\frac{1}{2}a_1 + p_1^2 - n_1^2 & -\frac{1}{2}b_1 + p_1 p_2 - n_1 n_2 \\ -\frac{1}{2}b_1 + p_1 p_2 - n_1 n_2 & \frac{1}{2}a_1 + p_2^2 - n_2^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

*This amounts to the under-determined system of three equations in six variables*

$$\begin{cases} p_1^2 - n_1^2 &= \frac{1}{2}a_1, \\ p_2^2 - n_2^2 &= -\frac{1}{2}a_1, \\ p_1p_2 - n_1n_2 &= \frac{1}{2}b_1. \end{cases}$$

*With $n_1$, $n_2$ and $b_1 \neq 0$ as free variables, the solution can be expressed as*

$$P_1 = \begin{bmatrix} \frac{\sqrt{(n_1+n_2)^2+b_1}\pm\sqrt{(n_1-n_2)^2-b_1}}{2} \\ \frac{\sqrt{(n_1+n_2)^2+b_1}\mp\sqrt{(n_1-n_2)^2-b_1}}{2} \end{bmatrix} \quad or \quad \begin{bmatrix} \frac{-\sqrt{(n_1+n_2)^2+b_1}\pm\sqrt{(n_1-n_2)^2-b_1}}{2} \\ \frac{-\sqrt{(n_1+n_2)^2+b_1}\mp\sqrt{(n_1-n_2)^2-b_1}}{2} \end{bmatrix}.$$

*Similarly, assuming the remaining two eigenvectors are complex, that is, $r = 0$, then we need to determine $H_0^C = \begin{bmatrix} a_1 & b_1 \\ b_1 & -a_1 \end{bmatrix}$ and $C_1 = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$ from the equation*

$$C_0H_0^CC_0^\top + C_1\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}C_1^\top = \begin{bmatrix} -\frac{1}{2}a_1 + c_{11}^2 - c_{12}^2 & -\frac{1}{2}b_1 + c_{11}c_{21} - c_{12}c_{22} \\ -\frac{1}{2}b_1 + c_{11}c_{21} - c_{12}n_2 & \frac{1}{2}a_1 + c_{21}^2 - c_{22}^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},$$

*or, equivalently,*

$$\begin{cases} c_{11}^2 - c_{12}^2 &= \frac{1}{2}a_1, \\ c_{21}^2 - c_{22}^2 &= -\frac{1}{2}a_1, \\ c_{11}c_{21} - c_{12}c_{22} &= \frac{1}{2}b_1. \end{cases}$$

*With $c_{12}$, $c_{22}$, and $b_1 \neq 0$ arbitrary, the solution can be expressed as*

$$C_1 = \begin{bmatrix} \frac{\sqrt{(c_{12}+c_{22})^2+b_1}\pm\sqrt{(c_{12}-c_{22})^2-b_1}}{2} & c_{12} \\ \frac{\sqrt{(c_{12}+c_{22})^2+b_1}\mp\sqrt{(c_{12}-c_{22})^2-b_1}}{2} & c_{22} \end{bmatrix} \quad or \quad \begin{bmatrix} \frac{-\sqrt{(c_{12}+c_{22})^2+b_1}\pm\sqrt{(c_{12}-c_{22})^2-b_1}}{2} & c_{12} \\ \frac{-\sqrt{(c_{12}+c_{22})^2+b_1}\mp\sqrt{(c_{12}-c_{22})^2-b_1}}{2} & c_{22} \end{bmatrix}.$$

*In this example, these two scenarios are essentially the same, but the general case can be considerably more complicated.*

We conclude this section with two important comments about the advantages of our approach for QIEPs. The first is that we are able to bypass the peculiar rank condition for the QIEP speculated in [114] because we reckon $H_0^P$ and $H_0^N$ as additional variables which are more general than the sip matrix. The second is that we are able to answer an open question raised in the same paper by bringing in $H_0^C$ as an additional variable. In both cases, our approach by utilizing the general $H$ matrix offers more flexibility in dealing with the QIEPs than any of the previous approaches using just the sip matrix. We summarize the new results below.

**Theorem 2.3.2** *Suppose that all eigenvalues are simple. Let $n = t + r$.*

1. *Suppose that all real eigenpairs are given, that is, only $P_0 \in \mathbb{R}^{n\times r}$ and $N_0 \in \mathbb{R}^{n\times r}$ are specified. Then, a necessary condition for the IMP to be solvable is that the system*

$$P_0 H_0^P P_0^\top - N_0 H_0^N N_0^\top + C_1 \Upsilon C_1^\top = 0. \tag{2.27}$$

*has a nontrivial solution for nonnegative diagonal matrices $H_0^P$, $H_0^N \in \mathbb{R}^{r\times r}$ and $C_1 \in \mathbb{R}^{n\times 2t}$. The maximal allowable number $r$ of columns for each $P_0$ and $N_0$ is bounded by*

$$r < \frac{n(3n-1)}{4(n-1)}. \tag{2.28}$$

2. *Suppose that all complex eigenpairs are given, that is, only $C_0 \in \mathbb{R}^{n\times 2t}$ is specified. Then, a necessary condition for the IMP to be solvable is that the system*

$$C_0 H_0^C C_0^\top = N_1 N_1^\top - P_1 P_1^\top. \tag{2.29}$$

*has a nontrivial solution for a block diagonal matrix $H_0^C \in \mathbb{R}^{2t\times 2t}$ with $2 \times 2$ symmetric blocks and $P_1$, $N_1 \in \mathbb{R}^{n\times r}$. The maximal allowable number $t$ of complex conjugate eigenvectors in $C_0$ is bounded by*

$$t < \frac{n(3n-1)}{4(n-1)}. \tag{2.30}$$

### 2.3.1 Issues related to numerical computation

We stress again that in our IMP approach for the QIEP the unspecified eigenvalues can be arbitrary, but the eigenvectors need to be extended to its full list. To carry out a specific eigenvector completion for an IMP of size $n$, we will assume that there are $t$ pairs of complex conjugate eigenvalues and $r$ $(= n - t)$ pairs of real eigenvalues with equal numbers of sign characteristics in the constructed quadratic matrix polynomial. Out of the $k$ prescribed eigenvectors, there already exist $k_P + k_N$ prescribed real eigenvectors. Among them, we generally do not know a priori the associated sign characteristics of their corresponding eigenvalues. The assignment of sign characteristics to the corresponding real eigenvalues therefore are at random. By contrast, the identification of the $2k_C$ $(= k - k_P - k_N)$ complex conjugate eigenvectors is easy. Also, it is necessary that $k_C \leq t$. To study whether the IMP is solvable for arbitrary splitting $n = t + r$ and distribution $k = 2k_C + k_P + k_N \leq k_{max}$ is another fascinating research topic which will not be addressed in the current research. (See Chapter 9.) In the subsequent discussion, we assume that $(t, r)$ and $(k_C, k_P, k_N)$ are given.

Our theory asserts a manifold of solutions for (2.23) so long as $k \leq k_{max}$. However, we must take into account that the diagonal matrices $H_0^P$ and $H_0^N$ should be positive. One possible approach to enforce the positivity is to consider the constrained nonlinear least squares optimization problem

$$\begin{cases} \text{Minimize} & f(H_0^C, H_0^P, H_0^N, C_1, P_1, N_1), \\ \text{Subject to} & H_0^P \geq 1 \text{ and } H_0^N \geq 1, \end{cases} \tag{2.31}$$

with the objective function defined by

$$f(H_0^C, H_0^P, H_0^N, C_1, P_1, N_1) := \frac{1}{2}\langle \Omega, \Omega \rangle, \tag{2.32}$$

where $\Omega$ is defined in (2.23), $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product, and to avoid the trivial solution, we have scaled "upward" the positivity of $H_0^P$ and $H_0^N$. Ideally, we would like to see a zero objective value at an optimal solution and our extensive numerical experiments suggest that this is not difficult to accomplish. We quickly point out another important topic for future study, that is, to develop a theory showing the existence of a positive solution on the manifold.

There are readily available software packages for solving (2.31). For preliminary testing, we find that the MATLAB routine fmincon which implements a subspace trust-region approach based on the interior-reflective Newton method and the preconditioned conjugate gradients method seems capable of finding a solution to (2.23) with high precision. We also have experimented with other optimization packages such as SNOPT [65] with similar success.

Of particular advantage in our formulation is that the derivatives of $f$ are readily available in closed form, which would help to enhance the efficiency in the optimization process. Specifically, by identifying the objective functional as a map $f : \mathbb{R}^{2k_C} \times \mathbb{R}^{k_P} \times \mathbb{R}^{k_N} \times \mathbb{R}^{n(2t-2k_C)} \times \mathbb{R}^{n(r-k_P)} \times \mathbb{R}^{n(r-k_N)} \to \mathbb{R}$, we can calculate the first-order partial derivatives of $f$ with respect to each group of variables as follows.

**Lemma 2.3.1** *Let $\frac{\partial f}{\partial \Phi}$ denote the partial gradient of $f$ in (2.32) with respect to $\Phi$ where the symbol $\Phi$ stands for any of the six variables $(H_0^C, H_0^P, H_0^N, C_1, P_1, N_1)$. Then*

$$\frac{\partial f}{\partial H_0^C} = [\gamma_{1,1} - \gamma_{2,2}, 2\gamma_{21}, \ldots, \gamma_{2k_C-1,2k_C-1} - \gamma_{2k_C,2k_C}, 2\gamma_{2k_C,2k_C-1}]^\top, \tag{2.33}$$

*with $\gamma_{i,j}$ denoting the $(i,j)$ entry of the matrix $C_0^\top \Omega C_0$,*

$$\begin{cases} \frac{\partial f}{\partial H_0^P} &= \mathbf{diag}(P_0^\top \Omega P_0), \\ \frac{\partial f}{\partial H_0^N} &= -\mathbf{diag}(N_0^\top \Omega N_0), \end{cases} \tag{2.34}$$

*with* $\mathbf{diag}(A)$ *denoting the column vector of the diagonal of the matrix* $A$*, and*

$$\begin{cases} \frac{\partial f}{\partial C_1} &= \mathbf{vec}(2\Omega C_1 \Upsilon), \\ \frac{\partial f}{\partial P_1} &= \mathbf{vec}(2\Omega P_1), \\ \frac{\partial f}{\partial N_1} &= -\mathbf{vec}(2\Omega N_1), \end{cases} \tag{2.35}$$

*with* $\mathbf{vec}(B)$ *denoting the vectorization of the matrix* $B$ *by stacking the columns of* $B$ *into a single column vector.*

**Proof.** It is most convenient to work in matrix form by employing rules from matrix calculus. For each variable $\Phi$, let $Z_\Phi$ denote an arbitrary element in the space where the partial derivative operator $\frac{\partial}{\partial \Phi}$ is acting on, that is, the matrix $Z_\Phi$ and the variable $\Phi$ have exactly the same structure. The type of $Z_\Phi$ and the particular action $\frac{\partial \Omega}{\partial \Phi}.Z_\Phi$ can be summarized as follows.

| $\Phi$ | $H_0^C$ | $H_0^P$ | $H_0^N$ | $C_1$ | $P_1$ | $N_1$ |
|---|---|---|---|---|---|---|
| $Z_\Phi$ | block diagonal w. $2 \times 2$ blocks | diagonal | diagonal | full | full | full |
| $\frac{\partial \Omega}{\partial \Phi}.Z_\Phi$ | $C_0 Z_{H_0^C} C_0^\top$ | $P_0 Z_{H_0^P} P_0^\top$ | $-N_0 Z_{H_0^N} N_0^\top$ | $Z_{C_1}\Upsilon C_1^\top + C_1 \Upsilon Z_{C_1}^\top$ | $Z_{P_1} P_1^\top + P_1 Z_{P_1}^\top$ | $-(Z_{N_1} N_1^\top + N_1 Z_{N_1}^\top)$ |

Fix all other variables and consider the Fréchet derivative of $f$ with respect to $H_0^C$ alone. Because of the way we identify $H_0^C$ as an element in $\mathbb{R}^{2k_C}$, the target point $Z_{H_0^C}$ on which the partial derivative acts should be a matrix in $\mathbb{R}^{2k_C \times 2k_C}$ that has exactly the same structure as $H_0^C$. Such an action is given by

$$\frac{\partial f}{\partial H_0^C}.Z_{H_0^C} = \langle C_0 Z_{H_0^C} C_0^\top, \Omega \rangle = \langle Z_{H_0^C}, C_0^\top \Omega C_0 \rangle = \langle Z_{H_0^C}, \rho(C_0^\top \Omega C_0) \rangle,$$

where $\rho$ stands for the projection of $\mathbb{R}^{2k_C \times 2k_C}$ onto the subspace of $k_C \times k_C$ block diagonal matrices of $2 \times 2$ blocks. By the Riesz representation theorem, we know therefore that the partial gradient $\frac{\partial f}{\partial H_0^C}$ with respect to the Frobenius inner product (over the subspace where

$Z_{H_0^C}$ resides) is $\rho(C_0^\top \Omega C_0)$. The crisscrossed pattern in (2.33) is simply due to a compaction of the Frobenius inner product to the standard Euclidian inner product.

The other partial gradients can be calculated in a similar way. The two operators **diag** and **vec** are needed only for a similar purpose of properly identifying the ambient space where the variable resides. □

To describe the Hessian of $f$ in the conventional $(n(2n-k)+k) \times (n(2n-k)+k)$ matrix form is a little bit more challenging. Nonetheless, the action of the Hessian $\nabla^2 f$ in the operator form can easily be characterized.

**Lemma 2.3.2** *Let $\frac{\partial^2 f}{\partial \Psi \partial \Phi}$ denote the partial derivative of $\frac{\partial f}{\partial \Phi}$ with respect to $\Psi$ where the symbols $\Phi, \Psi$ stand for any of the six variables $(H_0^C, H_0^P, H_0^N, C_1, P_1, N_1)$. Then the sectional Hessian $\frac{\partial^2 f}{\partial \Psi \partial \Phi}.Z_\Psi$ assumes the following actions,*

$$\begin{aligned}
\tfrac{\partial^2 f}{\partial \Psi \partial H_0^C}.Z_\Psi &= \rho\left(C_0^\top \left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) C_0\right), \\
\tfrac{\partial^2 f}{\partial \Psi \partial H_0^P}.Z_\Psi &= \mathrm{diag}\left(P_0^\top \left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) P_0\right), \\
\tfrac{\partial^2 f}{\partial \Psi \partial H_0^N}.Z_\Psi &= -\mathrm{diag}\left(N_0^\top \left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) N_0\right), \\
\tfrac{\partial^2 f}{\partial \Psi \partial C_1}.Z_\Psi &= \begin{cases} 2\left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) C_1 \Upsilon, & \text{if } \Psi \neq C_1, \\ 2\left(\left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) C_1 + \Omega Z_{C_1}\right) \Upsilon, & \text{if } \Psi = C_1, \end{cases} \\
\tfrac{\partial^2 f}{\partial \Psi \partial P_1}.Z_\Psi &= \begin{cases} 2\left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) P_1, & \text{if } \Psi \neq P_1, \\ 2\left(\left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) P_1 + \Omega Z_{P_1}\right), & \text{if } \Psi = P_1, \end{cases} \\
\tfrac{\partial^2 f}{\partial \Psi \partial P_1}.Z_\Psi &= \begin{cases} -2\left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) N_1, & \text{if } \Psi \neq N_1, \\ -2\left(\left(\tfrac{\partial \Omega}{\partial \Psi}.Z_\Psi\right) N_1 + \Omega Z_{N_1}\right), & \text{if } \Psi = N_1, \end{cases}
\end{aligned} \tag{2.36}$$

*where the action $\frac{\partial \Omega}{\partial \Psi}.Z_\Psi$ is given in the proof for Lemma 2.3.1.*

As an action from the Fréchet derivative, each of these operations in (2.36) is linear in $Z_\Psi$. The Hessian $\nabla^2 f$ is composed of these operations section by section on a long vector of length $2(n-k)+k$. It is not difficult to convert the operations in (2.36) into matrix-vector multiplication algorithmically, so long as we can identify which section in the vector $Z_\Psi$ is referred to. The analytic Hessian given above is not always critically needed in computation, but has its own merits in improving the efficiency. Given the closed form of gradient in Lemma 2.3.1, most optimization software can estimate the Hessian numerically by, say, the finite-difference method, which is what we choose to do in our current numerical experiments.

## 2.4 Model updating with no spill-over

By a model updating problem, we mean to update a portion of a given quadratic model by some newly measured eigeninformation. One challenge which is of practical importance in engineering applications is to update this model while keeping vibration parameters not related to the newly measured parameters invariant. The model updating problem can be described as follows;

**(MUP)** Given a real symmetric quadratic model with coefficient matrices $(\widetilde{M}, \widetilde{C}, \widetilde{K})$ and a few of its associated eigenpairs $\{(\lambda_j, \mathbf{x}_j)\}_{j=1}^k$ with $k < n$, assume that new eigenpairs $\{(\sigma_j, \mathbf{y}_j)\}_{j=1}^k$ have been measured. Update matrices $(\widetilde{M}, \widetilde{C}, \widetilde{K})$ to a new real symmetric quadratic model $(M, C, K)$ such that

(i) The newly measured $\{(\sigma_j, \mathbf{y}_j)\}_{j=1}^k$ form $k$ eigenpairs of the new model $(M, C, K)$.

(ii) The remaining $2n - k$ eigenpairs of $(M, C, K)$ are kept the same as those of the original $(\widetilde{M}, \widetilde{C}, \widetilde{K})$.

The second condition above is known as the *no spill-over phenomenon*. Model updating with no spill-over has been studied extensively. See, for example, [32, 40, 41, 62, 116]. No spill-over is required in the updating process either because the unrelated parameters are already acceptable in the previous model or engineers do not have any information about these. Hence it is expected that changes be made only to those newly measured parameters when updating the model. Indeed, it is highly desirable to construct the update $(M, C, K)$ *without* the knowledge of the remaining $2n - k$ eigeninformation. Our IMP approach can help to resolve the MUP.

The following formulation of a solution for MUP bears considerable resemblance to the solution for the eigenvalue embedding problems (EEP) except that the EEP intends to keep all eigenvalues invariant and pays no attention to whatever way the eigenvectors might be altered [41]. The EEPs are typically regarded as more manageable "locum tenentes" in the literature for the much harder MUPs, and now we have almost identical recipes for the solutions. The breakthrough hinges upon a recent discovery by Chu, Lin, and Xu [40, Theorem 4.1] about a necessary condition that the updated eigenvectors $\{\mathbf{y}\}_{j=1}^k$ must satisfy.

Assume as before that all eigenvalues of the original model $(\widetilde{M}, \widetilde{C}, \widetilde{K})$ are simple. Let the $k$ eigenpairs $\{(\lambda_j, \mathbf{x}_j)\}_{j=1}^k$ to be modified and the remaining $2n - k$ invariant eigenpairs $\{(\lambda_i, \mathbf{x}_i)\}_{i=k+1}^{2n}$ of the original model be denoted in real-valued form by

$$\left\{ \begin{array}{rcl} \Lambda_1 & := & \mathrm{diag}\left\{ \begin{bmatrix} \alpha_1 & \beta_1 \\ -\beta_1 & \alpha_1 \end{bmatrix}, \ldots, \begin{bmatrix} \alpha_{\ell_1} & \beta_{\ell_1} \\ -\beta_{\ell_1} & \alpha_{\ell_1} \end{bmatrix}, \lambda_{2\ell_1+1}, \ldots, \lambda_k \right\}, \\ X_1 & := & [\mathbf{x}_{1R}, \mathbf{x}_{1I}, \ldots, \mathbf{x}_{\ell_1 R}, \mathbf{x}_{\ell_1 I}, \mathbf{x}_{2\ell_1+1}, \ldots, \mathbf{x}_k] \end{array} \right. \tag{2.37}$$

and

$$\left\{\begin{array}{rcl} \Lambda_2 & := & \mathrm{diag}\Big\{\begin{bmatrix} \alpha_{k+1} & \beta_{k+1} \\ -\beta_{k+1} & \alpha_{k+1}\end{bmatrix}, \ldots, \begin{bmatrix} \alpha_{k+\ell_2} & \beta_{k+\ell_2} \\ -\beta_{k+\ell_2} & \alpha_{k+\ell_2}\end{bmatrix}, \lambda_{k+2\ell_2+1}, \ldots, \lambda_{2n}\Big\}, \\ X_2 & := & [\mathbf{x}_{(k+1)R}, \mathbf{x}_{(k+1)I}, \ldots, \mathbf{x}_{(k+\ell_2)R}, \mathbf{x}_{(k+\ell_2)I}, \mathbf{x}_{k+2\ell_2+1}, \ldots, \mathbf{x}_{2n}], \end{array}\right. \tag{2.38}$$

respectively. Since $X_1$ is to be updated, we may regard $X_2$ as the matrix $[C_0, P_0, N_0]$ in reference to (2.20). Recall that the corresponding $\widetilde{H} = \mathscr{H}([X_1, X_2], \mathrm{diag}\{\Lambda_1, \Lambda_2\})$ is block diagonal (See (2.17)). Partitioning $\widetilde{H}$ into two block diagonal submatrices according to the definition

$$\left\{\begin{array}{rcl} \widetilde{H}_1 & := & \begin{bmatrix} X_1 \\ X_1\Lambda_1\end{bmatrix}^\top \begin{bmatrix} \widetilde{C} & \widetilde{M} \\ \widetilde{M} & 0\end{bmatrix} \begin{bmatrix} X_1 \\ X_1\Lambda_1\end{bmatrix}, \\ \widetilde{H}_2 & := & \begin{bmatrix} X_2 \\ X_2\Lambda_2\end{bmatrix}^\top \begin{bmatrix} \widetilde{C} & \widetilde{M} \\ \widetilde{M} & 0\end{bmatrix} \begin{bmatrix} X_2 \\ X_2\Lambda_2\end{bmatrix}, \end{array}\right.$$

we should have the relationship

$$X_1\widetilde{H}_1^{-1}X_1^\top + X_2\widetilde{H}_2^{-1}X_2^\top = 0. \tag{2.39}$$

By Theorem 2.2.1, we know that

$$\left\{\begin{array}{rcl} \widetilde{M}^{-1} & = & X_1\Lambda_1\widetilde{H}_1^{-1}X_1^\top + X_2\Lambda_2\widetilde{H}_2^{-1}X_2^\top, \\ \widetilde{C} & = & -\widetilde{M}\big(X_1\Lambda_1^2\widetilde{H}_1^{-1}X_1^\top + X_2\Lambda_2^2\widetilde{H}_2^{-1}X_2^\top\big)\widetilde{M}, \\ \widetilde{K} & = & -\widetilde{M}\big(X_1\Lambda_1^3\widetilde{H}_1^{-1}X_1^\top + X_2\Lambda_2^3\widetilde{H}_2^{-1}X_2^\top\big)\widetilde{M} + \widetilde{C}\widetilde{M}^{-1}\widetilde{C}. \end{array}\right.$$

Assume that the structure of the newly measured eigeninformation $\{(\sigma_j, \mathbf{y}_j)\}_{j=1}^k$ is conformal to that of $\{(\lambda_j, \mathbf{x}_j)\}_{j=1}^k$. Let $(\Sigma, Y_1)$ denote the corresponding real-valued representation of $\{(\sigma_j, \mathbf{y}_j)\}_{j=1}^k$. On one hand, for a solvable MUP it is now known that we must have

$$Y_1 = X_1T \tag{2.40}$$

for some nonsingular matrix $T \in \mathbb{R}^{k\times k}$ [40, Theorem 4.1]. On the other hand, to avoid spill-over in the model updating, our theory demands a nonsingular matrix $\widehat{H} = \mathrm{diag}\{\widehat{H}_1, \widehat{H}_2\}$, with $\widehat{H}_1$ and $\widehat{H}_2$ having the same block structures respectively as those of $\widetilde{H}_1$ and $\widetilde{H}_2$, such that

$$Y_1\widehat{H}_1^{-1}Y_1^\top + X_2\widehat{H}_2^{-1}X_2^\top = 0, \tag{2.41}$$

even before the eigenvalues are updated. Upon substituting (2.40) into (2.41) and comparing

with (2.39), we find an obvious solution $\widehat{H}$ for (2.19) by choosing

$$\begin{cases} \widehat{H}_1 &:= T^\top \widetilde{H}_1 T \\ \widehat{H}_2 &:= \widetilde{H}_2. \end{cases} \tag{2.42}$$

By Theorem 2.2.2, we only need to make sure that $T$ is such that

$$\begin{cases} T^\top \widetilde{H}_1 T\Sigma \text{ is symmetric,} \\ X_1 T\Sigma T^{-1}\widetilde{H}_1^{-1}X_1^\top + X_2\Lambda_2\widetilde{H}_2^{-1}X_2^\top \text{ in nonsingular,} \end{cases} \tag{2.43}$$

then the MUP is solvable. In this case, the recipe in Theorem 2.2.1 gives rise to one particular solution to the MUP by

$$\begin{cases} M^{-1} &= X_1 T\Sigma T^{-1}\widetilde{H}_1^{-1}X_1^\top + X_2\Lambda_2\widetilde{H}_2^{-1}X_2^\top, \\ C &= -M\big(X_1 T\Sigma^2 T^{-1}\widetilde{H}_1^{-1}X_1^\top + X_2\Lambda_2^2\widetilde{H}_2^{-1}X_2^\top\big)M, \\ K &= -M\big(X_1 T\Sigma^3 T^{-1}\widetilde{H}_1^{-1}X_1^\top + X_2\Lambda_2^3\widetilde{H}_2^{-1}X_2^\top\big)M + CM^{-1}C. \end{cases}$$

Combining (2.40) with (2.44), we see that the update takes place in the following way:

$$\begin{cases} M^{-1} = \widetilde{M}^{-1} + X_1(T\Sigma T^{-1} - \Lambda_1)\widetilde{H}_1^{-1}X_1^\top, \\ C \;\; = M\big[\widetilde{M}^{-1}\widetilde{C}\widetilde{M}^{-1} - X_1(T\Sigma^2 T^{-1} - \Lambda_1^2)\widetilde{H}_1^{-1}X_1^\top\big]M, \\ K \;\; = M\big[\widetilde{M}^{-1}(\widetilde{K} - \widetilde{C}\widetilde{M}^{-1}\widetilde{C})\widetilde{M}^{-1} - X_1(T\Sigma^3 T^{-1} - \Lambda_1^3)\widetilde{H}_1^{-1}X_1^\top\big]M + CM^{-1}C \end{cases} \tag{2.44}$$

It is critically essential to note in formula (2.44) that the update from $(\widetilde{M}, \widetilde{C}, \widetilde{K})$ to $(M, C, K)$ *does not* involve any information about $(\Lambda_2, X_2)$ at all. This satisfies precisely the fundamental spirit in model updating.

## 2.5 Role of eigenvalues

Thus far, we have shown that eigenvalues play a very small role in the real symmetric IMPs. Only the structure of eigenvalues in $J$ is needed for the eigenvector completion process. The reconstructed $(M, C, K)$ literally can have arbitrary eigenvalues. In other words, *one cannot "see" the sound of a string*![5] What happens is that the structural constraint of $(M, C, K)$ being merely real and symmetric is too loose. Only when additional constraints are imposed upon $(M, C, K)$, the information of eigenvalues might become essential.

In this section we concentrate on one particular constraint usually entailed in structured QIEPs. We demonstrate the role of eigenvalues by considering the case when $M$ and $K$ are

[5] Likewise, twenty-six years after [104], it was answered that one cannot hear the shape of a drum [73].

required to be positive definite and $C$ positive semi-definite. An IMP with this kind of structure becomes a much harder problem.

Assume that zero is not an eigenvalue of the desirable quadratic matrix polynomial. Define the moments $\Gamma_j$, $j = -1, 0, 1, 2$, by

$$\Gamma_j := XJ^jH^{-1}X^\top. \tag{2.45}$$

By the fact that $HJ = (HJ)^\top$, all moments $\Gamma_j$ are symmetric. We have already seen in Theorem 2.2.1 that

$$\begin{cases} \Gamma_0 &= 0, \\ \Gamma_1 &= M^{-1}, \\ \Gamma_2 &= -M^{-1}CM^{-1}. \end{cases} \tag{2.46}$$

Post-multiplying both sides of (2.10) by $J^{-1}H^{-1}X$ and using (2.46), we obtain the relationship

$$\Gamma_{-1} = -K^{-1}, \tag{2.47}$$

which is another representation of K much less involved when comparing with the formula in (2.4).

The following theorem characterizes the positive semi-definiteness for $(M, C, K)$ in terms of moments which, in turn, relate to the eigenvalue matrix $J$ [114, 165].

**Theorem 2.5.1** *Given $(X, J)$, let $(M, C, K)$ be the symmetric coefficient matrices constructed from (2.4). Then*

1. *If $M \succ 0$, $K \succ 0$, and $C \succeq 0$, then all eigenvalues of $J$ have non-positive real part, $\Gamma_1$, $\Gamma_{-1}$ are nonsingular, and $\Gamma_2 \preceq 0$.*

2. *If all eigenvalues of $J$ have negative real part, $\Gamma_1$, $\Gamma_{-1}$ are nonsingular, and $\Gamma_2 \preceq 0$, then $M \succ 0$, $K \succ 0$, and $C \succeq 0$.*

**Proof.** The first part is straightforward. Suppose that $(M, C, K)$ has the described definiteness. Then by (2.46) and (2.47) it is obvious that $\Gamma_1$, $\Gamma_{-1}$ are nonsingular and $\Gamma_2 \preceq 0$. Suppose $(\lambda, \mathbf{u})$ is an eigenpair for $\mathfrak{Q}(\lambda) = M\lambda^2 + C\lambda + K$. Trivially, $(\lambda, \mathbf{u})$ also satisfies

$$\lambda^2\mathbf{u}^*M\mathbf{u} + \lambda\mathbf{u}^*C\mathbf{u} + \mathbf{u}^*K\mathbf{u} = 0,$$

from which we see that

$$\lambda = \frac{-\mathbf{u}^*C\mathbf{u} \pm \sqrt{(\mathbf{u}^*C\mathbf{u})^2 - 4(\mathbf{u}^*M\mathbf{u})(\mathbf{u}^*K\mathbf{u})}}{2(\mathbf{u}^*M\mathbf{u})},$$

implying that the real part of $\lambda$ cannot be positive.

The second part has already been established earlier in [114] by using a much heavy machinery. Since this part is most relevant to our inverse problem, we briefly outline a direct proof with ideas from [165] to make this presentation more self-contained. Define

$$\mathscr{V} := \tfrac{1}{2}\begin{bmatrix} K & 0 \\ 0 & M \end{bmatrix}, \quad \mathscr{W} := \begin{bmatrix} 0 & 0 \\ 0 & -C \end{bmatrix}. \tag{2.48}$$

By assumption, $\mathscr{V}$ is nonsingular and $\mathscr{W} \preceq 0$. It is easy to see that the triplet $(\mathscr{V}, \mathscr{C}, \mathscr{W})$ satisfies the Lyapunov equation

$$\mathscr{C}^\top \mathscr{V} + \mathscr{V}\mathscr{C} = \mathscr{W}. \tag{2.49}$$

Recall that $J$ represents precisely the spectrum of $\mathscr{C}$ which, by assumption, has no pure imaginary eigenvalues. It follows that that $\mathscr{V}$ has exactly $2n$ positive eigenvalues [26], implying that $K$ and $M$ must be positive definite. □

Solving the IMP with real, symmetric, and positive semi-definite $(M, C, K)$ therefore means that the eigenvalues of $J$, including those already prescribed and those to be completed, must be such that the matrix $XJ^2H^{-1}X^\top$ is negative semi-definite. The completion of both the eigenvectors and the eigenvalues simultaneously for structured IMPs is rather challenging task. To our knowledge, this area is still open for further research. We have developed some innovative techniques employing the notion from semi-definite programming and truncated $QR$ [55, 123], which will be discussed in Chapter 3 of this thesis. The following example demonstrates the necessity of completing both eigenpair $(X, J)$ simultaneously in order to solve this IMP with positive semi-definite coefficient matrices $(M, C, K)$.

**Example 2.5.1** *Consider the scenarios described in Example 2.3.1 where the complex eigenvectors are prescribed through the matrix $C_0$. Assume the prescribed eigenvalues are given by $J_0^C = \begin{bmatrix} -2 & 6 \\ -6 & -2 \end{bmatrix}$. Consider the first case $r = 1$ where $H_0^C$ and the two real eigenvectors $P_1$ and $N_1$ are to be constructed. Taking advantage of the free parameters already established in Example 2.3.1 we assume $n_1 = 2$, $n_2 = -1$ and $b_1 = 4$ so that the completed eigenvectors are given by*

$$X = \frac{1}{2}\begin{bmatrix} 0 & -\sqrt{2} & 2\sqrt{5} & 4 \\ \sqrt{2} & 0 & 0 & -2 \end{bmatrix}.$$

*Let the eigenvalues corresponding to $P_1$ and $N_1$ be noted as $\lambda_3$ and $\lambda_4$, respectively. Certainly, $\lambda_3$ and $\lambda_4$ must be real and negative. Additionally, in order that $XJ^2H^{-1}X^\top$ be negative semi-*

*definite, its principal minors must alternate signs, leading to the inequalities*

$$\begin{aligned} 80 + 5\lambda_3^2 - 4\lambda_4^2 &\leq 0, \\ -8000 + 80\lambda_4^2 - 400\lambda_3^2 - 5\lambda_3^2\lambda_4^2 &\geq 0. \end{aligned}$$

*The curves where these minors vanish are sketched in Figure 2.1. It can be checked that all points* $(\lambda_3, \lambda_4)$ *below the solid curve satisfy the inequalities and, therefore, can be used to complete the spectrum* $J$*.*

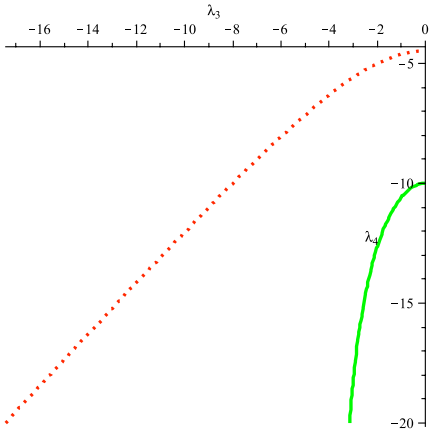


Figure 2.1: Curves where the principle minors of $XJ^2H^{-1}X^\top$ vanish.

*On the other hand, if we change to* $n_2 = -3$ *while keeping other parameters the same, the corresponding matrix*

$$X = \frac{1}{2}\begin{bmatrix} 0 & -\sqrt{2} & \sqrt{5}+\sqrt{21} & 4 \\ \sqrt{2} & 0 & \sqrt{5}-\sqrt{21} & -6 \end{bmatrix}.$$

*remains to be a solution to (2.23), but the determinant of* $XJ^2H^{-1}X^\top$ *is given by*

$$-58400 - 4000\sqrt{105} - (1648 + 224\sqrt{105})\lambda_3^2 - (688 - 64\sqrt{105})\lambda_4^2 - \frac{1}{2}(5\sqrt{105} + 73)\lambda_3^2\lambda_4^2,$$

*which obviously is always negative, implying that the spectrum* $J$ *can never be completed with this choice of* $P_1$ *and* $N_1$ *to make* $XJ^2H^{-1}X^\top$ *negative semi-definite.*

## 2.6 Conclusion

Our emphasis in this chapter is the important role played by this special block diagonal matrix $H$ whose structure is pre-determined by the desirable numbers $(t, r)$ of complex and real eigenvalues, but not by the actual eigenvalues. Exploiting this matrix as some free parameters, we can identify the coefficient matrices of real and symmetric IMPs by solving an under-determined homogeneous equation (2.23). Our approach has the advantages that any possible splitting $n = t + r$ and distribution $k = 2k_c + k_p + k_n \leq k_{max}$, where $k_{max}$ is given in (2.25), $k_c \leq t$, $k_p, k_n \leq r$, of complex and real eigenpairs with sign characteristic are allowable. We have shown that eigenvectors alone are sufficient to determine a solution whereas eigenvalues literally can be arbitrary. As an important consequence of our framework, the difficult task of modeling updating problem with no spill-over can easily be accomplished now.

Nevertheless, if $(M, C, K)$ are required to possess some additional structures such as stated positive definiteness, then properties of eigenvalues must be taken into account. In this cases, the completed eigenpair $(X, J)$ must be handled simultaneously.

# Chapter 3

# Inverse Problems with Partial Eigenpair Information

## 3.1 Overview

Vibration analysis often involves quadratic matrix polynomials. Following such a setting, there is always the predestinated inner-connectivity among its components and the compulsory non-negativity of its parameters. As such, coefficients of the quadratic matrix polynomial often are required to hold some intrinsic properties for any given vibration system. It is desirable that the reconstructing system preserves not only the specified eigeninofrmation but also certain intrinsic structure. Solving a structured QIEP is important and challenging both theoretically and practically. Because of the difficulty, especially because of the versatility of the structural constraints which vary from system to system, very few general results are available for the structured QIEPs in the literature. The earliest attempt seems to be made in [33] for a linearly linked system, which respects both the connectivity and the nonnegativity.

The crux of our work in this chapter is an approach that could be applied to structured QIEPs arising from general arbitrarily linked systems. To help achieving this goal, a mechanism is provided to systematically and automatically convert any given layout of linkage into a proper equality system. Since the measured eigeninformation is not always precise, the allowance for some leeway in the accuracy is expected. Based on a user-supplied tolerance, our algorithm can evaluate the consistency of a certain inequality system. If it is solvable within the tolerance, an approximate solution to the structured QIEP is computed. A posterior estimate of the residual error is returned output for deciding the suitability to accept or reject the constructed model. In this way, our approach greatly enhances the capacity of solving more complicated problems than those in [33]. Specifically, we develop a problem-independent rule for the structured QIEP and provide a error control strategy. We believe our contribution is innovative and should be

of great significance for structured QIEPs.

The organization of this chapter is as follows. In Section 3.2 we describe a general procedure for transforming any given layout of inner-connectivity into a linear equality system. This formulation follows various physical laws that govern the interaction among components. At present, a fundamental module for the mass-spring systems subject to Hooke's law for vibration is provided. We stress that our setting can handle models with arbitrary connectivity configuration, which effectively generalizes the work discussed in [33]. In order to get a nontrivial solution, it is necessary to have this equality system with rank deficiency. Hence, in Section 3.2.3 we derive a set of principles arising from properties of the rank deficiency and nonnegative constraints, inherent in the corresponding QIEP. By these principles, a structured QIEP is capable of being changed into a maximin problem and can be handled by many available optimization software packages, such as the "fminimax" in MATLAB. In Section 3.3, we show the tactics of handling a prescribed set of inexact eigeninformation. The presence of inexact data, which almost always happens in practice, has the consequence of dissolving the expected rank deficiency and causing inconsistency. To remedy this, in Section 3.3 we relax the equality system by generating an approximate but consistent subsystem. Also, a posterior estimate on the resulting residual error is offered to evaluate the quality of the reconstructed model. All these strategies and numerical exploration of some interesting perturbation results are discussed through the numerical tactic in Section 3.4.

## 3.2 Handling exact eigenpairs

It is not absolutely needed, but it will be instructive to keep vibration as basic models in the following discussion. Without loss of generality, assume that the prescribed eigenpairs $\{(\lambda_i, \mathbf{x}_i)\}_{i=1}^{k}$ are closed under complex-conjugation. Let $(\Lambda, X)$ represent the matrix representation of the prescribed eigenpairs, where

$$\Lambda := \operatorname{diag}\left\{ \begin{bmatrix} \alpha_1 & \beta_1 \\ -\beta_1 & \alpha_1 \end{bmatrix}, \ldots, \begin{bmatrix} \alpha_{k_c} & \beta_{k_c} \\ -\beta_{k_c} & \alpha_{k_c} \end{bmatrix}, \lambda_{2k_c+1}, \ldots, \lambda_k \right\} \in \mathbb{R}^{k\times k}, \tag{3.1}$$

$$X := [\mathbf{x}_{1R}, \mathbf{x}_{1I}, \ldots, \mathbf{x}_{k_cR}, \mathbf{x}_{k_cI}, \mathbf{x}_{2k_c+1}, \ldots, \mathbf{x}_k] \in \mathbb{R}^{n\times k}, \tag{3.2}$$

are as characterized in (2.15). Then the coefficient matrices $(M, C, K)$ for the general QIEPs should necessarily satisfy the algebraic system

$$MX\Lambda^2 + CX\Lambda + KX = 0_{n\times k}. \tag{3.3}$$

Clearly, the system (3.3) is linear in the unknowns $(M, C, K)$. Typically these coefficient matrices $(M, C, K)$ are composed of, in accordance with the underlying physical law, some positive physical parameters $\mathbf{u}$. Entries in $\mathbf{u}$ characterize the physical nature, such as mass, elasticity, or damping, among elements in the system. The inter-connectivity of elements defines the combined effect of these parameters which, in turn, characterizes an elaborate configuration of the coefficient matrices $(M, C, K)$. It is important to note that the practical structure for $(M, C, K)$ is much more complicated than the general requirements such as merely symmetry and positive semi-definiteness. The following two examples should demonstrate our ideas clearly.

**Example 3.2.1** *Consider a four-degrees-of-freedom vibration system whose masses, dampers, and springs are joined together as that depicted in Figure 3.1. Assume that the restoring force*

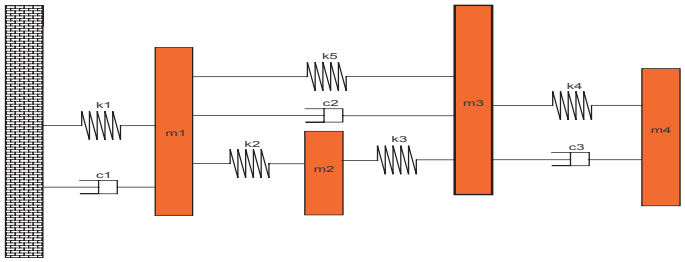


Figure 3.1: A four-degrees-of-freedom mass-spring system.

*follows Hooke's law and that the damping is negatively proportional to the velocity. Then, the corresponding coefficient matrices $(M, C, K)$ for the dynamical system (1.1) should be constructed as follows:*

$$M = \begin{pmatrix} m_1 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 \\ 0 & 0 & m_3 & 0 \\ 0 & 0 & 0 & m_4 \end{pmatrix}, C = \begin{pmatrix} c_1 + c_2 & 0 & -c_2 & 0 \\ 0 & 0 & 0 & 0 \\ -c_2 & 0 & c_2 + c_3 & -c_3 \\ 0 & 0 & -c_3 & c_3 \end{pmatrix},$$

$$K = \begin{pmatrix} k_1 + k_2 + k_5 & -k_2 & -k_5 & 0 \\ -k_2 & k_2 + k_3 & -k_3 & 0 \\ -k_5 & -k_3 & k_3 + k_4 + k_5 & -k_4 \\ 0 & 0 & -k_4 & k_4 \end{pmatrix}, \tag{3.4}$$

*where* $(M, C, K)$ *are parameterized with system parameters* $\mathbf{m}$*,* $\mathbf{c}$*, and* $\mathbf{k}$ *respectively such that.*

$$\begin{cases} \mathbf{m} &= (m_1, m_2, m_3, m_4)^\top, \\ \mathbf{c} &= (c_1, c_2, c_3)^\top, \\ \mathbf{k} &= (k_1, k_2, k_3, k_4, k_5)^\top, \end{cases} \tag{3.5}$$

It is true indeed that the structures stipulated above, together with the nonnegativity of the parameters, give rise to symmetry and positive semi-definiteness for these coefficient matrices, but the converse is not true. Merely assuming symmetry and positive semi-definiteness, as is usually done in the literature, is not enough to ensure the desired structure of connectivity and nonnegativity of physical parameters. Considerable work is needed when setting up and solving the QIEPs for mass-spring systems while respecting the inherent structure such as that specified in (3.4) and the nonnegativity of the parameters $(\mathbf{m}, \mathbf{c}, \mathbf{k})$.

**Example 3.2.2** *Depicted in Figure 3.2 is a closed-loop resonant circuit with three inductors, four resistors, and three capacitors. After applying Ohm's law and Kirchhoff's law to this*

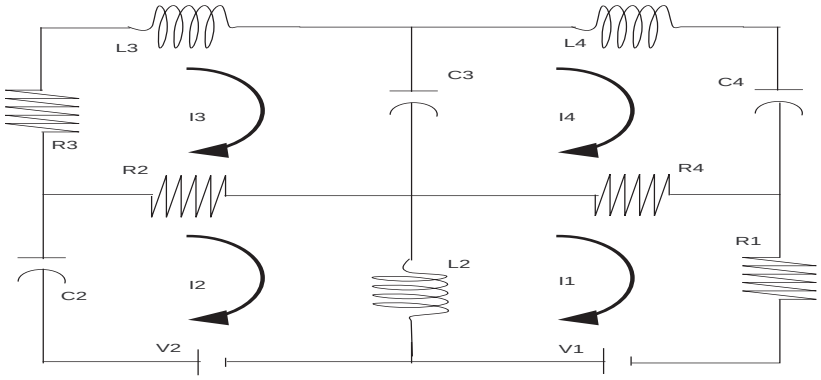


Figure 3.2: An RLC electronic network.

*connected circuit, the coefficient matrices should be structured as in (3.6). In contrast to the coefficient matrices in (3.4), we find that the coefficient matrices in this resonant circuit are neither positive-definite nor symmetric anymore.*

$$M = \begin{pmatrix} -L_1 & L_1 & 0 & 0 \\ L_1 & -L_1 & 0 & 0 \\ 0 & 0 & L_2 & 0 \\ 0 & 0 & 0 & L_3 \end{pmatrix}, C = \begin{pmatrix} 0 & R_2 & -R_2 & 0 \\ R_1 + R_4 & 0 & 0 & -R_4 \\ 0 & -R_2 & R_2 + R_3 & 0 \\ -R_4 & 0 & 0 & R_4 \end{pmatrix},$$
$$K = \begin{pmatrix} 0 & \frac{1}{C_1} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{C_2} & -\frac{1}{C_2} \\ 0 & 0 & -\frac{1}{C_2} & \frac{1}{C_2} + \frac{1}{C_3} \end{pmatrix} \tag{3.6}$$

The point we want to make is that the construction of coefficient matrices $(M, C, K)$ depends highly on the connection of its corresponding system and the underlying physical law, which vary from problem to problem. It appears necessary to modify the setup and even the numerical techniques while addressing different connectivity configurations. This undertaking also increases the difficulty in developing a general approach for QIEPs. This is precisely where our contribution in this research seems to become critical because we are able to handle QIEPs with inherently different structures by a unified scheme.

It would be informative if we continue using the mass-spring system to demonstrate the idea. Define $\mathbf{u} := (\mathbf{m}^\top, \mathbf{c}^\top, \mathbf{k}^\top)^\top$ to be the vector of $d$ parameters where $d = \dim(\mathbf{m}) + \dim(\mathbf{c}) + \dim(\mathbf{k})$. This vector $\mathbf{u}$ represents the basic variables in the QIEPs to be solved. In the context of a mass-spring system, the variables denote mass $\mathbf{m}$, damping $\mathbf{c}$ and stiffness $\mathbf{k}$, respectively. By linearity, we want to rewrite the linear algebraic system in (3.3) as a homogeneous system

$$A\mathbf{u} = 0_{nk\times 1}, \tag{3.7}$$

where $A$ is a matrix of size $nk \times d$. Our ultimate goal is to obtain a positive solution $\mathbf{u}$ for this linear equality system (3.7). The mathematical transformation to this equality system (3.7) is not a big issue, but two computational challenges are worthy our attention. Firstly, could the matrix $A$ be generated from prescribed connectivity configuration automatically? Secondly, can an algorithm be developed to have the capacity of dealing with inexact eigeninformation?

Theoretically, the homogeneous system (3.3) could be solved without any difficulty by using the notion of Kronecker product. The solution to the system (3.3) is equivalent to the solution of

$$\Big( \{(X\Lambda^2)^\top \bigotimes I\} \quad \{(X\Lambda)^\top \bigotimes I\} \quad \{X^\top \bigotimes I\} \Big)_{nk\times 3n^2} \begin{pmatrix} \mathrm{vec}(M) \\ \mathrm{vec}(C) \\ \mathrm{vec}(K) \end{pmatrix}_{3n^2\times 1} = 0_{nk\times 1}, \tag{3.8}$$

but such a process ignores the sparsity pattern embedded due to the inherent connectivity. We prefer to propose a mathematically equivalent mechanism that can build not only the matrix $A$ but also take advantage of the sparsity by tackling directly with the parameters. The potential of applications is too diverse to be covered thoroughly in this presentation. Thus, we shall illustrate the construction of this matrix $A$ through this four-degrees-of-freedom mass-spring system depicted in Figure 3.1. We do have developed a beta-version software package that can automatically handle the above mentioned task from any input of connectivity configuration and eigeninformation by users. We are continuing the expansion of its library to include various settings according to the underlying physical laws. We believe the approach outlined in the sequel can be generalized to different systems.

### 3.2.1 Inherent structure

Most vibration systems are modeled by following some kinds of general principles such as Hooke's law, Ohm's law, Kirchhoff's law and so on. On the other hand, many physical phenomena could be interpreted through a mass-spring system. A comprehensive study on constructing vibrating mass-spring systems therefore is of fundamental importance. Assuming that the restoring force follows Hooke's law and that the damping is negatively proportional to the velocity, the structure of the coefficient matrices $(M, C, K)$ in mass-spring systems could be well described through the following rules [103].

**Theorem 3.2.1** *Consider a mass-spring system with $d$ degrees of freedom where its motion is limited to one dimension. Let* $\mathbf{m}$, $\mathbf{c}$, *and* $\mathbf{k}$ *denote the vectors of masses, damping and stiffness coefficients, respectively. Then the inherent structure of matrices $(M, C, K)$ satisfies the following rules:*

1. *The mass matrix $M$ is a diagonal matrix with masses $(m_1, \ldots, m_d)$ along its diagonal.*

2. *The damping matrix $C$ is symmetric and positive semi-definite. If there are $\ell$ dampers, identified by $i_1, \ldots, i_\ell$, between the $p$-th mass and the $q$-th mass, then the entries $C_{pq}$ and $C_{qp}$ of the damping matrix are given by $-\sum_{s=1}^{\ell} c_{i_s}$, where $c_{i_s}$ is the damping coefficient of damper $i_s$. Otherwise, $C_{pq} = C_{qp} = 0$. If the $p$-th mass is connected to the dampers $j_1, \ldots, j_\ell$, then $C_{pp} = \sum_{s=1}^{\ell} c_{j_s}$.*

3. *The stiffness matrix $K$ is symmetric and positive semi-definite. If there are $\ell$ springs, identified by $i_1, \ldots, i_\ell$, between the $p$-th mass and the $q$-th mass, then the entries $K_{pq}$ and $K_{qp}$ of the stiffness matrix are given by $-\sum_{s=1}^{\ell} k_{i_s}$, where $k_{i_s}$ is the stiffness coefficient of spring $i_s$. Otherwise $K_{pq} = K_{qp} = 0$. If the $p$-th mass is connected to the springs $j_1, \ldots, j_\ell$, then $K_{pp} = \sum_{s=1}^{\ell} k_{j_s}$.*

Clearly, the matrices $(M, C, K)$ constructed earlier in Example 3.2.1 satisfy these rules in Theorem 3.2.1. Most important of all, such rules could be fully implemented into computers and we have done so in [55]. In other words, once the specified connectivity configuration is given, the intrinsic structure of coefficient matrices $(M, C, K)$ could be characterized automatically via a library of different rules similar to that described in Theorem 3.2.1.

### 3.2.2 Linear equality system

Once the inherent structure of $(M, C, K)$ is known, it remains to provide procedure transforming the linear algebraic equation (3.3) in the form $A\mathbf{u} = 0$. It is seemingly an easy mathematical job by using the Kronecker product and collecting like terms together. Our point, again, is to provide an efficient method which can simplify this tedious job, especially when the involved structure is complicated. This method of constructing matrix $A$ can be illustrated by using this four-degrees-of-freedom mass-spring system in Example 3.2.1, given $k$ prescribed eigenpairs, through the following steps.

Note that the first rows of $MX\Lambda^2$, $CX\Lambda$, and $KX$ could be written, respectively, as

$$\begin{cases} \begin{pmatrix} m_1 & 0 & 0 & 0 \end{pmatrix} X\Lambda^2 = \underbrace{\begin{pmatrix} m_1 & m_2 & m_3 & m_4 \end{pmatrix}}_{\mathbf{m}^\top} \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\mathfrak{M}_1} X\Lambda^2, \\ \begin{pmatrix} c_1 + c_2 & 0 & -c_2 & 0 \end{pmatrix} X\Lambda = \underbrace{\begin{pmatrix} c_1 & c_2 & c_3 \end{pmatrix}}_{\mathbf{c}^\top} \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\mathfrak{C}_1} X\Lambda, \\ \begin{pmatrix} k_1 + k_2 + k_5 & -k_2 & -k_5 & 0 \end{pmatrix} X = \underbrace{\begin{pmatrix} k_1 & k_2 & k_3 & k_4 & k_5 \end{pmatrix}}_{\mathbf{k}^\top} \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 \end{pmatrix}}_{\mathfrak{K}_1} X. \end{cases} \tag{3.9}$$

Continuing this process, it follows that

$$\left(\mathbf{m}^\top \mathbf{c}^\top \mathbf{k}^\top\right)_{1\times 12} \underbrace{\begin{pmatrix} \mathfrak{M}_1 X\Lambda^2 & \mathfrak{M}_2 X\Lambda^2 & \mathfrak{M}_3 X\Lambda^2 & \mathfrak{M}_4 X\Lambda^2 \\ \mathfrak{C}_1 X\Lambda & \mathfrak{C}_2 X\Lambda & \mathfrak{C}_3 X\Lambda & \mathfrak{C}_4 X\Lambda \\ \mathfrak{K}_1 X\Lambda & \mathfrak{K}_2 X\Lambda & \mathfrak{K}_3 X\Lambda & \mathfrak{K}_4 X \end{pmatrix}}_{A^\top_{12\times 4k}} = 0_{1\times 4k}. \tag{3.10}$$

In the above, the operators $\mathfrak{M}_i$, $\mathfrak{C}_i$, and $\mathfrak{K}_i$ are predetermined as a consequence from Figure 3.1. In general, given any linear structured matrices $M$, $C$, and $K$ of size $n \times n$ with $k$ prescribed eigenpairs, each row of the product of $MX\Lambda^2 + CX\Lambda + KX$ can be constructed from operator matrices $\mathfrak{M}_i$, $\mathfrak{C}_i$, and $\mathfrak{K}_i$ of sizes $\dim(\mathbf{m}) \times n$, $\dim(\mathbf{c}) \times n$, and $\dim(\mathbf{k}) \times n$ whose true contents are determined from the underlying connectivity configuration and physical law. In this way, instead of (3.3), we obtain a linear system (3.7) in terms of the parameters $(\mathbf{m}, \mathbf{c}, \mathbf{k})$ where the coefficient matrix $A$ is given by

$$A = \begin{pmatrix} (\mathfrak{M}_1 X\Lambda^2)^\top & (\mathfrak{C}_1 X\Lambda)^\top & (\mathfrak{K}_1 X)^\top \\ & \vdots & \\ (\mathfrak{M}_n X\Lambda^2)^\top & (\mathfrak{C}_n X\Lambda)^\top & (\mathfrak{K}_n X)^\top \end{pmatrix}_{nk\times d}. \tag{3.11}$$

Being characterized directly in terms of the physical parameters, the matrix $A$ is of size mush smaller than that derived in (3.8). That is, the condensed expression will have increased capability of handling larger scale problems. We quickly point out that the above expression is only for explanatory purpose. To generalize this idea to any prescribed configuration, all we need to have is a few indices specifying the locations where the connectivity is taken place. More programming details about this development is given in [55]. Our software package accepts two different input modes. One picks out these indices from an input file such as text interface. The other generates the the indices directly from the connectivity configuration and the underlying physical law.

To demonstrate the input mode, below we summarize the rules by which the indices for constructing the operator matrices $\mathfrak{M}_i, \mathfrak{C}_i, \mathfrak{K}_i$ for a general mass-spring system can be generated directly from the connectivity configuration without making any reference to the inherent structure in $(M, C, K)$.

**Theorem 3.2.2** *Consider a mass-spring system with $n$ degrees of freedom where the motion is limited to one dimension. Assume that* $\dim(\mathbf{m})$, $\dim(\mathbf{c})$, *and* $\dim(\mathbf{k})$ *represent the number of masses, dampers, and springs, respectively. Then for $1 \le p, q \le n$, the operator matrices are defined as follows:*

1. *Each $\mathfrak{M}_p$ is of size* $\dim(\mathbf{m}) \times n$. *Its entries are all zero except 1 at the $(p, p)$ position.*

2. *Each* $\mathfrak{C}_p$ *is of size* $\dim(\mathbf{c}) \times n$. *Its entries are all zero with the following exceptions:*

   - *If there are* $\ell$ *dampers, identified by* $i_1, \ldots, i_\ell$, *linking the p-th mass to the q-th mass, then the q-th column of the matrix* $\mathfrak{C}_p$ *has -1 at its* $i_1, \ldots, i_\ell$ *entries. Similarly, the p-th column of* $\mathfrak{C}_q$ *has -1 at its* $i_1, \ldots, i_\ell$ *entries.*
   - *If the p-th mass is connected to the dampers* $j_1, \ldots, j_\ell$, *then the p-th column of* $\mathfrak{C}_p$ *has 1 at its* $j_1, \ldots, j_\ell$ *entries.*

3. *Each* $\mathfrak{K}_p$ *is of size* $\dim(\mathbf{k}) \times n$. *Its entries are all zero with the following exceptions:*

   - *If there are* $\ell$ *springs, identified by* $i_1, \ldots, i_\ell$, *linking the p-th mass to the q-th mass, then the q-th column of* $\mathfrak{K}_p$ *has -1 at its* $i_1, \ldots, i_\ell$ *entries. Similarly, the p-th column of* $\mathfrak{K}_q$ *has -1 at its* $i_1, \ldots, i_\ell$ *entries.*
   - *If the p-th mass is connected to the springs* $j_1, \ldots, j_\ell$, *then the p-th column of* $\mathfrak{K}_p$ *has 1 at its* $j_1, \ldots, j_\ell$ *entries.*

### 3.2.3 Linear inequality system

We have addressed the issue of structural constraints in a QIEP. Now we have to stress that the solvability of a QIEP hinges upon the existence of positive solution $\mathbf{u}$ for the system $A\mathbf{u} = 0$. Before we propose our approach for solving this equality system, two important issues must be considered first.

Firstly, when the output solution $\mathbf{u}$ appears to contain some infinitesimal entries, we have to discern the true meaning of a numerical small number. Suppose, for example $m_1 = 10^{-10}$. It might mean that the value $m_1$ is recovered with a small physical quantity or the smallness of $m_1$ reflects the machine zero (up to the prescribed stopping criterion). For the former case, we might just rescale the whole quantity of $\mathbf{u}$, say, by changing into a different physical unit. Otherwise, we must be cautious about this numerical answer since it implies that the system is degenerate in the sense that one component should "disappear". This would completely changes the configuration of inter-connectivity.

Secondly, it is well recognized in numerical computation that we cannot fully trust the positivity of an infinitesimal number due to the ramification of round-off errors, to the effect that most of the optimization tools developed for inequality constrained problems cannot handle strict inequalities well when a solution is nearby the "boundary". Likewise, we are going to solve the equality system $A\mathbf{u} = 0$ with a nonnegative constraint, $\mathbf{u} \geq 0$, not $\mathbf{u} > 0$.

Suppose for the moment that the prescribed eigenpairs $(\Lambda, X)$ are exact. For the system (3.7) to have a nontrivial solution, the matrix $A$ must be rank deficient. Let $s := \text{rank}(A)$. Then the *compact QR factorization* with column pivoting for the matrix $A$ (See Appendix A)

is written as

$$A = QRP, \tag{3.12}$$

where $Q \in \mathbb{R}^{nk \times s}$ has orthogonal column and $P \in \mathbb{R}^{d \times d}$ is a permutation matrix such that

$$R = (S, T) \in \mathbb{R}^{s \times d} \tag{3.13}$$

is upper "trapezoidal" matrix with $S \in \mathbb{R}^{s \times s}$ being upper triangular, nonsingular, and having non-increasing absolute values along its diagonal. Define

$$P\mathbf{u} := (\overline{\mathbf{u}}_1^\top, \overline{\mathbf{u}}_2^\top)^\top, \tag{3.14}$$

with $\overline{\mathbf{u}}_1 \in \mathbb{R}^s$. Then the following two expressions are clearly equivalent

$$A\mathbf{u} = 0 \Longleftrightarrow \overline{\mathbf{u}}_1 = -S^{-1}T\overline{\mathbf{u}}_2. \tag{3.15}$$

By (3.15), we have thus recast the structured QIEP into an inequality system.

**Theorem 3.2.3** *Consider a structured QIEP with prescribed eigenpairs $(\Lambda, X)$ and connectivity configuration. Let $S$ and $T$ be the two matrices associated to the matrix $A$ as above. Then finding the nonnegative parameters $\mathbf{u} \in \mathbb{R}^n$ for the structured QIEP is equivalent to solving the inequality system*

$$\overline{\mathbf{u}}_2 \geq 0, \quad S^{-1}T\overline{\mathbf{u}}_2 \leq 0 \tag{3.16}$$

*for $\overline{\mathbf{u}}_2 \in \mathbb{R}^{n-s}$.*

Once the solution is found, it can be scaled by any positive constant. It is thus sufficient to assume that entries of $\overline{\mathbf{u}}_2$ are limited to the interval $[0, 1]$. Of course a trivial solution $\overline{\mathbf{u}}_2 = 0$ is not desirable and should be discarded. This inequality system (3.16) is a classical convex problem which has been widely discussed in the literature, including the well known Farkas Lemma [20], the von Neumann theorem [162], and the Ky Fan theorem [60]. We further reformulate the inequality system (3.16) by considering the maximin problem:

$$\max_{0 \leq \overline{\mathbf{u}}_2 \leq \mathbf{1}} \min \left(-S^{-1}T\overline{\mathbf{u}}_2\right), \tag{3.17}$$

where the MIN function is taken over all entries of the vector $-S^{-1}T\overline{\mathbf{u}}_2$. Since the objective function $\min\left(-S^{-1}T\overline{\mathbf{u}}_2\right)$ is concave over a convex feasible set, the solution to (3.17) is unique and global. For a solvable QIEP, it is necessary that the inequality system (3.16) has a nonnegative

solution. Consequently, the solvability of inequality system (3.16) depends on a nonnegative objective value in (3.17).

Note that the rank deficiency of the matrix $A$ comes from the *unrealistic* assumption that the prescribed eigenpairs are exact for some (unknown) quadratic matrix polynomials. Without the rank deficiency, the nontrivial nonnegative solution for $A\mathbf{u} = 0$ will not exist. In practice, however, prescribed eigeninformation is most likely inexact. The matrix $A$ constructed with the inexact data is generically of full rank, so the procedure described above cannot go through. The method of how to overcome this difficulty will be discussed in the next section.

## 3.3 Handling inexact eigenpairs

One of the practical reasons for considering QIEPs is to update or reconstruct quadratic models based on newly measured eigeninformation. It is very likely for the very same reason that the new eigeninformation used for the update or reconstruction is inexact in itself. The repercussion is that the resulting matrix $A$ in (3.7) will be of full rank or some obscured numerical rank. Without any treatment, the solution of $A\mathbf{u} = 0$ would be either zero or undesirable. As the eigeninformation still contains useful substance inside, it is often the case that an approximate solution satisfying physical constraints such as nonnegative is acceptable for engineering applications. Our goal in this section is to provide an approximate solution with a posterior error estimation.

In order to derive an approximate solution for (3.7) , the most direct approach is to consider the quadratic programming (QP) problem:

$$\begin{aligned} \text{minimize} \quad & \|A\mathbf{u}\|_2^2 \\ \text{subject to} \quad & \mathbf{u} \geq \mathbf{1}. \end{aligned} \tag{3.18}$$

The constraint $\mathbf{u} \geq \mathbf{1}$ is meant to avoid the trivial solution $\mathbf{u} = 0$. A substantial difference among elements of $\mathbf{u}$, however, might be an indication that the smallest element is numerically zero and cause degeneracy. For an optimal solution $\mathbf{u}^*$, the observation of a small residual $\|A\mathbf{u}^*\|_2$ "might" suggest heuristically that the problem is approximately solved. Although (3.18) provides an intuitive idea for solving (3.7), its calculation involves the full length $d$ of $\mathbf{u}$. We have a better approach that involves only $d - s$ variables and provides a posterior estimate which could be used to discrete whether to accept or reject the constructed model.

We begin by introducing the *full* QR factorization with column pivoting of the matrix $A$ such that

$$A = QRP, \tag{3.19}$$

where $Q \in \mathbb{R}^{nk\times nk}$ has orthogonal column and $R \in \mathbb{R}^{nk\times d}$ is an upper trapezoidal matrix with diagonal elements arranged in terms of non-increasing absolute values by the permutation matrix $P$. Define

$$\overline{\mathbf{u}} := P\mathbf{u}. \tag{3.20}$$

Trivially we have the following equivalent relationship:

$$A\mathbf{u} = 0 \Longleftrightarrow R\overline{\mathbf{u}} = 0, \tag{3.21}$$

Due to inexact eigeninformation, generally the matrix $A$ is of full rank. This implies that $\mathbf{u} = 0$ is the only solution for $A\mathbf{u} = 0$, which is of little value in practice. Hence the introduction of some low rank approximations to the matrix $R$ is necessary in order to obtain some meaningful nontrivial solutions.

Given a predetermined threshold $\epsilon$, let $R_\epsilon \in \mathbb{R}^{t\times d}$ be the submatrix of $R$ made of $t$ rows of $R$ whose diagonal elements are greater than $\epsilon$. Then a nonnegative nontrivial solution to $R_\epsilon\overline{\mathbf{u}} = 0$ is regraded as an approximate solution to $R\overline{\mathbf{x}} = 0$. The choice of $\epsilon$ is often based on the trial-and-error basis. To bring forth meaningful truncation, the value of $\epsilon$ should be large enough, otherwise, $R_\epsilon\overline{\mathbf{u}} = 0$ still has only trivial solution. But, it should also be small enough to avoid losing too much information about $A$. Assume $t < d$. Define $R_\epsilon = (S_\epsilon, T_\epsilon)$, where $S_\epsilon \in R^{t\times t}$ is invertible, and partition $\overline{\mathbf{u}} = (\overline{\mathbf{u}}_1^\top, \overline{\mathbf{u}}_2^\top)^\top$ accordingly. From the equivalence

$$R_\epsilon\overline{\mathbf{u}} = 0 \Longleftrightarrow \overline{\mathbf{u}}_1 = -S_\epsilon^{-1}T_\epsilon\overline{\mathbf{u}}_2, \tag{3.22}$$

our current intention is to solve the inequality system

$$\overline{\mathbf{u}}_2 \geq 0, \quad -S_\epsilon^{-1}T_\epsilon\overline{\mathbf{u}}_2 \leq 0 \tag{3.23}$$

for some $\overline{\mathbf{u}}_2 \in \mathbb{R}^{d-t}$. From this point on, the maximin ideas outlined in Section 3.2.3 could be used to solve (3.23).

Note that if $\overline{\mathbf{u}}_\epsilon^*$ is a nonnegative solution to the truncated linear system $R_\epsilon\overline{\mathbf{u}} = 0$, the vector $\mathbf{u}_\epsilon^* := P^\top\overline{\mathbf{u}}_\epsilon^*$ is considered to be an approximate nonnegative solution to the original system $A\mathbf{u} = 0$. The following theorem gives a justification for the notion of this truncation.

**Theorem 3.3.1** *For a matrix $A \in \mathbb{R}^{nk\times d}$, let $A = QRP$ denote its* QR factorization with column pivoting*, where $Q \in \mathbb{R}^{nk\times nk}$ has orthogonal column and $R \in \mathbb{R}^{nk\times d}$ is an upper trapezoidal matrix with diagonal elements arranged in descending absolute value by the permutation matrix $P$. Given a positive number $\epsilon$, let $R_\epsilon$ denote the truncated submatrix of $R$ consisting of rows of $R$ whose diagonal elements are greater than $\epsilon$. Suppose $\overline{\mathbf{u}}_\epsilon$ is a solution to the system $R_\epsilon\overline{\mathbf{u}} = 0$.*

*(For our application, we are interested in a nonnegative solution $\overline{\mathbf{u}}_\epsilon$, but that is not needed in this theorem.) Define $\mathbf{u}_\epsilon := \mathbf{P}^\top \overline{\mathbf{u}}_\epsilon$. Then*

$$\frac{\|A\mathbf{u}_\epsilon\|_2}{\|\mathbf{u}_\epsilon\|_2} = \mathcal{O}(\epsilon)$$

**Proof.** For any integer $1 \le j \le nk$, let $R_{[j]}$ denote the lower right submatrix of $R$ by deleting its first $j-1$ rows and columns. The permutation matrix $P$ is chosen so that $|r_{jj}|$ is greater than or equals to the 2-norm of any columns in $R_{[j]}$. Consequently, it is true that

$$\|R_{[j]}\|_2 \le \sqrt{d-j+1}\|R_{[j]}\|_1 \le \sqrt{d-j+1}\sqrt{nk-j+1}|r_{jj}|.$$

Note that for a given $\epsilon$, the only possible nonzero elements in $R - R_\epsilon$ would be those in $R_{[i+1]}$ where $i$ is the smallest integer such that $|r_{i+1,i+1}| < \epsilon$. Since $\overline{\mathbf{u}}_\epsilon$ is a solution of $R_\epsilon \overline{\mathbf{u}} = 0$ and $\|\overline{\mathbf{u}}_\epsilon\|_2 = \|\mathbf{u}_\epsilon\|_2$, we have

$$\begin{aligned} \|A\mathbf{u}_\epsilon\|_2 = \|R\overline{\mathbf{u}}_\epsilon\|_2 &= \|R\overline{\mathbf{u}}_\epsilon - R_\epsilon\overline{\mathbf{u}}_\epsilon + R_\epsilon\overline{\mathbf{u}}_\epsilon\|_2 \\ &\le \|R - R_\epsilon\|_2 \cdot \|\overline{\mathbf{u}}_\epsilon\|_2 + \|R_\epsilon\overline{\mathbf{u}}_\epsilon\|_2 \\ &< \sqrt{d-i}\sqrt{nk-i}\|\mathbf{u}_\epsilon\|_2\epsilon. \end{aligned}$$

□

In most cases, we have no idea of how to pick up a suitable $\epsilon$ to truncate the matrix. The larger the $\epsilon$ is, the less precision the solution will be. On the other hand, too small $\epsilon$ might result in no feasible solution. The following strategy provides a dynamic way of automatically truncating the matrix with a self-examining truncation strategy.

Assume that the matrix $R$ has numerical rank $p$. For $1 \le i \le p$, let $R^{[i]}$ denote the submatrix of the first $i$ rows of $R$. For each $i$, write $R^{[i]} = (S_i, T_i)$ with $S_i \in \mathbb{R}^{i\times i}$ and partition $\overline{\mathbf{u}} = (\overline{\mathbf{u}}_1^\top, \overline{\mathbf{u}}_2^\top)^\top$ with $\overline{\mathbf{u}}_1 \in \mathbb{R}^i$. Starting with $i = p$ and gradually decreasing to $i = 1$, we solve the maximin problem

$$\max_{0\le\overline{\mathbf{u}}_2\le\mathbb{1}} \min \left(-S_i^{-1}T_i\overline{\mathbf{u}}_2\right) \tag{3.24}$$

over $\overline{\mathbf{u}}_2 \in \mathbb{R}^{n-i}$ successively. Terminate the process at the first (largest) $i$ when a nonnegative objective value is found. This approach provides the most suitable truncation while keeping the best integrity of $R$ and seeking consistency.

We truncate the matrix $R$ only when it absolutely cannot compromise with the consistency. We also take advantage of the fact that the maximin problem returns a global solution. The essence in both approaches described above is to approximate the original system $A\mathbf{x} = 0$ by a

lower rank matrix in exchange for nonnegative solutions. The latter approach differs from the former approach in that it avoids truncating the matrix $R$ too much. Assume that the dynamic process stops at $i$. Then for any $\epsilon < r_{ii}$ the system (3.23) is inconsistent. On the other hand, for $\epsilon = r_{ii}$, we have $R_\epsilon = R^{[i]}$. Clearly, for any $\epsilon > r_{ii}$, the truncated system $R_\epsilon \overline{\mathbf{u}} = 0$ has a nonnegative solution but a relatively larger residual.

Our algorithm checks to determine $R_\epsilon$ so that $R_\epsilon \overline{\mathbf{u}} = 0$ has a nonnegative solution. In order to achieve consistency, it is possible that the approximate matrix $R_\epsilon$ (or $R^{[i]}$) has a fairly low rank. That is, the value of $\epsilon$ might be relatively large. However, once such a nonnegative solution is found to exist, Theorem 3.3.1 implies that the ratio of the norm of the residual vector to that of the nonnegative solution is independent of scaling. The scaling is only for computational convenience and has no effect on this relative error. In other words, our reconstructed parameters not only guarantee the maintenance of connectivity constraint but also estimate $\|M^*X\Lambda^2 + C^*X\Lambda + K^*X\|_F = \|A\mathbf{u}^*\|_2$, which is of order $\mathcal{O}(\epsilon)$ relative to $\|\mathbf{u}^*\|_2$ for further discernment of whether accepting the reconstruction or not.

## 3.4 Numerical experiments

Based on the computational framework we have described above, we have developed a software package, named Opt4QIEP, that is able to reckon nonnegative parameters for the structured QIEPs [55]. A conceptual organization of the algorithm is outlined in Algorithm 1. We emphasize that Algorithm 1 is capable of handling almost all arbitrary connectivity configuration and that we are continuing the expansion of its library modules. For demonstration purpose, we focus on the mass-spring system specified in Figure 3.1 and illustrate some interesting points worthy of attention.

To begin with, we randomly generate positive physical parameters,

$$\begin{aligned} \mathbf{m} &= [0.42052, 0.95581, 0.94875, 0.65968]^\top, \\ \mathbf{c} &= [0.55187, 1.00000, 0.91316]^\top, \\ \mathbf{k} &= [0.71675, 0.90909, 0.73377, 0.38006, 0.32200]^\top, \end{aligned}$$

where the largest parameter has been normalized to unity. Using these parameter values we construct the quadratic polynomial $\lambda^2 M + \lambda C + K$ corresponding to the configuration specified in (3.4). The resulting QEP has fours pairs of complex conjugate eigenvalues and eigenvectors. For numerical testing, we assume that the following two eigenpairs (and their complex conjugates),

$$\lambda_1 = -0.2560344023 + 1.5586653651i, \quad \lambda_2 = -0.0775078020 + 0.3777316241i,$$

**Algorithm 1**: QIEP Algorithm

**Input**:

- The desirable connectivity configuration.
- Partial eigeninformation in the form of eigenpairs closed under complex conjugation.

**Output**:

- If the QP approach has a feasible solution, or the maximin approach returns a nonnegative objective value, the structured QIEP is deemed solvable within a specified tolerance on its residual.
- Otherwise, the structured QIEP has no solution.

**begin**
  Identify the inherent stsructure of the coefficient matrices $(M, C, K)$;
  Prepare $X\Lambda^2$ and $X\Lambda$;
  Transform the equation (3.3) into $A\mathbf{x} = 0$ with variable $\mathbf{x}$ denoting parameters to be reconstructed;
  Apply either the QP or the maximin techniques to search for an approximate nonnegative solution;
**end**

$$\mathbf{u}_1 = \begin{pmatrix} -0.4849049878 + 1.2935348932i \\ -1.5045182718 - 1.2912238078i \\ 0.5225204188 + 1.0465626189i \\ 1 \end{pmatrix}, \mathbf{u}_2 = \begin{pmatrix} 0.4687637938 + 0.1026944256i \\ 0.6690611955 + 0.1297948996i \\ 0.8080014019 + 0.0892317746i \\ 1 \end{pmatrix},$$

form the *exact* eigeninformation $(\Lambda, X)$. Note that the last entry of each eigenvector has been normalized to unity and exactly *five* digits have been purposefully assigned to the physical parameters. Strictly speaking, the exact eigenpairs should be accurately represented at least up to the machine precision instead of just the first ten digits. However, this less accurate representation of prescribed eigeninformation is precisely our intention so as to experment with the effectiveness of our algorithm upon various scenarios of inexact data.

**Example 3.4.1** *Suppose first that* $(\lambda_1, \mathbf{u}_1)$ *(and its complex conjugate) is the only prescribed eigenpair. Note that this eigenpair is known a priori to have accuracy only up to the tenth digit. With this slightly perturbed eigeninformation, our algorithm does find out a nonnegative*

*solution. After being rounded to five digits, its computed parameters are equal to*

$$\begin{aligned}\mathbf{m}^* &= [0.42052, 0.95581, 0.94875, 0.65968]^\top,\\ \mathbf{c}^* &= [0.55187, 1.00000, 0.91316]^\top,\\ \mathbf{k}^* &= [0.71675, 0.90909, 0.73377, 0.38006, 0.32200]^\top.\end{aligned}$$

*In other words, these computed parameters are consistent with the original physical ones up to the fifth digit. The residual* $\|M^*X\Lambda^2 + C^*X\Lambda + K^*X\|_F$ *with the computed physical parameters is around* $10^{-11}$*, justifying a successful calculation.*

In most cases, inverse eigenvalue problems are ill-posed and typically do not possess a unique solution. To our surprise, the computed parameters in Example 3.4.1 are in agreement with the original physical parameters up to the fifth digit. It is worthy of further investigation to see whether this coincidence is definitely attributed to the normalization of the largest parameter to unity, the nonnegative constraints, and the inequality setting which satisfies the feature that a minimizer, if exists, is necessarily a global minimizer.

**Example 3.4.2** *Our approach works indiscriminately with different cardinalities of prescribed eigenpairs. Suppose now we are given two eigenpairs* $(\lambda_1, \mathbf{u}_1)$ *and* $(\lambda_2, \mathbf{u}_2)$ *and their conjugates simultaneously. It is possible that the prescribed eigeninformation is overdone. Nonetheless, we are not surprised to obtain the same computed parameters with residual* $\|M^*X\Lambda^2 + C^*X\Lambda + K^*X\|_F \approx 10^{-11}$*. Though this extra eigeninformation* $(\lambda_2, \mathbf{u}_2)$ *increases the complexity of matrix A, it seems to have no impact on the final result.*

**Example 3.4.3** *In this experiment, we further perturb the original exact eigenpairs to the extent that the prescribed eigenpairs are given by*

$$\widetilde{\lambda}_1 = -0.25603 + 1.55867i, \quad \widetilde{\lambda}_2 = -0.07751 + 0.37773i,$$

$$\widetilde{\mathbf{u}}_1 = \begin{pmatrix} -0.48490 + 1.29353i \\ -1.50452 - 1.29122i \\ 0.52252 + 1.04656i \\ 1 \end{pmatrix}, \quad \widetilde{\mathbf{u}}_2 = \begin{pmatrix} 0.46876 + 0.10269i \\ 0.66906 + 0.12979i \\ 0.80800 + 0.08923i \\ 1 \end{pmatrix}.$$

*That is, the order of accuracy of the prescribed eigenpairs is only at the fifth decimal. Typically, we are not able to judge a priori the accuracy of a prescribed eigeninformation. Thus, it could not guarantee that any quadratic matrix polynomial structured as in (3.4) could have* $(\widetilde{\lambda}_1, \widetilde{\mathbf{u}}_1)$ *and* $(\widetilde{\lambda}_2, \widetilde{\mathbf{u}}_2)$ *as its eigenpairs. This is exactly the difficulty of solving the structured QIEP for nonnegative solutions.*

*Upon applying our algorithm, however, we do find a nonnegative solution*

$$\begin{aligned}\mathbf{m}^* &= [0.42053, 0.95585, 0.94878, 0.65971]^\top,\\ \mathbf{c}^* &= [0.55188, 1.00000, 0.91319]^\top,\\ \mathbf{k}^* &= [0.71677, 0.90911, 0.73383, 0.38009, 0.32200]^\top,\end{aligned}$$

*by which the residual* $\|\mathbf{M}^*\widetilde{\mathbf{X}}\widetilde{\mathbf{\Lambda}}^2+\mathbf{C}^*\widetilde{\mathbf{X}}\widetilde{\mathbf{\Lambda}}+\mathbf{K}^*\widetilde{\mathbf{X}}\|_F$ *of the reconstructed quadratic pencil is of the order* $10^{-6}$*. This residual also provides a numerical justification of our theory in Theorem 3.3.1.*

Again, we generally have no clue in telling the feasibility of prescribed eigenpairs. Also, we are not sure whether the prescribed eigenpairs, $(\widehat{\lambda}_1, \widehat{\mathbf{u}}_1)$ and $(\widehat{\lambda}_2, \widehat{\mathbf{u}}_2)$, satisfy a specifically structured QIEP. What we do know for sure is that if a solution or a nearby solution ever exists, our method will find it with high sensitivity, otherwise it will indicate either a large residual or no solution with high specificity.

**Example 3.4.4** *When the prescribed eigenpairs are infeasible, we would like to know that our method is able to indicate that this QIEP is not solvable. To demonstrate the situation, consider the randomly generated eigenpairs,*

$$\widehat{\lambda}_1 = 0.6068 + 0.8913i, \quad \widehat{\lambda}_2 = 0.4860 + 0.7621i,$$

$$\widehat{\mathbf{v}}_2 = \begin{pmatrix} 0.0185 + 0.4057i \\ 0.8214 + 0.9355i \\ 0.4447 + 0.9169i \\ 1 \end{pmatrix}, \quad \widehat{\mathbf{v}}_2 = \begin{pmatrix} 0.7919 + 0.8936i \\ 0.9218 + 0.0579i \\ 0.7382 + 0.3529i \\ 1 \end{pmatrix}.$$

*Applying our method to the above randomly generated eigenpairs, we do obtain a nonnegative "approximate" solution, but the corresponding residuals of the quadratic matrix polynomial are* 1.2885 *and* 0.7667*, respectively, at each eigenpair. This would obviously indicate that the reconstructed model is not acceptable.*

This result seems to imply that for structured coefficient matrices $(\mathbf{M}, \mathbf{C}, \mathbf{K})$, the corresponding eigenvalues and eigenvectors cannot be arbitrary. The eigeninformation might preserve certain structure and worthy of further investigation. Our algorithm could serve as a numerical tool to boost the investigation in this area.

## 3.5 Conclusion

Just as the QEPs arise in many applications, so are the QIEPs fundamental to many fields of disciplines. Because the specifics of the dynamical systems deviate from each other in different environments, the corresponding coefficient matrices need to satisfy different kinds of structural constraints. The diversity in structure, together with the imposition of nonnegativity on intrinsic parameters, makes solving QIEPs quite challenging. To our knowledge, there is no theory thus far that can comprehensively resolve these difficulties.

The main contribution of this chapter is to provide a general purpose and robust numerical technique, based on properly established inequality systems and suitable optimization solvers, for solving structured QIEPs. The setting is to find out a global minimizer as the approximate solution. Based on an estimation of residual, this method has high capacity in determining numerically whether a QIEP is solvable or not. Also, it has no restriction on how many eigenpairs should be given and the ability to solve almost all kinds of connectivity configurations.

# Chapter 4

# Inverse Problems with Structured Models

## 4.1 Overview

In our software package Opt4QIEP, we have already implemented the notions of automated structure generation to handle various connectivity configurations and error correction to deal with inexact eigeninformation. In yet another advance, we now propose a powerful optimization technique for solving QIEPs. Semi-definite programming (SDP) is a relatively new field of optimization techniques arising from convex optimization. It is known for its capacity of handing an extraordinarily wide range of problems. In this chapter, we are going to describe an innovative application of SDP techniques to structured QIEPs. We repeat that by a structured QIEP, it means the construction of real coefficient matrices $M$, $C$ and $K$ satisfying $k$ prescribed eigenpairs $\{(\lambda_j, \mathbf{x}_j)\}_{j=1}^{k}$ and meeting certain distinctive conditions imposed upon their respective structures. We have already seen that the prescribed spectral conditions transform into a linear algebraic system (3.3) for the coefficient matrices $(M, C, K)$. The challenge now is to select, if possible, matrices with specified structures out of the solution space. This purpose of this chapter is to employ SDP techniques to accomplish this task [2, 3, 158]. The basic idea is to optimize a suitable linear functional subject to linear equation constraints and some additional condition such as positive semi-definiteness.

In recent years, SDP has emerged as an important tool in mathematical programming for two reasons. The first reason is its versatility to model problems in broad discipline areas ranging from mathematical studies in combinatorial optimization, Boolean and non-convex quadratic programming, min-max eigenvalue problems, and matrix completion problems to engineering applications in nonlinear and time-varying system analysis, controller synthesis, computer-aided control system design, network queueing, optimal statistical model designs,

and structural optimization. Examples of converting these problems into the standard primal problem or its dual can be found in [20, 158, 168]. The second reason is its close comparability to the well known linear programming (LP). Most issues such as the dual theory, interior point algorithms, convergence and polynomial time-complexity for LP can be extended to SDP [1, 134]. This generalization provides efficient procedures for finding the optimal solution based on iterating interior points that either follow the central path or decrease a potential function. Profuse research results are available in the literature. For example, the book on SDP [168] lists 877 references, while the online bibliography collected by Wolkowicz [167] keeps adding new references to its database continually. We find the comprehensive treatise in the two books [3, 20] and two review articles [1, 158] offer quick and useful grasp of this interesting and intensely studied subject.

The rest of this chapter is organized as follows. In Section 4.2 we present a brief overview of semidefinite programming. More advanced treatment can be found in Nesterov and Nemirovskii [134], Todd [158], Boyd and Vandenberghe [20, 160]. Our main thrust in this investigation is to experiment the applicability of the SDP techniques on various structural constraints. Thus, we begin in Section 4.3 with the most basic symmetric and positive semi-definite models and conjecture some new solvable conditions via the testing results. In Section 4.4 we explore inverse eigenvalue problems with a mixture of linear types arising in general gyroscopic systems. We believe that our work in this section alone is innovative as no other QIEP techniques developed thus far can address a mixture of structural constraints simultaneously. In Section 4.5 we discuss how to handle models with inherent sparsity patterns. Different from the numerical approaches applied in Chapter 3, we discuss how to use the SDP approach to solve QIEPs more robustly. We explain how to embed the sparse structures into the optimization problem. This idea is generalized in Section 4.6 to solve models with prescribed entries. In Section 4.7 we present a numerical approach of handling model updating problems with minimal changes while preserving specifically embedded structures, a task which has been proclaimed difficult in the past.

## 4.2 Semidefinite programming

In this section we present a brief overview of semidefinite programming. More advanced treatment can be found in Nesterov and Nemirovskii [134], Todd [158], Boyd and Vandenberghe [20, 160]. Our discussion is based on the review article by Todd [158]. The SDP problem in the

*primal standard form* is formulated as

$$\begin{array}{lll} \text{(P)} & \text{Minimize}_X & C \bullet X \\ & \text{Subject to} & A_i \bullet X = b_i, \quad i = 1, \ldots, m \\ & & X \succeq 0, \end{array} \tag{4.1}$$

where all $A_i \in S\mathbb{R}^{n\times n}$, $C \in S\mathbb{R}^{n\times n}$, $\mathbf{b} \in \mathbb{R}^m$ are given, $X \in S\mathbb{R}^{n\times n}$ is the variable, and $C \bullet X$ denotes the Frobenius inner product between $C$ and $X$ defined by

$$C \bullet X = \text{trace}(X^\top C) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij},$$

with $C = [c_{ij}]$ and $X = [x_{ij}]$. The corresponding *dual standard form* of the SDP problem is expressed as

$$\begin{array}{lll} \text{(D)} & \text{Maximize}_{\mathbf{y}} & \mathbf{b}^\top \mathbf{y} \\ & \text{Subject to} & \sum_{i=1}^{m} y_i A_i + S = C, \\ & & S \succeq 0, \end{array} \tag{4.2}$$

with a "slack" matrix $S \in S\mathbb{R}^{n\times n}$. This dual standard form is also equal to

$$\begin{array}{ll} \text{Maximize}_{\mathbf{y}} & \mathbf{b}^\top \mathbf{y} \\ \text{Subject to} & C - \sum_{i=1}^{m} y_i A_i \succeq 0, \end{array} \tag{4.3}$$

which is most relevant to our QIEPs.

Note that if $X$ is a feasible solution for $(P)$ and $\{\mathbf{y}, S\}$ is a feasible solution for $(D)$, then

$$C \bullet X - \mathbf{b}^\top \mathbf{y} = X \bullet S \geq 0. \tag{4.4}$$

This is the so called *weak duality* property of the SDP problem (see Todd [158] for more discussion in this regard). The difference between the optimal solution of $(P)$ and that of $(D)$ is called the *duality gap* whose "ideal" value would be zero. Nonetheless, we illustrate an example from Vandenberghe and Boyd [160] where there is no duality gap, but the optimal solution for the primal problem could not be reached.

**Example 4.2.1** *Consider the optimization problem in dual standard form:*

$$\begin{array}{ll} \textit{Maximize}_{\mathbf{y}\in\mathbb{R}^2} & y_2 \\ \textit{Subject to} & \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} y_1 + \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} y_2 \preceq \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \end{array} \tag{4.5}$$

*The feasible set can be written as* $\{\mathbf{y} = [y_1, y_2]^\top \in \mathbb{R}^2 : y_1 < 0, y_2 < 0, y_1 y_2 \geq 1\}$. *It is clear*

*that we cannot find a feasible point to achieve the optimal solution* $0 = sup_{\mathbf{y}\in\mathbb{R}^2} \quad y_2$. *We only can approach the optimal value 0 by feasible points of the form* $[1/\alpha, \alpha]^\top$ *with arbitrarily small positive* $\alpha$. *The problem (4.5) is not solvable. On the other hand, its primal problem is the form:*

$$\begin{array}{ll} \textit{Minimize}_{X\in S\mathbb{R}^{2\times 2}} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \bullet X \\ \textit{Subject to} & \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \bullet X = 0, \quad \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \bullet X = 1, \\ & X \succeq 0, \end{array} \tag{4.6}$$

*The only feasible solution for problem (4.6) is* $X = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ *with optimal value* 0. *This concludes that the duality gap is zero, but one of the problems problem is not solvable.*

More examples showing the nonzero duality gap can be found in [126, 158, 160]. Our primary focus is on the case when the duality gap is zero. The proof of the following theorem essentially follows from the *Hahn-Banach separation theorem*, which is fundamental in convex and is referred to [158].

**Theorem 4.2.1** *Assume that there exist a feasible point* $X \in S\mathbb{R}^{n\times n}$ *for (P) and a strictly feasible pair* $\{\mathbf{y}, S\} \in \mathbb{R}^m \times S\mathbb{R}^{n\times n}$, *namely* $S \succ 0$ *for (D). Then the set of the optimal solutions of (P) is nonempty and compact, and the* duality gap *is zero. That is, the optimal values of (P) and (D) are equal.*

We describe a few examples below to demonstrate the flexibility and versatility of the SDP formulation [158].

**Example 4.2.2** *Consider the primal LP problem,*

$$\begin{array}{ll} \textit{Minimize}_{\mathbf{x}} & \mathbf{c}^\top \mathbf{x} \\ \textit{Subject to} & A\mathbf{x} = \mathbf{b}, \\ & \mathbf{x} \geq 0, \end{array} \tag{4.7}$$

*where* $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ *are given,* $\mathbf{x} \in \mathbb{R}^n$, *and its dual*

$$\begin{array}{ll} \textit{Maximize}_{\mathbf{y}} & \mathbf{b}^\top \mathbf{y} \\ \textit{Subject to} & \mathbf{c} - A^\top \mathbf{y} \geq 0, \end{array} \tag{4.8}$$

*with* $\mathbf{y} \in \mathbb{R}^m$. *Defining* $C = Diag(\mathbf{c})$ *and* $A_i = Diag(\mathbf{a_i})$ *with* $A^\top = [\mathbf{a_1}, \mathbf{a_2}, \ldots, \mathbf{a_m}]$, *we*

*immediately see that the dual LP problem is equivalent to*

$$\begin{array}{ll} \textit{Maximize}_{\mathbf{y}} & \mathbf{b}^\top \mathbf{y} \\ \textit{Subject to} & \textit{Diag}(\mathbf{c} - A^\top \mathbf{y}) = C - \sum_{i=1}^m y_i A_i \succeq 0, \end{array} \tag{4.9}$$

*Thus, a dual LP problem is also a dual standard SDP problem. Its primal standard SDP formulation, however, consists of the variable $X \in S\mathbb{R}^{n\times n}$ which is not quite equivalent to the standard LP problem. To clarify this disparity, we claim that this optimal variable $X \in S\mathbb{R}^{n\times n}$ can be made into a diagonal matrix. Observe that, by definition, we can write*

$$\begin{array}{rcccl} C \bullet X & = & c_1 x_{11} + c_2 x_{22} + \ldots + c_n x_{nn} & = & \mathbf{c}^\top \mathbf{x}, \\ A_i \bullet X & = & \mathbf{a}_{i1} x_{11} + \mathbf{a}_{i2} x_{22} + \ldots + \mathbf{a}_{in} x_{nn} & = & b_i, \quad \textit{for each } i, \end{array} \tag{4.10}$$

*with $\mathbf{a}_i = [a_{i1}, \ldots, a_{in}]^\top$, $X = [x_{ij}]$, and $\mathbf{x} = [x_{11}, \ldots, x_{nn}]^\top$. Let*

$$\bar{X} = \textit{Diag}(\mathbf{x}). \tag{4.11}$$

*Then $\bar{X} \succeq 0$ as $X \succeq 0$. Following from (4.10), we know that*

$$\begin{array}{rcccl} C \bullet \bar{X} & = & C \bullet X & = & \mathbf{c}^\top \mathbf{x}, \\ A_i \bullet \bar{X} & = & A_i \bullet X & = & b_i, \quad \textit{for each } i, \end{array} \tag{4.12}$$

*Hence, if $X$ is a feasible solution in (P), then $\bar{X}$ is also a feasible solution in (P) with the same optimal value. Also, $\bar{X} \succeq 0$ is equal to $\mathbf{x} \geq 0$. This concludes that LP problem is a special form of SDP.*

The optimization of eigenvalue-induced functional is a classical problem with applications is areas such as optimal control, graph theory, combinatorial optimization, and so on [89, 131, 136]. The SDP approach makes this challenging task a lot easier than conventional approach [158, 160].

**Example 4.2.3** *Consider the minimization of the maximum eigenvalue of a symmetric matrix $A(\mathbf{z})$ defined by*

$$A(\mathbf{z}) = A_0 + z_1 A_1 + \ldots + z_p A_p, \tag{4.13}$$

*with given and fixed $A_i = A_i^\top \in \mathbb{R}^{n\times n}$. Note that $\zeta \geq \lambda_{max}(A(\mathbf{z}))$ iff $\lambda_{min}(\zeta I_n - A(\mathbf{z})) \geq 0$. Thus, we can formulate this problem as a SDP in the dual standard form*

$$\begin{array}{ll} \textit{Maximize}_y & -\zeta \\ \textit{Subject to} & \zeta I_n - A(\mathbf{z}) \succeq 0, \end{array} \tag{4.14}$$

*where the variable* $y = (\zeta, \mathbf{z})^\top \in \mathbb{R}^{(p+1)\times 1}$.

Unlike other numerical methods, the SDP approach offers a unified, efficient, and tractable scheme for solving QIEPs. In this work, we apply some state-of-the-art SDP software packages [125, 154, 159] successfully to some otherwise very difficult structured QIEPs. We shall demonstrate that structures such as positive definiteness, nonnegativity, mixture type, sparsity patterns, prescribed entries for QIEPs and the associated model updating problems can all be handled effectively by our SDP approach. With this highly efficacious method in hand which provides global and conclusive results, we are at a vantage point to investigate new areas related to QIEPs.

We do not claim that we have developed new algorithms. Rather, this chapter is to explain the usage of SDP techniques to embrace different structures within QIEPs. In this sequel, we concentrate on how to formulate QIEPs under different structures for SDP applications. Once the formulation is set up, it is easy to call up available SDP packages to carry out the computation. Our point is that the SDP approach can undertake the various challenges associated with QIEPs with remarkable uniformity, simplicity and effectiveness. The approach offers uniformity because it can handle almost all structures of practical interest under one framework; simplicity because programming for various QIEPs often involves just a few lines of MATLAB commands through the interface with of our core code; and effectiveness because this approach generally provides make-or-break information on the solvability of a QIEP and computes the optimal solution in a user-specified sense.

## 4.3 Models with symmetric and positive semi-definite structures

By Theorem 2.3.1, we already know that the QIEP with real and symmetric coefficient matrices $M$, $C$ and $K$ is generically solvable if $k < k_{max}$. In real applications, however, one or all coefficient matrices are further required to be positive semi-definite. Semi-definiteness translates into complicated algebraic conditions. For this reason, only a few partial results are available in the literature [24, 38, 112, 117], most of which are characterized by some sufficient conditions imposed upon the prescribed eigeninformation.

Taking advantage of the SDP techniques, the difficult task of constructing any positive semi-definite coefficient matrices could be accomplished from *any* prescribed partial eigeninformation $(\Lambda, X)$. By accomplishment, we mean that either the coefficient matrices are found or that a decision of nonexistence can be made inclusively. The coding, such as the one demonstrated in Table 4.1, often involves only a few lines of YALMIP commands. We remark that YALMIP is a free MATLAB-based toolbox [125] which serves as a convenient interface for multiple external

optimization solvers. Its commands unify and facilitate the different formats in SDP software. While used for solving QIEPs, it significantly simplifies the description of various structural constraints and, within the specified numerical tolerance, offers a reliable and conclusive answer via the well established SDP theory and algorithms.

Table 4.1: An SDP code calling routines from YALMIP to solve QIEPs for positive semi-definite $(M, C, K)$.

```
% Define symmetric variables
sM = sdpvar(n,n);
sC = sdpvar(n,n);
sK = sdpvar(n,n);

% Specify equality and positive semi-definite constraints
F = set(sM*X*Lambda^2 + sC*X*Lambda + sK*X == zeros(n,k));
F = F + set(sM >= 0) + set(sC >= 0) + set(sK >= 0);

% Select SDPT3 as the solver
ops = sdpsettings('solver','sdpt3');

% Invoke SDP solver
solvesdp(F,[],ops);

% Retrieve numerical solution
M = double(sM)
C = double(sC)
K = double(sK)

% Check relative residual
norm(M*X*Lambda^2 + C*X*Lambda + K*X,'fro')/norm([M,C,K],'fro')
```

In Table 4.1, the command `sdpvar(n,n)` defines a symmetric $n \times n$ matrix as a variable. Also, the algebraic condition (3.3) and the positive semi-definitenesss constraints `F` are entered into the code through the command `set`. This macro command `set` can be generalized to handle other types of structural constraints, which will be demonstrated in subsequent discussion. In this code, the package SDPT3 by Tütüncü, Toh, and Todd [159] is designated as the SDP solver for constructing a numerical solution, if exists, for the QIEP with symmetric and positive semi-definite coefficient matrices. This program in Table 4.1 is a working code, although there

are many other options for fine tuning the computation, including modifying the stopping criterion or selecting a different SDP solver. It is important to point out that, except specifying those constrained conditions, no objective function is needed (denoted by `[]` in the command `solvesdp`) to get a solution for these coefficient matrices. That is, for the moment the purpose is only to find *any* feasible solution for this QIEP. No other extra string of conditions is attached.

It is worth noting that the set of feasible solutions, containing at least the trivial solution, is a convex cone. This implies that once a numerical solution $(M, C, K)$ is found (retrievable through the YALMIP command `double`), any positive scalar multiplication is also a solution. Hence, a relative residual through an appropriate normalization serves as a good indicator to recognize whether the QIEP has been solved satisfactorily. Unlike the eigenvector completion procedure we have described in Chapter 3, the unspecified eigenvectors of the reconstructed system are determined inherently and cannot be influenced from outside.

Even at its simplicity, the code in Table 4.1 provides us a fundamental tool to investigate numerically many interesting questions within QIEPs. One such instance that we can explore is to study whether symmetric QIEPs with positive semi-definite coefficient matrices can have $k$ *arbitrarily* prescribed eigenpairs, provided all prescribed eigenvalues necessarily have negative real part. To carry out the experiment, we randomly generate eigenpairs ($X$ and $\Lambda$) such that all prescribed eigenvalues have negative real part, and input the information to the code in Table 4.1. It is of great interest to find that after large number of random tests, we seem to have reached a rather surprising discovery. If $k \leq k_{max} - 4$, regardless of how $k$ is distributed among the numbers $k_c$ of complex-conjugate eigenpairs, $k_p$ of real eigenpairs with positive sign characteristic and $k_n$ of real eigenpairs with negative sign characteristic [70, 117], the feasible set defined by $F$ seems to always contain nontrivial solutions for random $(X, \Lambda)$. It would be fantastic if such an observation could be confirmed by theoretical proof.

Table 4.2 lists some testing results with the case $n = 10$, implying $k_{max} = 16$, and randomly generated $k$ prescribed eigenpairs corresponding to the preselected $(k_c, k_p, k_n)$. The symbols ✓ and ✗ indicate whether a nontrivial solution to the QIEP with positive semi-definite coefficient matrices can be found or not, whereas ❋ indicates that both cases are possible. The value $t$ in the last row is the number of complex conjugate eigenvalues of the reconstructed quadratic model. It must be stressed that these are generic results in the sense that each conclusion is subject to large number of repeated experiments and no exception has been observed. Accordingly, a positive semi-definite solution can always been derived if $k \leq 12$, while $k = 13$ serves as a borderline in which both possibilities can occur.

Table 4.2: Existence of positive definite $M$, $C$ and $K$ to the symmetric QIEP with random eigenstructure.

| $k_c$ | 6 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 4 | 3 | 3 | 4 | 4 | 3 | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_p$ | 0 | 2 | 2 | 1 | 0 | 0 | 1 | 4 | 3 | 4 | 3 | 2 | 2 | 3 | 4 |
| $k_n$ | 0 | 1 | 0 | 2 | 3 | 2 | 1 | 1 | 2 | 3 | 3 | 3 | 2 | 4 | 4 |
| $k$ | 12 | 13 | 12 | 13 | 13 | 12 | 12 | 13 | 13 | 13 | 12 | 13 | 12 | 13 | 12 |
| ☞ | ✓ | ❄ | ✓ | ❄ | ❄ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ❄ | ✓ |
| $t$ | 8 | 7 | | | | | 6 | | | | | 5 | | 4 | 3 |

## 4.4 Models with gyroscopic structures

Consider a dynamical system described by the following quadratic model:

$$\mathfrak{Q}(\lambda) := \mathfrak{Q}(\lambda; M, \mathscr{C}, \mathscr{G}, \mathscr{K}, \mathscr{N}) = \lambda^2 M + \lambda \underbrace{(\mathscr{C} + \mathscr{G})}_{C} + \underbrace{(\mathscr{K} + \mathscr{N})}_{K}, \tag{4.15}$$

where $M$, $\mathscr{C}$, and $\mathscr{K}$, as the usual mass, damping and stiffness matrices, are symmetric and positive semi-definite, but $\mathscr{G}$ and $\mathscr{N}$, representing the gyroscopic and circulatory matrices, are skew-symmetric [172]. The combination such as $C = \mathscr{C} + \mathscr{G}$ or its like is referred to as a mixture of linear types. Note that $C$ is neither symmetric, nor general, but is still of some special structure. This is the so called gyroscopic system which often appears in vibrations of rotating machines or in moving coordinate frames.

Numerical investigation for forward gyroscopic eigensystems have been conducted in [21, 59, 143], but the attention mostly is on the damping free case, i.e., $C = \mathscr{G}$ is skew-symdsmetric. To our knowledge, there is no discussion of the gyroscopic inverse eigenvalue problem except for the work in [4] where a hybrid optimization scheme for identifying inherent parameters of flexible rotor-bearing systems is proposed. Typical decomposition techniques might have trouble in differentiating the positive semi-definite matrix $\mathscr{C}$ from the skew-symmetric matrix $\mathscr{G}$ within the coefficient matrix $C$. In our SDP approach, however, such difficulties could be tackled by simply adding the YALMIP commands

```
% Define skew-symmetric variables

sG = sdpvar{n,n,'skew','real');
sN = sdpvar(n,n,'skew','real');
```

for skew-symmetric matrices and modifying the equality constraint to

```
F = set(sM*X*Lambda^2 + (sC+sG)*X*Lambda + (sK+sN)*X == zeros(n,k));
```

in Table 4.1. The QIEP in the form (4.15) can now be handled without much trouble.

Since each of the skew-symmetric matrices $\mathscr{G}$ and $\mathscr{N}$ adds $\frac{n(n-1)}{2}$ extra variables to the linear system (3.3), it can be proved that for any $k$ the solution to a general (4.15) always exists with respect to any number of prescribed eigenpairs. Curiously, we wonder that if the constraint of positive semi-definiteness for $M$, $\mathscr{C}$ and $\mathscr{K}$ is imposed, how would the inclusion of $\mathscr{G}$ or $\mathscr{N}$ improve the solvability for a QIEP in the gyroscopic form (4.15) from a relative larger number $k$ arbitrarily prescribed eigenpairs? For simplicity, assume $\mathscr{N}$ is a zero matrix and $k_r = k - 2k_c$ stands for the total number of real eigenpairs (since real eigenvalues of a gyroscopic system have no sign characteristic, another significant difference between symmetric and non-symmetric pencils), the numerical evidence summarized in Table 4.3 seems to support affirmatively this added effect of the skew-symmetric matrix $\mathscr{G}$ — the maximal allowable $k$ of arbitrarily prescribed eigenpairs increases by $\lfloor \frac{n-1}{2} \rfloor$. Once again, we stress that such an experiment would have been very difficult by other numerical techniques.

Table 4.3: Existence of positive definite $M$, $\mathscr{C}$, $\mathscr{K}$, and skew-symmetric $\mathscr{G}$ to the QIEP with random eigenstructure.

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $k_c$ | 8 | 7 | 7 | 7 | 6 | 6 | 7 | 7 | 6 | 6 | 6 | 6 | 5 | 5 | 5 | 5 | 3 | 4 | 3 | 0 | 3 |
| $k_r$ | 0 | 1 | 0 | 2 | 3 | 2 | 4 | 3 | 5 | 6 | 5 | 4 | 7 | 6 | 5 | 4 | 6 | 6 | 6 | 6 | 8 |
| $k$ | 16 | 15 | 15 | 16 | 15 | 14 | 18 | 17 | 17 | 18 | 17 | 16 | 17 | 16 | 15 | 14 | 12 | 14 | 12 | 6 | 14 |
| ☞ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| $t$ | 9 | | | 8 | | | 7 | | | | | | 6 | | | | | 5 | | | 4 |

## 4.5 Models with sparsity patterns

Finite element models, just like a mass-spring system or an RLC circuit, often have specific sparsity patterns in $M$, $C$ and $K$. That is, corresponding to inner connectivity, the physical parameters, such as mass, stiffness, voltage, resistance and so on, are embedded in the coefficient matrices $(M, C, K)$ in a fixed but often mixed way. For the sake of physical feasibility, the

recovered parameters must also be nonnegative. In this section, we demonstrate how the SDP approach can meet the connectivity and the nonnegative constraints easily.

We need to point out that our software package Opt4QIEP developed in Chapter 3 includes two features. First, it is a tool which exploits the underlying matrix structure based on specified connectivity. Second, it employs the truncated QR decomposition to create, if possible, a consistent linear inequality system. These features can be combined with the SDP technique to offer a more powerful optimization approach toward the QIEPs.

For demonstration, consider the same mass-spring system in Example 3.2.1. Note that in the system the maintenance of the connectivity and nonnegativity is more important than symmetry and positive semi-definiteness because the preservation of the former automatically implies the latter. Such an endeavor can easily be achieved simply by formulating the QIEP as an SDP problem. Employing Opt4QIEP to create the sparse pattern in the coefficient matrices $M$, $C$, and $K$, we impose the equality and the nonnegative constraints by YALMIP commands as follows.

```
% Define physical parameters

sm = sdpvar(4,1);
sc = sdpvar(3,1);
sk = sdpvar(5,1);

% Define coefficient matrices based on connectivity constraints

sM = diag(sm);
sC  = [sc(1)+sc(2)   0      -sc(2)        0;
            0        0        0           0;
          -sc(2)     0   sc(2)+sc(3)   -sc(3);
            0        0      -sc(3)      sc(3)];
sK  = [sk(1)+sk(2)+sk(5)       -sk(2)        -sk(5)           0;
            -sk(2)          sk(2)+sk(3)      -sk(3)           0;
            -sk(5)             -sk(3)   sk(3)+sk(4)+sk(5)  -sk(4);
              0                  0           -sk(4)         sk(4)];

% Define equality and nonnegativity constraints

F = set(sM*X*Lambda^2 + sC*X*Lambda + sK*X == zeros(n,k));
F = F + set(sm > 0) + set(sc > 0) + set(sk > 0);
```

Note that the intermediate matrix variables `sM`, `sC`, and `sK` are used only for convenience to define the equality constraint (3.3). The true variables, `sm`, `sc`, and `sk`, in the calculation are restricted to nonnegative values elementwise through the `set` command. With care, more programming details would enable us to bypass generating the intermediate matrices explicitly and, thus, we may deal with large scale dynamical systems. Since solutions, if exist, of a QIEP form convex cone, an artificial objective function such as

```
solvesdp(F,(sm(1,1)-1)^2,ops);
```

is recommended to prevent arbitrary scaling by normalizing the first mass $m_1$ to unity. Note that even with the added normalization this is still a convex programming problem. The return from the SDP computation therefore tells either a completely successful reconstruction of positive physical parameters or a utterly disastrous failure giving rise to either the trivial solution or a degenerated system. In our opinion, this is perhaps the most attractive feature in the SDP approach.

Since the mass-spring system in Figure 3.1 has 12 parameters, it would be interesting to ask two further questions. Can the parameters be chosen so that the new system has $k$ arbitrarily prescribed eigenpairs? Even more restrictively but curiously, given an original system with parameters $(\mathbf{m}_0, \mathbf{c}_0, \mathbf{k}_0)$, can the mass parameters $\mathbf{m}$ (or others) alone be altered so that the new system $(\mathbf{m}, \mathbf{c}_0, \mathbf{k}_0)$ has one specific but arbitrarily prescribed eigenpair? The answers to these two problems can easily be investigated numerically by slight modifications of the above SDP code. It turns out that both answers are negative. This is reasonable because an arbitrary assignment is almost doomed due to the fact that the eigenstructure of a structured model should also be structured, even if we do not know of what structure it should be.

## 4.6 Models with prescribed entries

Till now our concern is focused on the construction of the coefficient matrices $(M, C, K)$ as a whole. Even in the preceding section where structure of $(M, C, K)$ is restricted to a certain sparsity pattern , we are still talking about building the entire matrices. In practice, quite often the reconstruction of the model is limited to a local portion of the system, instead of rebuilding the entire system. In other words, in a QIEP it is sometimes needed to modify only a subset of the parameter or a specific part of the coefficient matrices. The former case has already been illustrated by a simple example in the preceding section where only mass parameters alone are subject to adjustment with the (unrealized) hope of updating one or more specific eigenpairs. The latter case could be exemplified by the vibration of bridges, highways, building or automobiles where the spatial representation for substructures such as abutments,

the foundations, or chassis is fixed. In a sense, QIEPs with prescribed entries can be considered as matrix completion problems.

Matrix completion subject to both structural and spectral constraints with prescribed entries is a difficult problem. Constructive proofs for some classical results such as the Schur-Horn theorem [92] and the Hershkowitz theorem [86, 87, 99] are elegant mathematically in their own right, but beyond these, very few theories or numerical algorithms are available. Now, we can use the SDP techniques to analyze and solve QIEPs with prescribed entries without any impediment. In the preceding section we have described a means to impose the sparse pattern by those fixed zeros while defining the `sdpvar` variables (See also the next section). In exactly the same way, we can assign the relevant entries with the prescribed values. After assigning prescribed values to designated positions in the matrix coefficients and calling up some appropriate SDP solvers, we will be given a make-or-break decision on the solvability of the QIEP with prescribed entries. We will further illustrate the application in the next section.

## 4.7 Model updating problems

We have already explored the idea of model updating with no spill-over in Section 2.4. In this section, we study another aspect of model updating with minimum changes. Given original matrices $(M_0, C_0, K_0)$ with specified structures in $\mathbb{R}^{n\times n}$ satisfying

$$\mathfrak{Q}_0(\lambda) := M_0\lambda^2 + C_0\lambda + K_0, \tag{4.16}$$

with coefficients matrices $(M_0, C_0, K_0)$ in some specified structures, the concern in the current model updating is to minimize the difference

$$\|(M_0, C_0, K_0) - (M, C, K)\|_F \tag{4.17}$$

(or a weighted variant) subject to the conditions that the updated coefficient matrices $(M, C, K)$ meet the algebraic constraint (1.2) under $k$ prescribed eigenpairs and maintain the same prescribed structure. Updating under such a context has been widely applied as an important tool for the design, construction and maintenance of mechanical systems [9, 12]. A good discussion about general principles of model updating can be found in the book by Friswell and Mottershead [62]. The basic idea is to refine, correct, or update the current dynamic model (4.16) with minimal changes when natural frequencies and mode shapes do not match well with experimentally measured or desirable frequencies and mode shapes. We could have included the requirement of no spill-over while searching for the minimal changes, but that is going to be a more restrictive process.

For practical consideration, any of the previously discussed structures are possible for

$(M, C, K)$. However, we find that the most commonly discussed cases in the literature are when $M = M_0 \succeq 0$ and $C$ and $K$ are symmetric [6, 61, 112, 115] or when $C = C_0 = 0$ [9, 27, 50, 164, 171]. The absence of discussions for other structures in the literature is not due to their lack of importance. Rather, it is owing to the difficulties associated with the more complicated constraints. Currently, the most predominant updating techniques seem to be the Lagrange multiplier approach adopted in [62], the direct approach via dimension reduction proposed in [112], and the Newton-type iteration used in [6]. They, however, can only handle quadratic model updating on a structure by structure basis. Indeed, it was proclaimed in [62] that "Updating is a process fraught with numerical difficulties."

Amazingly, the SDP techniques with the aid of the YALMIP interface easily give us the ability to hand almost all kinds of structural constraints within the same framework. Suppose there is a sparsity pattern, say, in $K_0$. To maintain the same sparsity in the updated matrix $K$, we could issue commands such as

```
sK = sdpvar(n,n).*abs(sign(K_0));
```

where the element to element multiplication `.*` is to pick up any nonzero locations of $K_0$. Suppose the nearness of the updated model to the original model is measured by the objective function

$$J = \mu\|M_0^{-\frac{1}{2}}(M_0 - M)M_0^{-\frac{1}{2}}\|_F^2 + \nu\|M_0^{-\frac{1}{2}}(C_0 - C)M_0^{-\frac{1}{2}}\|_F^2 + \|M_0^{-\frac{1}{2}}(K_0 - K)M_0^{-\frac{1}{2}}\|_F^2, \quad (4.18)$$

where $\mu$ and $\nu$ are some preselected weight factors [112]. Note that $J$ is a convex but nonlinear function in $(M, C, K)$. Then we can easily transform (4.18) into a second-order Lorentz cone programming problem via the YALMIP commands

```
% Define variables, structured if necessary

sdpvar uM uC uK
sM = ...
   :

W = inv(M_0^(1/2));

% Define equality and structural constraints

F = ...
```

```
% Include rotated Lorentz cones

F = F + set(rcone(reshape(W*(M_0-sM)*W,n^2,1),uM,1/2));
F = F + set(rcone(reshape(W*(C_0-sC)*W,n^2,1),uC,1/2));
F = F + set(rcone(reshape(W*(K_0-sK)*W,n^2,1),uK,1/2));

% Define objective function and call for second order cone programming solvers

sdpsolve(F,mu*uM + nu*uC + uK,ops);
```

where the command `rcone(z,x,y)` with a column vector $\mathbf{z}$ and scalars $x, y > 0$ defines a rotated second order cone constraint $\|\mathbf{z}\|_2^2 < 2xy$. Again, we stress that the structural constraints to be imposed upon the updated model $(M, C, K)$ at the line `F = ...` can be quite general, including all or some of $(M, C, K)$ being positive semi-definite, of sparsity patterns, mixture of linear types, or with fixed entries, as we have discussed earlier. Finally, taking advantage of convex programming, the return from the SDP computation gives us a make-or-break decision about whether the update is achievable and, if it is so, a global optimal solution is obtained.

Consider the updating of the linear model

$$(\lambda M_0 - K_0)\mathbf{x} = 0, \tag{4.19}$$

with some realistic data.

**Example 4.7.1** *We first choose $M_0$ and $K_0$ to be the matrices BCSSTM01 and BCSSTK01, respectively, from the Harwell-Boeing Collection [18], BCSSTRUC1. These are $48 \times 48$ matrices whose sparsity patterns are plotted in Figure 4.1. Specifically, $M_0$, with $\|M_0\|_F \approx 7.7 \times 10^2$, is a nonnegative diagonal matrix with 24 zeros along its diagonal. $K_0$, with $\|K_0\|_F \approx 7.5 \times 10^9$, is symmetric and positive definite. These original data impose two numerical difficulties. First, there are 24 positive eigenvalues with the remaining 24 at positive infinity. Second, the two coefficient matrices are not of comparable scales. Both aspects suggest that eigenvalue computation would be particularly challenging. Indeed, the residuals $\|(\lambda M_0 - K_0)\mathbf{x}\|_2$ of the 24 real-valued eigenvalues and eigenvectors computed by MATLAB ranges from $5.5 \times 10^{-9}$ to $1.6 \times 10^{-6}$, indicating a huge loss of significant digits. Because of this disparity, we have to use our discretion carefully in deciding whether the SDP scheme has returned a satisfactory updating in its computation. We carry out our experiment as follows.*

*Despite the fact that the magnitude of $K_0$ is several order higher that of $M_0$, the SDP calculation can endeavor to overcome the dissimilarity even if we use the non-weighted objective*

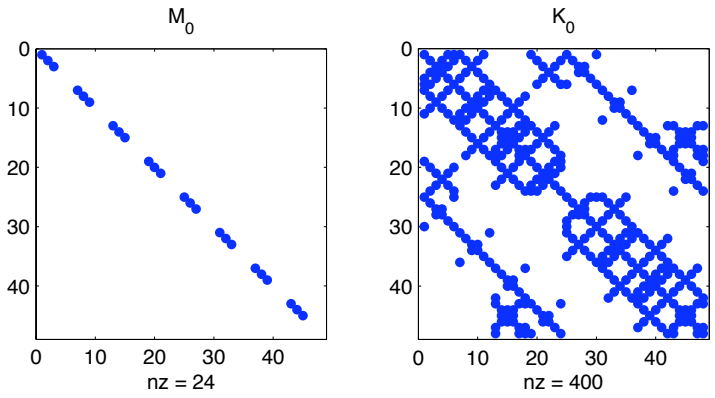


Figure 4.1: The $48 \times 48$ sparse matrices $M_0$ and $K_0$.

*function*

$$J(M, K) = \|M_0 - M\|_F^2 + \|K_0 - K\|_F^2 \tag{4.20}$$

*to measure the nearness of updating. Fix one eigenpair* $(\lambda, \mathbf{x})$ *of the linear pencil (4.19) with* $\|(\lambda M_0 - K_0)\mathbf{x}\|_2 \approx 5.5 \times 10^{-9}$. *Suppose we want to find out the nearest coefficient matrices* $M$ *and* $K$ *satisfying an updated eigenpair* $(\mu, \mathbf{x})$. *Specifically, keep the eigenvector invariant and let the new eigenvalue be a random number generated by*

$$\mu = \lambda(1 + \sigma|\text{randn}(1)|)$$

*with* $\sigma = 10^{-p}$, $p = 1, \cdots, 10$. *Here,* randn(1) *represents a pseudo-random number generated from a normal distribution with mean zero and standard deviation one. For different* $p$ *values, our idea is to let* $\sigma$ *represent the "variance" of the "one-sided" perturbation. Then, we want to check the feasibility of a linear model* $(M, K)$ *containing the eigenpair* $(\mu, \mathbf{x})$ *subject to the same sparse patterns and positive semi-definite properties embedded in* $(M_0, K_0)$.

*The feasibility of the computed results is measured by the residual value* $\|(\mu M - K)\mathbf{x}\|_2$. *Ideally, the residual should be zero. In practice, the floating-point arithmetic can return at most the machine precision if we carefully fine-tune the* `sdpsettings` *in YALMIP. In fact, the singularity and the imbalance in scaling of the original pencil* $(M_0, K_0)$ *make it fairly difficult to*

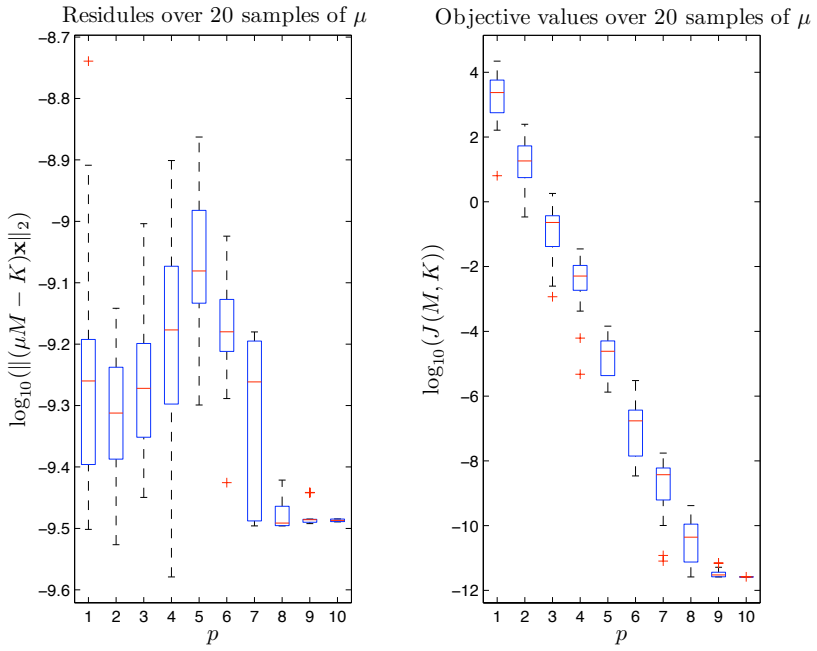


Figure 4.2: Performance of model updating for Harwell-Boeing test data BCSST*01 (n = 48).

*verify feasibility even with exact data. The graph on the left side in Figure 4.2 depictes boxplots of residuals over 20 samples of $\mu$ for each $p$. Assume that the feasibility conditions are satisfied if the residuals are reasonably small. Within this "feasible set", we search for a unique pair $(M, K)$ that is nearest to $(M_0, K_0)$ by the objective function $J(M, K)$. The graph on the right side represents boxplots of the corresponding objective values $J(M, K)$. The objective values deteriorate rapidly as the perturbation of eigenvalues for the structured pencil increases in the sense of its variance. However, the relative discrepancy* $\left(\frac{\|M_0-M\|_F}{\|M_0\|_F}, \frac{\|K_0-K\|_F}{\|K_0\|_F}\right)$ *returned by the SDP approach is surprisingly reasonable with respect to the large magnitude of entries in* $K_0$.

Note that the peculiar characteristics of the BCSST*01 data makes the performance of the SDP procedure dubious. For example, if the SDP solver `SeDuMi` is applied in YALMIP, it returns the warning message "no sensible solution found". To check out whether this confusing message persists, we consider next the BCSST*02 pair which results from applying static condensation to the oil rig model.

**Example 4.7.2** *The two matrices* $(M_0, K_0)$ *in BCSST*02 are of size* $66 \times 66$*, where* $M_0$ *is a diagonal matrix but* $K_0$ *is a dense matrix. Both matrices are symmetric and positive semi-definite matrices and* $M_0$ *is nonsingular.*

*There is still a considerable variation of magnitudes in the entries with* $\|M_0\|_F \approx 8.2 \times 10^{-1}$

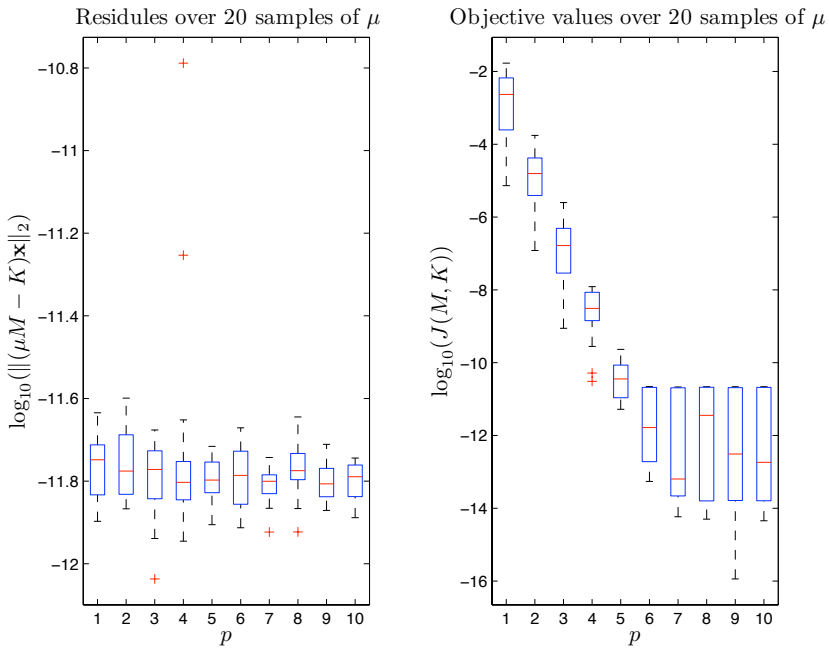


Figure 4.3: Performance of model updating for Harwell-Boeing test data BCSST*02 (n = 66).

*and* $\|K_0\|_F \approx 5.3 \times 10^4$. *But, the eigenvalue computation by MATLAB is much more stable with* $\|(\lambda M_0 - K_0)\mathbf{x}\|_2$ *approximately of the order* $10^{-11}$. *Given one computed eigenpair* $(\lambda, \mathbf{x})$ *with* $\|(\lambda M_0 - K_0)\mathbf{x}\|_2 \approx 2.2 \times 10^{-11}$, *we carry out a similar experiment using the SDP procedure and report the test results in Figure 4.3. In contrast to the BCSST*01 pair, the model updating for BCSST*02 seems more satisfactory.*

## 4.8 Conclusion

The SDP techniques have emerged as a very useful paradigm for solving many convex optimization problems. We have observed how the very challenging solvability issues and complicated computational tasks for various structured QIEPs can be handled using standard software for SDP. Some of cases discussed in this chapter have not even been considered in the literature. What is really remarkable is that using the powerful SDP approach, we are able to solve apparently very difficult QIEPs simply, effectively and efficiently. This SDP approach offers a unified and effectual avenue of attack on the QIEPs in general and the MUPs in particular, which we think deserves attention from practitioners in this field.

The entire procedure can be carried over with little effort to other types of structure such as Toeplitz, Hankel, or even palindrome, and so on. In fact, the idea can even be generalized

to inverse eigenvalue problems for matrix polynomials of arbitrary degrees which, so far as we know, have not been studied at all in the literature but our SDP approach makes a numerical exploration or justification possible.

One possible drawback of the SDP approach is that the interior point methods which are the main engine behind most SDP algorithms cannot handle large scale problems effectively. Todd [158] suggested that "The interior-point methods we have discussed can solve most problems with up to about a thousand linear constraints and matrices of order up to a thousand or so." However, QIEPs under the SDP paradigm are in the dual form. Thus, instead of tracking both dual and primal problems, it is possible to develop a specic technique working on this dual part directly. This might boost the SDP techniques and it is a subject currently under investigation.

# Part II

# Low Rank Factorizations

# Chapter 5

# Nonnegative Matrix Factorization via Polytope Approximation

## 5.1 Overview

The problem of nonnegative matrix factorization (NMF), defined in (1.3), arises in a large variety of disciplines in sciences and engineering. Its wide range of important applications such as text mining, cheminformatics, factor retrieval, image articulation, dimension reduction and so on has attracted considerable research efforts. Many different kinds of NMF techniques have been proposed in the literature, notably the popular Lee and Seung iterative update algorithm [119, 120]. Successful applications to various models with nonnegative data values are abounding. We can hardly be exhaustive by suggesting [56, 91, 95, 105, 108, 121, 124, 137, 139, 142] and the references contained therein as a partial list of interesting work. The review article [152] contains many other more recent applications and references.

The basic idea behind the NMF is the typical linear model,

$$A = UV, \tag{5.1}$$

where $A = [a_{ij}] \in \mathbb{R}^{m\times n}$ denotes the matrix of "observed" data with $a_{ij}$ representing, in a broad sense, the *score* obtained by entity $j$ on variable $i$, $U = [u_{ik}] \in \mathbb{R}^{m\times p}$ is a matrix with $u_{ik}$ representing the *loading* of variable $i$ from factor $k$ or, equivalently, the *influence* of factor $k$ on variable $i$, and $V = [v_{kj}] \in \mathbb{R}^{p\times n}$ with $v_{kj}$ denoting the *score* of factor $k$ by entity $j$ or the *response* of entity $j$ to factor $k$. The particular emphasis in NMF is that all entries of the matrices are required to be nonnegative. To provide a little bit motives on the study of the NMF, we briefly outline two applications below. A good survey of other interesting NMF applications can be found in [152, Section 6.2].

The *receptor model* is an observational technique commonly employed by the air pollution research community which makes use of the ambient data and source profile data to apportion sources or source categories [90, 91, 106, 151]. The fundamental principle in this model is the conservation of masses. Assume that there are $p$ sources which contribute $m$ chemical species to $n$ samples. The *mass balance equation* within this system can be expressed via the relationship,

$$a_{ij} = \sum_{k=1}^{p} u_{ik} v_{kj}, \tag{5.2}$$

where $a_{ij}$ is the elemental concentration of the $i$th chemical measured in the $j$th sample, $u_{ik}$ is the gravimetric concentration of the $i$th chemical in the $k$th source, and $v_{kj}$ is the airborne mass concentration that the $k$th source has contributed to the $j$th sample. In a typical scenario, only values of $a_{ij}$ are observable whereas neither the sources are known nor the compositions of the local particulate emissions are measured. Thus, a critical question is to estimate the number $p$, the compositions $u_{ik}$, and the contributions $v_{kj}$ of the sources. For physical feasibility, the source compositions $u_{ik}$ and the source contributions $v_{kj}$ must all be nonnegative. The identification and apportionment, therefore, becomes a nonnegative matrix factorization problem of $A$.

In another application, the NMF has been suggested as a way to identify and classify intrinsic "parts" that make up the object being imaged by multiple observations. More specifically, each column $\mathbf{a}_j$ of a nonnegative matrix $A$ now represents $m$ pixel values of one image. The columns $\mathbf{u}_k$ of $U$ are basis elements in $\mathbb{R}^m$. The columns of $V$, belonging to $\mathbb{R}^p$, can be thought of as coefficient sequences representing the $n$ images in the basis elements. In other words, the relationship,

$$\mathbf{a}_j = \sum_{k=1}^{p} \mathbf{u}_k v_{kj}, \tag{5.3}$$

can be thought of as that there are *standard parts* $\mathbf{u}_k$ in a variety of positions and that each image $\mathbf{a}_j$ is made by superposing these parts together in some ways. Those parts, being images themselves, are necessarily nonnegative. The superposition coefficients, each part being present or absent, are also necessarily nonnegative. In either case above and in many other contexts of applications, we see that the $p$ factors, interpreted as either the sources or the basis elements, play a vital role. In practice, there is a need to determine as fewer factors as possible and, hence, a low rank nonnegative matrix approximation of the data matrix $A$ arises. The mathematical formulation of an NMF has been defined earlier in (1.3).

It has been argued that the NMF can be interpreted geometrically as the problem of finding a simplicial cone which contains a cloud of data points located in the first orthant [56]. Further extending that thought, we recast in this investigation the NMF as the problem of approximating a polytope on the probability simplex by another polytope which has fewer facets. Our

basic idea follows from computational geometry where a complex shape is to be "silhouetted" by a simpler one. In our particular setting, the complex shape refers to the boundary of a convex polytope of $n$ points in $\mathbb{R}^m$ whereas the simpler one refers to that of $p$ points in the same space. A unique characteristic in our approach is that the approximation is to take place on the probability simplex. We shall exploit the fact that the probability simplex, being a compact set with well distinguishable boundary, makes the optimization procedure easier to manage. For convenience, we denote henceforth a nonnegative matrix $U$ by the notation $U \geq 0$.

This chapter actually deals with two related but independent problems. The first problem considers the polytope approximation only on the probability simplex. The underlying geometry is easy to understand, but the problem is of interest in itself. More importantly, with slight modifications the idea lends it geometric characteristics naturally to the more difficult NMF problem. For both problems a common feature in our approach is alternating optimization. That is, by rewriting the equivalent formulations (at the global minimization point):

$$\min_{U,V\geq 0} \|A - UV\|_F^2 = \min_{U\geq 0} \left( \min_{V\geq 0} \|A - UV\|_F^2 \right) \tag{5.4}$$

$$= \min_{V\geq 0} \left( \min_{U\geq 0} \|A - UV\|_F^2 \right), \tag{5.5}$$

we solve $\min_{V\geq 0} \|A - U_0 V\|_F^2$ from an initial matrix $U_0$ to find the solution matrix $V_0$, and then solve $\min_{U\geq 0} \|A - UV_0\|_F^2$ to find the next iterate $U_1$. In general, $V_\ell$ is obtained from $U_\ell$ and $U_{\ell+1}$ is obtained from $V_\ell$. Our main contribution is to explain how the unique global minimizers to the two subproblems (5.4) and (5.5) can be computed alternatively.

We organize our presentation as follows. Beginning in section 5.2, we briefly describe how the original NMF can be formulated as a low dimensional polytope approximation through scaling. When limited to the probability simplex, we introduce in section 5.3 two basic mechanisms for polytope approximation — a recursive algorithm which projects a given point onto a prescribed polytope and a descent method which slides points along the boundary of the probability simplex to improve the objective value. We emphasize that the recursive algorithm determines the unique and global proximity map in finitely many steps and thus enables our NMF method, to be discussed in section 5.4, to obtain much better, generally of several order, improvement over the well know Lee-Seung updating algorithm [120] per iterative step.

## 5.2 Pull-back to the probability simplex

In this section, we develop a simple technique, based on the concept of "normalization", for obtaining an geometric representation of $A$ such that its NMF can be described as a low dimensional polytope approximation.

Given a nonnegative matrix $A = [\mathbf{a}_1, \ldots, \mathbf{a}_n] \in \mathbb{R}_+^{m\times n}$ with nonzero columns $\mathbf{a}_k \in \mathbb{R}^m$, the associated *pull-back map* $\vartheta(A)$ is defined by

$$\vartheta(A) := A\sigma(A)^{-1}. \tag{5.6}$$

where $\sigma(A)$ is the *scaling factor* given by

$$\sigma(A) := \operatorname{diag}\{\|\mathbf{a}_1\|_1, \ldots, \|\mathbf{a}_n\|_1\} \tag{5.7}$$

with $\|\cdot\|_1$ representing the 1-norm of a vector. Note that each column of $\vartheta(A)$ can be regarded as a point on the $(m-1)$-dimensional *probability simplex* $\mathcal{D}_m$ defined by

$$\mathcal{D}_m := \left\{\mathbf{a} \in \mathbb{R}_+^m | \mathbf{1}_m^\top \mathbf{a} = 1\right\}, \tag{5.8}$$

where $\mathbf{1}_m = [1, \ldots, 1]^\top$ stands for the vector of all 1's in $\mathbb{R}^m$. Clearly, all columns of $\vartheta(A)$ must be contained in its convex hull $\mathcal{C}(A)$ on $\mathcal{D}_m$. Actually, the vertices of $\mathcal{C}(A)$ are composed of a subset of columns of $\vartheta(A)$. We denote this fact by a submatrix $\widetilde{A} = [\mathbf{a}_{i_1}, \ldots, \mathbf{a}_{i_p}] \in \mathbb{R}^{m\times p}$ of $A$ such that

$$\mathcal{C}(A) := \operatorname{conv}(\vartheta(A)) = \operatorname{conv}(\vartheta(\widetilde{A})).$$

Then every column of $\vartheta(A)$ could be expressed in a convex combination of columns of $\vartheta(\widetilde{A})$. This fact can be denoted by the equation

$$\vartheta(A) = \vartheta(\widetilde{A})Q, \tag{5.9}$$

where $Q \in \mathbb{R}^{p\times n}$ itself represents $p$ points in the simplex $\mathcal{D}_p$. Together, we have obtained

$$A = \vartheta(A)\sigma(A) = \vartheta(\widetilde{A})(Q\sigma(A)) \tag{5.10}$$

which is an exact nonnegative matrix factorization of $A$.

It is worth noting that the integer $p$ in this setting represents the number of vertices of the convex hull $\operatorname{conv}(\vartheta(A))$. All we know is that $p \leq n$, but generally $p \geq m$. See Figure 5.1. In practice, we prefer to see that $\vartheta(A)$ is contained in convex hull with fewer than $\min\{m, n\}$ vertices, but clearly this is not always possible and thus the factorization has to be replaced by the notion of approximation.

Suppose that a given nonnegative matrix $A \in \mathbb{R}_+^{m\times n}$ can indeed be factorized as the product of two nonnegative matrices, $A = UV$, with $U \in \mathbb{R}_+^{m\times p}$ and $V_+ \in \mathbb{R}^{p\times m}$. Then

$$A \;=\; \vartheta(A)\sigma(A) = UV = \vartheta(U)\vartheta(\sigma(U)V)\sigma(\sigma(U)V). \tag{5.11}$$

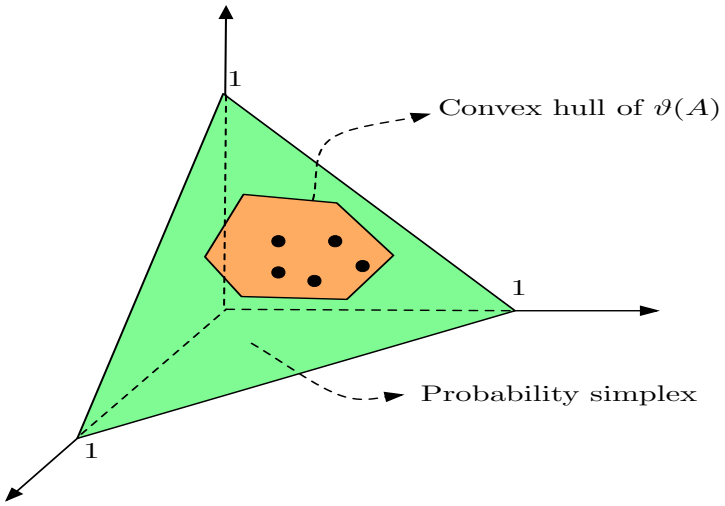


Figure 5.1: Convex hull of $\vartheta(A) \in \mathbb{R}^{3\times n}$ with $n = 11$.

where the product $\vartheta(U)\vartheta(\sigma(U)V)$ itself is on the simplex $\mathcal{D}_m$. It follows that

$$\vartheta(A) = \vartheta(U)\vartheta(\sigma(U)V), \tag{5.12}$$

$$\sigma(A) = \sigma(\sigma(U)V). \tag{5.13}$$

Since $UV = (UD)(D^{-1}V)$ for any invertible matrix $D \in \mathbb{R}^{p\times p}$, we may assume without loss of generality that $U = \vartheta(U)$ with $\sigma(U) = I_n$. In the context that $\vartheta(A) = \vartheta(U)\vartheta(V)$ and $\sigma(A) = \sigma(V)$, we may say that $\vartheta$ is a homomorphism and that $\sigma$ is length preserving.

So far we are interested in the matrix $A$ being factorized exactly into two nonnegative matrices $U$ and $V$. In order to ensure such a factorization, it require that the integer $p$ be large enough such that all columns of $\vartheta(A)$ are included in a convex hull generated by $p$ vertices in $\mathcal{D}_m$. A requirement of large $p$ is not of practical value in applications. But, the above analysis seems to shed some light on the geometric meaning of NMF.

If $p$ is not large enough, the equality most likely cannot hold. The next reasonable expection is to discover two matrices $U \in \mathbb{R}_+^{m\times p}$ and $V \in \mathbb{R}_+^{p\times n}$ minimizing the distance between $A$ and $UV$. In other words, we want to optimize the objective functional

$$f(U,V) = \frac{1}{2}\|A - UV\|_F^2 = \frac{1}{2}\| \left( \vartheta(A) - U \underbrace{V\sigma(A)^{-1}}_{W} \right) \sigma(A)\|_F^2, \tag{5.14}$$

with the assumption that the matrix $U$ is already normalized to $\mathcal{D}_m$. For convenience, denote

$$W = V\sigma(A)^{-1}. \tag{5.15}$$

Each column of the product can be interpreted as points in the *simplicial cone* of $U$. Thus the NMF is equal to find points $\{\mathbf{u}_1, \ldots, \mathbf{u}_p\}$ on the simplex $\mathcal{D}_m$ so that the distance from $\vartheta(A)$ to the simplicial cone spanned by $\{\mathbf{u}_1, \ldots, \mathbf{u}_p\}$, measured by a weighted norm, is minimized. The main trust of the remaining discussion is to explain the minimization process. Based on the the well known Hahn-Banach theorem, the optimization can be accomplished by simply utilizing the notion of supporting hyperplanes of the convex set.

Note that the scaling factor $\sigma(A)$ depends on each given $A$. The distance between $A$ and $UV$ measured in the Frobenius norm therefore can be viewed as the distance between $\vartheta(A)$ and $UW$ measured with the induced norm from the weighted inner product.

## 5.3 Polytope fitting problem

Before treating the general NMF problem, we consider in this section a special set estimation problem which is worthy of discussion in its own right — Use a simpler polytope on the simplex $\mathcal{D}_m$ to best approximate a more complicated polytope $\mathrm{conv}(\vartheta(A))$ such that the total distance from vertices of $\vartheta(A)$ to the polylope $\mathrm{conv}(U)$ is minimal. This consideration is mathematically equivalent to

$$\frac{1}{2}\|\vartheta(A) - UW\|_F^2, \tag{5.16}$$

subject to the constraints that columns of $U$ are in $\mathcal{D}_m$ and columns of $W$ are in $\mathcal{D}_p$.

This characteristic of this problem is similar in spirit to the well known sphere packing problem [46] as well as the classical problem of approximating convex bodies by polytopes [28, 42, 77, 78]. The underlying principle is also analogous with that of the $k$-plane method [128] or the support vector machines method [11]. Above and beyond, the geometric object used in our optimization, which is either a subspace or a polytope, is with high co-dimension and therefore harder to peculiarize. For applications, the set estimation is also essential to pattern analysis [83], robotic vision and tomography, although most of the applications investigated in pattern recognition are restricted to only 3-D objects while we are interested in higher dimensional entities.

This optimization problem described in (5.16) possess a profound geometric meaning in itself. Since $\mathrm{conv}(U)$ has fewer vertices than $\mathrm{conv}(\vartheta(A))$, the minimization of (5.16) conveys a sense of retrieving and regulating the "shape" of $\vartheta(A)$ via fewer facets. Note that columns of $W$ represents coefficients in this convex combination. It is necessary that the *diameter* of

conv($U$) should be at least as large as that of conv($\vartheta(A)$). That is, if the data $\vartheta(A)$ starts with an elongated distribution over $\mathcal{D}_m$, the convex hull at the optimal solution should preserve a similar shape. Unless the vertices of conv($\vartheta(A)$) are sitting on the boundary of $\mathcal{D}_m$, conv($U$) can always be "extended" outward a little bit toward the boundary of $\mathcal{D}_m$ without affecting the product of $UW$. The means that the solution $U$ clearly need not to be unique.

We demonstrate the idea of annexation for the solution $U$ in Figure 5.2 for the case $m = 3$ and $p = 2$. Note that by expanding the segment representing conv($U$) to the two sides of the

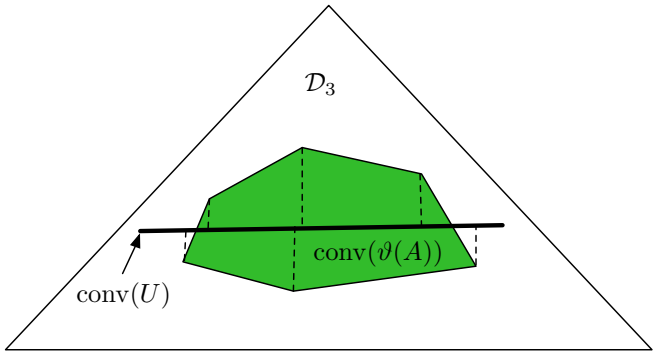


Figure 5.2: Convex hull of $\vartheta(Y)$ and $U$ in $\mathcal{D}_3$.

triangle, it consistently gives rise to the same nearest points to $\vartheta(Y)$. For this reason, the columns of $U$ can be assumed to reside on the boundary of the simplex $\mathcal{D}_m$. We can easily characterize a point on the boundary of $\mathcal{D}_m$ by the location of zero(s) in its coordinates. For example, each facet has one zero. The $i$th column in the following matrix, divided by the scalar $m - 1$,

$$\frac{1}{m-1}\begin{bmatrix} 0 & 1 & 1 & \dots & & 1 \\ 1 & 0 & 1 & \dots & & 1 \\ 1 & 1 & 0 & & & 1 \\ \vdots & & & \ddots & & \\ 1 & & & & & 1 \\ 1 & 1 & & & 1 & 0 \end{bmatrix}$$

represents the "midpoint" of the $i$th facet of the simplex $\mathcal{D}_m$. When there is more than one zero in the entries, we consider this point to be on a "ridge" and so on.

Combining the above thoughts together, the polytope fitting problem can be formulated as

a constrained optimization problem,

$$\begin{aligned} \text{minimize} \qquad & g(U,W) = \frac{1}{2}\|\vartheta(A) - UW\|_F^2, \\ \text{subject to} \qquad & \text{columns of } U \in \partial\mathcal{D}_m, \quad W \succeq 0, \quad \mathbf{1}_p^\top W = \mathbf{1}_n^\top, \end{aligned} \tag{5.17}$$

where $\partial\mathcal{D}_m$ stands for the boundary of $\mathcal{D}_m$. During the computation, it matters to track which boundary is being involved in $U$.

In the next two subsections, alternating direction iteration (ADI) will be used for solving (5.17). With slight modification, the idea will later be extended to the NMF problem.

### 5.3.1 Proximity map on the probability simplex

Let $U \in \mathbb{R}^{m\times p}$ be a fixed matrix with its columns in $\partial\mathcal{D}_m$. Temporarily ignoring the constraint $W \succeq 0$, define the *Lagrangian*

$$L(U,W;\boldsymbol{\mu}) := \frac{1}{2}\|\vartheta(A) - UW\|_F^2 + (\mathbf{1}_p^\top W - \mathbf{1}_n^\top)\boldsymbol{\mu}. \tag{5.18}$$

associated with the problem (5.17) with a vector $\boldsymbol{\mu} \in \mathbb{R}^{n\times 1}$. A necessary condition for the optimal solution $W$ is that partial gradient of (5.18) with respect to $W$ must be zero, that is,

$$\frac{\partial L(U,W;\boldsymbol{\mu})}{\partial W} = -U^\top(\vartheta(Y) - UW) + \mathbf{1}_p\boldsymbol{\mu}^\top = 0, \tag{5.19}$$

Thus we have

$$W := (U^\top U)^{-1}(U^\top\vartheta(Y) - \mathbf{1}_p\boldsymbol{\mu}^\top), \tag{5.20}$$

and

$$\boldsymbol{\mu}^\top = \frac{\mathbf{1}_p^\top(U^\top U)^{-1}U^\top\vartheta(Y) - \mathbf{1}_n^\top}{\mathbf{1}_p^\top(U^\top U)^{-1}\mathbf{1}_p}. \tag{5.21}$$

The trouble is that the formula (5.20) cannot guarantee $W \succeq 0$. As an alternative approach, we suggest employing the notion of proximity map based on the classical Hahn-Banach theorem to compute $W$.

The Hahn-Banach theorem is one of the most important results in functional analysis. It asserts that two disjoint convex sets in a topological vector space can be separated by a continuous linear map [20]. The following equivalent statement serves as the basis of our theory.

**Theorem 5.3.1** *Let $C$ be a closed convex set in $\mathbb{R}^m$. Then for each given point $\mathbf{x} \in \mathbb{R}^m$, there is a unique point $\rho(\mathbf{x}) \in C$ that is nearest to $\mathbf{x}$.*

The nearest point $\rho(\mathbf{x})$ is called the proximity map of $\mathbf{x}$ with respect to $C$. In a Euclidean

**Algorithm 2**: Sekitani and Yamamato Algorithm $[\hat{\mathbf{x}}] = \mathscr{N}(U)$

**Input**: Matrix $U \in \mathbb{R}^{m \times p}$ (A set of $p$ points in $\mathbb{R}^m$)

**Output**: $\hat{\mathbf{x}} = \rho(0)$ with respect to $\mathrm{conv}(U)$

`(For a given point set` $S$`, the symbol` $\mathscr{N}(S)$ `denotes the point on` $\mathrm{conv}(S)$ `which has the minimum Euclidean norm.)`

2.1 **begin**

2.2 $k \leftarrow 0$; $\mathbf{x}_0 \leftarrow$ an arbitrary point from $\mathrm{conv}(U)$;
`(find supporting hyperplane)`

2.3 $\alpha_k \leftarrow \min\left\{\mathbf{x}_{k-1}^\top \mathbf{p} | \mathbf{p} \in U\right\}$;

2.4 **if** $\|\mathbf{x}_{k-1}\|^2 \leq \alpha_k$ **then**

2.5 $\widehat{\mathbf{x}} \leftarrow \mathbf{x}_{k-1}$; stop;

2.6 **end**
`(recursion)`

2.7 $P_k \leftarrow \left\{\mathbf{p} | \mathbf{p} \in U \text{ and } \mathbf{x}_{k-1}^\top \mathbf{p} = \alpha_k\right\}$;

2.8 $\mathbf{y}_k \leftarrow \mathscr{N}(P_k)$;
`(check separation)`

2.9 $\beta_k \leftarrow \min\left\{\mathbf{y}_k^\top \mathbf{p} | \mathbf{p} \in U - P_k\right\}$;

2.10 **if** $\|y_k\|^2 \leq \beta_k$ **then**

2.11 $\widehat{\mathbf{x}} \leftarrow \mathbf{y}_k$; stop;

2.12 **end**
`(rotation and updating)`

2.13 $\lambda_k \leftarrow \max\left\{\lambda | \left((1-\lambda)\mathbf{x}_{k-1} + \lambda \mathbf{y}_k\right)^\top \mathbf{y}_k \leq \left((1-\lambda)\mathbf{x}_{k-1} + \lambda \mathbf{y}_k\right)^\top \mathbf{p}, \mathbf{p} \in U - P_k\right\}$;

2.14 $\mathbf{x}_k \leftarrow (1-\lambda_k)\mathbf{x}_{k-1} + \lambda_k \mathbf{y}_k$;

2.15 $k \leftarrow k+1$;

2.16 Go to line 2.3;

2.17 **end**

space, it is clear that the necessary and sufficient condition for qaulifying $\rho(\mathbf{x})$ is that the inequality

$$(\mathbf{x} - \rho(\mathbf{x}))^\top (\mathbf{z} - \rho(\mathbf{x})) \leq 0 \tag{5.22}$$

holds for all $\mathbf{z} \in C$. In particular, $\|\rho(0)\|^2 \leq \rho(0)^\top \mathbf{z}$ for all $\mathbf{z} \in C$.

Corresponding to each column of $\vartheta(A)$, our goal is to find out its unique nearest point (proximity map) in $\mathrm{conv}(U)$. Since this nearest point resides in $\mathrm{conv}(U)$, it is a convex combination of columns of $U$. It follows that $W \succeq 0$ automatically. Toward this goal, the remaining task is to develop an algorithm that is able to compute proximity map for each column of $\vartheta(A)$ with respect to $U$. The recursive algorithm proposed by Sekitani and Yamamato [148] seems to be a reasonable means for computing the proximity map when $p$ is small. To carry out the

computation, recall that a *hyperplane* is a set of the form

$$H(\mathbf{n}, c) := \{\mathbf{x} | \mathbf{n}^\top \mathbf{x} = c\}, \tag{5.23}$$

where the vector $\mathbf{n} \neq 0$ and the scalar $c$ are some given quantities. Each hyperplane divides $\mathbb{R}^m$ into two *half-space*, one of which defined by

$$H^+(\mathbf{n}, c) := \{\mathbf{x} | \mathbf{n}^\top \mathbf{x} \geq c\}$$

is of interest to us. Let $C$ be a given convex set and the origin $0 \notin C$. Then, the hyperplance $H(\rho(0), \|\rho(0)\|^2)$ is called a *supporting hyperplane* to $C$ at the point $\rho(0)$, provided

$$C \subset H^+(\rho(0), \|\rho(0)\|^2).$$

The main idea in the Sekitani and Yamamato algorithm, outlined in Algorithm 2, is built upon repeatedly searching for the best supporting hyperplane by discarding unneeded facets of the underlying polytope. The supporting hyperplanes are constructed though Steps 2.3 to 2.5 in Algorithm 2; unneeded facets are disposed of at Step 2.7; Steps 2.13 to 2.14 aim at improving the supporting hyperplanes. The entire process is repeated recursively until the nearest point is found.

It is worthwhile to remark three important features of the routine Sekitani and Yamamato algorithm. First, it does not need to solve systems of linear equations since it is not based on simplicial decomposition as in the classical method [166]. Second, it can begin with an arbitrary point in $\text{conv}(U)$. That is, it does not require any initial separating hyperplane. Third, it terminates in finite steps. More details about its mathematical theory are referred to [148].

With the aid of programming languages, such as MATLAB, that allow a subprogram to invoke itself recursively, we implement Algorithm 2 in the following code for our applications.

---

```
function [y,c] = sekitani(U,active);
%
% The code SEKITANI computes the nearest point on the polytope generated by
% the columns of U to the origin.
%
% Reference:
%
%   Sekitani and Yamamoto Algorithm, Math. Programming, 61(1993), 233-249.
```

```
%
% Input:
%
%   U = vertices of the given polytope
%   active = set of active column indices (of U) for recursive purpose.
%
% Output:
%
%   y = the point on conv(U) with minimal norm
%   c = convex coordinates of y with respect to U, that is, y = Uc
%

[m,p0] = size(U); p = length(active);
eps = 1.e-13;                                   % relaxed machine zero
looping = 'y';

temp0 = norm(U(:,active(1))); S = 1; for j = 2:p
    temp1 = norm(U(:,active(j)));
    if temp1 < temp0, temp0 = temp1; S = j; end
end
x = U(:,active(S));                             % starting point
xc = zeros(p0,1);
xc(active(S)) = 1;                              % initial coordinates of x

while looping == 'y';
    temp0 = x'*U(:,active);
    alpha = min(temp0);
    K0 = find(temp0<=alpha+eps);                % find supporting hyperplanes
    if norm(x)^2 <= alpha+eps
        y = x;
        c = xc;
        looping = 'n';
        return
    else
        Pk = active(K0);
        if length(K0) == 1,
            y = U(:,Pk);
```

```
            c = zeros(p0,1);
            c(Pk) = 1;
        else
            [y,c] = sekitani(U,Pk);             % recursion
        end
    end

    I_K0 = setdiff(1:p,K0);
    temp1 = y'*U(:,active(I_K0));
    beta = min(temp1);
    if norm(y)^2 <= beta+eps
        looping = 'n';
        return
    else
        temp3 = U(:,active(I_K0))-y*ones(1,length(I_K0));
        temp5 = find(((y-x)'*temp3)<0);
        temp4 = (x'*temp3(:,temp5))./((y-x)'*temp3(:,temp5));
        lambda = -max(temp4);                   % maximal rotation parameter
        delta = lambda*(y-x);
        if norm(delta) <= max([m,p])*eps
            looping = 'n';
            return
        else
            x = x + lambda*(y-x);
            xc = xc + lambda*(c-xc);            % updated coordinates of x
            looping = 'y';
        end
    end
end
```

---

The procedure described in [148] computes only the nearest point $\hat{\mathbf{x}}$ on the polytope $\text{conv}(U)$. In our applications, we need to know the convex combination coefficients of $\hat{\mathbf{x}}$ corresponding to $U$. It turns out that the coefficient $\mathbf{c}$ such that $\hat{\mathbf{x}} = U\mathbf{c}$ with $\sum_{i=1}^{p} c_i = 1$ and $c_i \geq 0$ could be collected with only a few vector operations in our code.

The Sekitani and Yamamato algorithm is meant to find the proximity map $\rho(0)$ on $\text{conv}(U)$ that is nearest to the origin. To generalize the proximity map $\rho(\mathbf{x})$ with respect to a general point $\mathbf{x}$, we only need to translate the origin to $\mathbf{x}$. Algorithm 3 illustrates the procedure of how

**Algorithm 3**: Polytope fitting problem on the probability simplex $[W] = \text{SIMPLEX}(U)$

**Input**: $Z = \vartheta(A) \in \mathbb{R}^{m\times n}$ and $U \in \partial\mathcal{D}_m$

**Output**: $W =$ the convex combination coefficients for the proximity map of $Z$ onto $\text{conv}(U)$

3.1 **begin**

3.2 **for** $i \leftarrow 1$ **to** $n$ **do**

3.3 $U0 \leftarrow U - Z(:, i)\mathbf{1}_p^\top$;

3.4 $active \leftarrow [12\cdots p]$;

3.5 $[s, c] \leftarrow sekitani(U0, active)$;

3.6 $W(:, i) \leftarrow c$;

3.7 **end**

3.8 **end**

to find the optimal solution of Problem (5.17) for each given $U$. We quickly point out that the computation for each $Z(i, :)$ between Steps 3.2 and 3.7 in Algorithm 3 is independent of each other. This segment is embarrassingly parallelizable and, thus, can be effectively executed on a parallel machine, if so desired.

### 5.3.2 Alternating direction iteration

Once the optimal $W$ is found for a given $U$, the next step is to redefine the convex $\text{conv}(U)$ to better fit $\text{cov}(\vartheta(Y))$. We need to update columns of $U$ along the boundary $\partial\mathcal{D}_m$. We choose to carry out this task by the projected gradient method with line search. It is easy to see that the gradient of $g$ with respect to $U$ at a feasible point $(U, W)$ is given by the $m \times p$ matrix

$$\nabla_U g(U, W) := \frac{\partial g}{\partial U} = -(\vartheta(Y) - UW)W^\top. \tag{5.24}$$

For $i = 1, \ldots p$, the $i$th column of $\nabla_U g(U, W)$ represents precisely the gradient $\nabla_{\mathbf{u}_i} g(U, W)$ of $g$ with respect to the column $\mathbf{u}_i$. The trouble is that this gradient often points away from the simplex. To remedy this, we have to project the gradient back to $\partial\mathcal{D}_m$. As a result, the movement of $\mathbf{u}_i$ in the negative direction of its projected gradient not only reduces the objective value of $g$ but also stays in $\partial\mathcal{D}_m$. Note that the projection of the gradient has to be computed with respect to different column vectors of $U$ step by step. Since the probability simplex $\mathcal{D}_m$ consists of $m$ facets, we can identify the facet by the location of its coordinate. In application, it is necessary to know which facet the current gradient $\nabla_{\mathbf{u}_i} g(U, W)$ is related to. Usually, we improve $\mathbf{u}_i$ within the same facet where $\mathbf{u}_i$ resides. While $\mathbf{u}_i$ happens to be at the intersection of two facets, a decision should be made to move the current $\mathbf{u}_i$ to a suitable facet for improve the minimization. The tactics for controlling the facet crossing will be investigated in the sequel.

Suppose that $\mathbf{u}_i$ resides on the $j$th facet of $\mathcal{D}_m$. The $j$th facet of $\mathcal{D}_m$ can be regarded as

the intersection of two hyperplanes $y_1 + \ldots + y_m = 1$ and $y_j = 0$ in $\mathbb{R}^m$. If we define the $m \times 2$ matrix

$$A_j := [\mathbf{1}_m, \mathbf{e}_j], \tag{5.25}$$

where $\mathbf{e}_j$ is the $j$th standard unit vector in $\mathbb{R}^m$, then the projection of $\nabla_{\mathbf{u}_i} g(U, W)$ onto the $j$th facet is given by

$$\nabla^j_{\mathbf{u}_i} g(U, W) := (I_m - A_j(A_j^\top A_j)^{-1} A_j^\top) \nabla_{\mathbf{u}_i} g(U, W). \tag{5.26}$$

Interestingly, without a difficult calculation, the projection matrix $A_j(A_j^\top A_j)^{-1} A_j^\top$ is attainable through the formula

$$A_j(A_j^\top A_j)^{-1} A_j^\top = \frac{1}{m-1} \begin{bmatrix} 1 & \ldots & 1 & 0 & 1 & \ldots & 1 \\ \vdots & \ddots & & \vdots & & & \\ 1 & & 1 & 0 & 1 & & \\ 0 & & 0 & m-1 & 0 & \ldots & 0 \\ 1 & & 1 & 0 & 1 & & 1 \\ \vdots & & & & & & \vdots \\ 1 & \ldots & 1 & 0 & 1 & \ldots & 1 \end{bmatrix}, \tag{5.27}$$

where the value $m - 1$ occurs at the $(j, j)$ position of the matrix. This formula provides an efficient and direct way for computing the projected gradient.

After getting the projection $\nabla^j_{\mathbf{u}_i} g(U, W)$, $i = 1, \ldots, p$, line search techniques are used to adjust $\mathbf{u}_i$ for reducing the object value. While applying the line search, any entry of the newly computed matrix must remain positive. Therefore, we must be careful to fine-tune the step size to avoid overshooting beyond the current facet. The optimal step size becomes the longest step which can reduce the object value most while keeping all entries of the updated matrix nonnegative. Whenever an entry is crossing zero, it indicates that the search path is now reaching a "ridge" and a change of the facet is to be discerned. The procedure about updating $U$ is summarized in Algorithm 4 below.

At Step 4.6 in Algorithm 4, finding the optimal $\alpha$ to minimize the objective value becomes an important issue in keeping the updated matrix $U$ nonnegative. We prefer a value of $\alpha$ that decreases $h$ rapidly, but we also need to reduce the size of $\alpha$ when its $\alpha$ value gives rise to negative entries in matrix $U$. When there are more than one zero in the current column of $U$, it means that the searching path is now reaching a "ridge" and the new appearing zero suggests the direction of changing the "facet".

**Algorithm 4**: IMPROVING $U$ $[U] = \text{IMPROVE}(U)$

**Input**: $\vartheta(A) \in \mathbb{R}_+^{m\times n}$, $U \in \partial\mathcal{D}_m$, and $W \in \mathbb{R}_+^{p\times n}$
**Output**: Updated matrix $U \in \partial\mathcal{D}_m$

4.1 **begin**
4.2 Use formula (5.24) to calculate the gradient of $g$ with respect to $U$;
4.3 **for** $i \leftarrow 1$ **to** $p$ **do**
4.4 Use formula (5.26) to compute the projection of $\nabla_{\mathbf{u}_i} g(U, W)$ onto the $j$th facet;
4.5 **end**
4.6 Choose $\alpha$ to minimize $h(\alpha) = g(U + \alpha\nabla_{\mathbf{u}} g(U, W), W)$;
4.7 Update $U$ while keeping each entry of $U$ nonnegative;
4.8 **end**

Given $Z = \vartheta(A) \in \mathbb{R}^{m\times n}$, $U \in \partial\mathcal{D}_m$ and $W \in \mathbb{R}^{p\times n}$, the following MATLAB code details the procedure of adjusting $U$ along $\partial\mathcal{D}_m$ while minimizing the objective $g(U, W)$.

```
function [U,facet] = update(Z,U,W,facet);
%
% Input:
%
%   Z = points to be aprroximated
%   U = vertices of the current polytope
%   W = convex combination coefficients
%   facet = indices of facets where U is residing
%
% Output:
%
%   U = vertices of new polytope
%   facet = indices of new facets where U is residing
%

[m,p] = size(U);
eps = 1.e-10;                          % termination tolerance

residue = Z-U*W;
gradient_U = residue*W';               % negative gradient
if norm(residue,'fro') < eps, return, end
```

```
tempA = eye(m)-ones(m)/(m-1);
for i = 1:p
    Proj_matrix = tempA;
    Proj_matrix(facet(i),:) = 0;
    Proj_matrix(:,facet(i)) = 0;
    temp0 = Proj_matrix*gradient_U(:,i);
    if norm(temp0) > eps*10
        temp0 = temp0/norm(temp0);
    else
        temp0 = temp0*0;
    end
    proj_grad_U(:,i) = temp0;           % normalized projected gradient
end

temp1 = trace(proj_grad_U*gradient_U');
temp2 = norm(proj_grad_U*W,'fro')^2;
alpha = temp1/temp2;                    % step size for steepest descent

Unew = U + alpha * proj_grad_U;
Unew = Unew.*(abs(Unew)>eps*10);

[row,column] = find(Unew<0);            % detecting zero crossing
if isempty(column) == 0,
    alpha_modify = alpha;
    for i = 1:length(column)
        temp = -U(row(i),column(i))/proj_grad_U(row(i),column(i));
        if abs(temp) < eps
            alpha_modify = 0;
            vertex = [row(i),column(i)];
        elseif  temp < alpha_modify
            alpha_modify = temp;
            vertex = [row(i),column(i)];
        end
    end
    U = U + alpha_modify*proj_grad_U;

    facet(vertex(2)) = vertex(1);       % facet change
```

```
else
    U = Unew;
end

U = U.*(abs(U)>eps);
```

---

Now we can solve the polytope approximation problem (5.17) by using Algorithms 3 and 4 alternatively between $U$ and $W$. Since this is a descent method in both direction, we will repeat the iteration until it converges. It is easy to see that limit point depends on the starting value $U^{(0)}$. Some numerical examples will be illustrated in Section 5.5.

## 5.4 Application to the nonnegative matrix factorization

In this section, we discuss the application of polytopy approximation to the NMF defined in (5.14). Compared with the above polytope approximation, this NMF has two different features. First, the matrix $A$ does not sit on the probability simplex $\mathcal{D}_m$ anymore. That is, the inner product used in the polytope approximation will be affected by the weight $\sigma(A)$. Second, each column sum of the matrix $W$ is no longer required to be one. Even so, we can easily generalize the idea investigated in the proceeding section can easily be generalized.

We first explain why the conventional method for finding the minimizers $U$ and $W$ might break down. Regardless of its deficiency, this traditional approach provides an insight into the geometric meaning which we shall discuss later. A constrained optimization problem for the NMF (5.14) is defined as:

$$\begin{aligned} \text{minimize} \qquad & f(U,V) = \frac{1}{2}\|\vartheta(A) - UV\|_F^2 = \frac{1}{2}\| \left( \vartheta(A) - U \underbrace{V\sigma(A)^{-1}}_{W} \right) \sigma(A)\|_F^2, \\ \text{subject to} \qquad & \mathbf{1}_m^\top U = \mathbf{1}_p^\top, \quad W \succeq 0, \end{aligned} \tag{5.28}$$

Again, temporarily ignoring the condition $W \succeq 0$, the Lagrangian is

$$L(U,W;\boldsymbol{\mu}) := \frac{1}{2}\langle(\vartheta(A) - UW)\sigma(A), (\vartheta(A) - UW)\sigma(A)\rangle + (\mathbf{1}_m^\top U - \mathbf{1}_p^\top)\boldsymbol{\mu}. \tag{5.29}$$

It follows that with respect to the Frobenius inner product over the product space $\mathbb{R}^{m\times p} \times \mathbb{R}^{p\times n}$

the partial gradients of $L$ are given by

$$\frac{\partial L(U,W;\boldsymbol{\mu})}{\partial U} = -(\vartheta(A) - UW)\sigma(A)^2 W^\top + \mathbf{1}_m \boldsymbol{\mu}^\top, \tag{5.30}$$

$$\frac{\partial L(U,W;\boldsymbol{\mu})}{\partial W} = -U^\top(\vartheta(A) - UW)\sigma(A)^2, \tag{5.31}$$

respectively. The first order optimality conditions requires that $U$ and $W$ must satisfy the equations

$$U = \left(\vartheta(A)\sigma(A)^2 W^\top - \mathbf{1}_m \boldsymbol{\mu}^\top\right)\left(W\sigma(A)^2 W^\top\right)^{-1}, \tag{5.32}$$

$$W = \left(U^\top U\right)^{-1} U^\top \vartheta(A), \tag{5.33}$$

and that the Lagrange multiplier $\boldsymbol{\mu}$ is given by

$$\boldsymbol{\mu}^\top = \frac{1}{m}\left(\mathbf{1}_n^\top \sigma(A)^2 W^\top - \mathbf{1}_p^\top W\sigma(A)^2 W^\top\right). \tag{5.34}$$

Note that the equation (5.33) is precisely the normal equation for the least squares problem

$$UW = \vartheta(A), \tag{5.35}$$

though we prefer to see further that $W \succeq 0$.

It would be instructive to consider the geometry associated with the simplest case when $m = 2$ and $p = 1$. For convenience, denote

$$\sigma(A) = \mathrm{diag}\{\sigma_1, \ldots, \sigma_n\}.$$

Then the NMF problem in (5.28) is equivalent to finding $\mathbf{u} \in \mathcal{D}_2$ and nonnegative scalars $w_1, \ldots, w_n$ such that

$$f(\mathbf{u}, w_1, \ldots, w_n) := \frac{1}{2}\sum_{i=1}^n \sigma_i^2 \|\vartheta(\mathbf{a}_i) - \mathbf{u}w_i\|_2^2, \tag{5.36}$$

is minimized. The relationships (5.32) and (5.33) become

$$\mathbf{u} = \sum_{i=1}^n \left(\frac{\sigma_i^2 w_i}{\sum_{i=1}^n \sigma_i^2 w_i^2}\vartheta(\mathbf{a}_i) - \frac{\sigma_i^2 w_i - \sigma_i^2 w_i^2}{2\sum_{i=1}^n \sigma_i^2 w_i^2}\mathbf{1}_2\right), \tag{5.37}$$

$$w_i = \frac{\mathbf{u}^\top \vartheta(\mathbf{a}_i)}{\mathbf{u}^\top \mathbf{u}}, \quad i = 1, \ldots, n. \tag{5.38}$$

Note that $w_i$ defined in (5.38) is precisely the projection of $\vartheta(\mathbf{y}_i)$ onto $\mathbf{u}$. The relevant geometry is sketched in Figure 5.3. In this case, the value of $w_i$ is guaranteed to be positive and is known

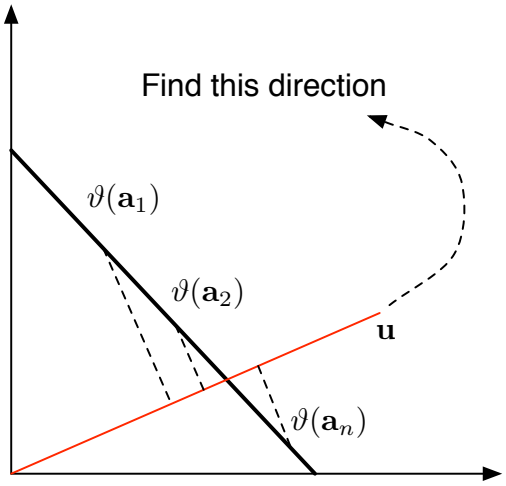


Figure 5.3: NNMF in $\mathbb{R}^2$ when $p = 1$.

as soon as $\mathbf{u}$ is given. It remains to turn the "knob" of a nonnegative vector $\mathbf{u}$ properly so as to satisfy (5.37) and $\mathbf{u} \in \mathcal{D}_2$. Nevertheless, this simple geometry cannot be generalized because the condition $W \succeq 0$ cannot be guaranteed by the projection (5.33) in higher dimensional space. In the following, we discuss how to solve NMF by the techniques developed for the polytope approximation in the preceding section.

### 5.4.1 Proximity map on the the simplicial cone

Substituting the variable $W = V\sigma(A)^{-1}$ in problem (5.28), we rewrite the objective functional as

$$h(U, W) = \frac{1}{2}\|(\vartheta(A) - UW)\sigma(A)\|_F^2 = \frac{1}{2}\sum_{i=1}^{n} \sigma_i^2 \|\vartheta(\mathbf{a}_i) - U\mathbf{w}_i\|_2^2, \tag{5.39}$$

which is equivalent to $f(U, V)$ in (5.14). It is clear that in order to minimize $h(U, W)$, each term in (5.39) must be minimized. Given a fixed $U$ and $\vartheta(A)$, we want to find $W$ so as to minimize $h(U, W)$. In other words, we need to best approximate each column of $\vartheta(A)$ within the simplicial cone of $U$.

To begin with, because $\|\vartheta(A)\|_1 = 1$, columns of $\vartheta(A)$ reside in the compact set $\mathcal{D}_m$, the nearest points of $\vartheta(A)$ on the simplicial cone of $U$ must be in a bounded set. We need not search far over the simplicial cone. All we need to do is to extend the columns of $U$ a little bit such that the proximity maps of $\vartheta(A)$ are included in a bounded convex set generated by the origin and columns of $U$. It is on this polytope where we can apply Algorithm 3 to find out all

nearest points. Mathematical details are described below.

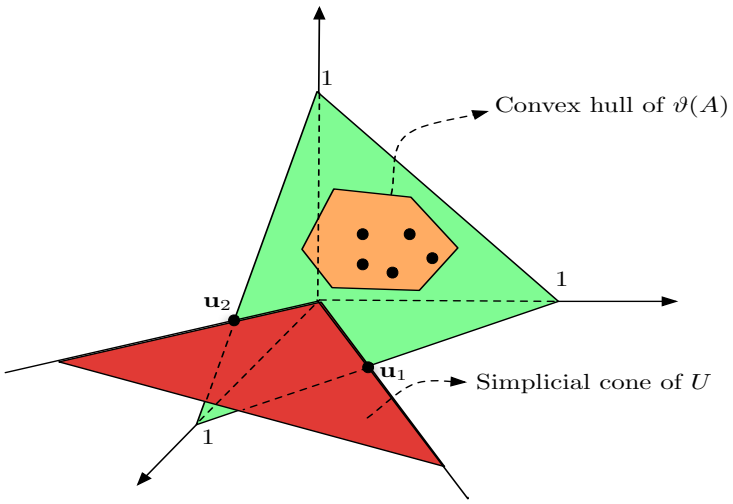


Figure 5.4: Simplicial cone of $U$ and convex hull of $\vartheta(Y) \in \mathbb{R}^{3\times n}$ with $p = 2$ and $n = 11$.

Given a large enough and fixed positive constant $\alpha$, define

$$\widetilde{U} = [0, \alpha\mathbf{u}_1, \ldots, \alpha\mathbf{u}_p] \tag{5.40}$$

to be the truncated simplicial cone. Columns of $\widetilde{U} \in \mathbb{R}^{m\times(p+1)}$ represent $p+1$ vertices of the polytope toward which $\vartheta(\mathbf{y}_i)$, $i = 1, \ldots, n$, is to find its nearest point on the simplicial cone of $U$. See Figure 5.4 for a geometric illustration. Now, apply Algorithm 3 to find the convex combination coefficients $\widetilde{W} \in \mathbb{R}^{(p+1)\times n}$ for the proximity map of $\vartheta(Y)$ onto $\text{conv}(\widetilde{U})$. To retrieve $W$, decompose $\widetilde{W}$ into two blocks,

$$\widetilde{W} = \begin{bmatrix} \mathbf{w}_0^\top \\ W_0 \end{bmatrix}, \tag{5.41}$$

where $\mathbf{w}_0^\top$ denotes the first row of $\widetilde{W}$ and $W_0 \in \mathbb{R}^{p\times n}$. It is easy to see that the nearest points of the matrix $\vartheta(Y)$ on the simplicial cone of $U$ is given by the product of matrices,

$$\widetilde{U}\widetilde{W} = UW, \tag{5.42}$$

with $W = \alpha W_0$. By construction, it is definitely true that $W \succeq 0$, but $W$ is no longer on $\mathcal{D}_p$. With a suitable $\alpha$ value, we have found this optimal $W$.

Indeed, since

$$\min_{\mathbf{w}_i \succeq 0} \|\vartheta(\mathbf{y}_i) - U\mathbf{w}_i\|_2 = \min_{\mathbf{w}_i \succeq 0} \|\vartheta(\mathbf{y}_i) - (\alpha U)\frac{\mathbf{w}_i}{\alpha}\|_2, \tag{5.43}$$

we should be able to scale down the length of the vector so that $\|\frac{\mathbf{w}_i}{\alpha}\|_1 \leq 1$ for all $i = 1, \ldots, n$ with a large enough positive scalar $\alpha$. That is, $\|W_0\|_1 \leq 1$ if the value $\alpha$ is large enough. It is important to note that if $\|W_0\|_1 \leq 1$, then the proximity map of $\vartheta(Y)$ must reside within $\widetilde{U}$. Consequently, it makes the application of Algorithm 3 sensible and optimal.

The following theorem suggests that a reasonable truncated simplicial cone should be such that $\alpha \geq 2\sqrt{m}$.

**Theorem 5.4.1** *For matrices $\vartheta(Y) \in \mathbb{R}_+^{m\times n}$, $U = \vartheta(U) \in \mathbb{R}_+^{m\times p}$, and $i = 1, \ldots, p$, if $\hat{\mathbf{w}}_i$ is the minimizer of*

$$\min_{\mathbf{w}_i \succeq 0} \|\vartheta(\mathbf{y}_i) - U\mathbf{w}_i\|_2 \tag{5.44}$$

*then $\|\hat{\mathbf{w}}_i\|_1 \leq 2\sqrt{m}$.*

**Proof.** Since $\min_{\mathbf{w}_i \succeq 0} \|\vartheta(\mathbf{y}_i) - U\mathbf{w}_i\|_2 \leq \|\vartheta(\mathbf{y}_i)\|_2$, observe that at the minimizer $\hat{\mathbf{w}}_i$ of (5.44) we have

$$\|\vartheta(\mathbf{y}_i) - U\hat{\mathbf{w}}_i\|_2 \leq \|\vartheta(\mathbf{y}_i)\|_2 \leq \|\vartheta(\mathbf{y}_i)\|_1 = 1, \tag{5.45}$$

and

$$\|\vartheta(\mathbf{y}_i) - U\hat{\mathbf{w}}_i\|_2 \geq \|U\hat{\mathbf{w}}_i\|_2 - \|\vartheta(\mathbf{y}_i)\|_2 \geq \|U\hat{\mathbf{w}}_i\|_2 - \|\vartheta(\mathbf{y}_i)\|_1 = \|U\hat{\mathbf{w}}_i\|_2 - 1. \tag{5.46}$$

It follows that

$$\|U\hat{\mathbf{w}}_i\|_2 \leq 2. \tag{5.47}$$

Note that

$$\frac{1}{\sqrt{m}}\|U\hat{\mathbf{w}}_i\|_1 \leq \|U\hat{\mathbf{w}}_i\|_2. \tag{5.48}$$

Since $\vartheta(U) = U$, then

$$\begin{aligned} \|U\hat{\mathbf{w}}_i\|_1 &= \|\mathbf{u}_1\hat{\mathbf{w}}_{i1}\|_1 + \cdots + \|\mathbf{u}_p\hat{\mathbf{w}}_{ip}\|_1 \\ &= |\hat{\mathbf{w}}_{i1}| + \cdots + |\hat{\mathbf{w}}_{ip}| \\ &= \tfrac{1}{\sqrt{m}}\|\hat{\mathbf{w}}_i\|_1 \end{aligned} \tag{5.49}$$

where $U = [\mathbf{u}_1, \ldots, \mathbf{u}_p]$ and $\hat{\mathbf{w}}_i = [\hat{\mathbf{w}}_{i1}, \ldots, \hat{\mathbf{w}}_{ip}]$. Then, it follows from formulae (5.47), (5.48), and (5.49) that $\|\hat{\mathbf{w}}_i\|_1 \leq 2\sqrt{m}$. □

Going back to the original NMF, given $A$ and $U$, our Algorithm 3 computes a $p \times n$ non-negative matrix,

$$V = W\sigma(A), \tag{5.50}$$

which is a *global minimizer* for the objective functional $f(U, V)$. This is in contrast with the nature of most available NMF algorithms where only a local minimizer is attained. It is especially so in the well known Lee and Seung's multiplicative update rule [120],

$$V^+ = V. * (U^\top A)./(U^\top UV). \tag{5.51}$$

We also emphasize that the cost of Algorithm 3 is comparable with that of the multiplicative update since the former involves only a few inner products in the process of seeking the supporting hyperplanes. We do point out that extensive indexing in our algorithm might become a major shortcoming if the code is to executed on a machine with slow I/O capacity.

### 5.4.2 Alternating direction search

Now we discuss two ways to update $U$ for the NMF from a given $W \succeq 0$.

The first approach is by means of the steepest descent method. Since the minimum of a function over a larger domain generally is lower than that over a smaller domain, it is reasonable to expand the "angle" of the simplicial cone as wide as possible, to the extent that it is beneficial to make the simplicial cone of $U$ touch the boundary of $\mathcal{D}_m$. That is, we could assume that columns of $U$ sit on the boundary $\partial\mathcal{D}_m$ to begin with. See the idea demonstrated in Figure 5.2 and Figure 5.4. With this in mind, we compute the gradient of $h(U, W)$ with respect to U and obtain

$$\nabla_U h(U, W) := \frac{\partial h}{\partial U} = -(\vartheta(A) - UW)\sigma(A)^2 W^\top. \tag{5.52}$$

After this calculation, we can apply the same approach described earlier in Section 5.3.2 to calculate the projected gradient of $\nabla_U h(U, W)$ onto proper facets and adjust $U$ along the boundary $\partial\mathcal{D}_m$ by means of the steepest descent method.

The second approach is to simply interchange the roles of $U$ and $W$ and repeat the procedure described in the preceding section. More precisely, we rewrite

$$f(U, V) = \frac{1}{2}\left\| \left( \vartheta(A^\top) - \vartheta(V^\top)(\underbrace{\sigma(V^\top)U^\top\sigma(A^\top)^{-1}}_{\Phi}) \right) \sigma(A^\top) \right\|_F^2. \tag{5.53}$$

Under this transformation, we want to calculate the proximity map of $\vartheta(A^\top)$ within the simplicial cone of $\vartheta(V^\top)$. After applying the procedures proposed in Section 5.4.1 to calculate the

**Algorithm 5**: NMF $[U, V] = \text{NMF}(U)$

**Input**: $A \in \mathbb{R}_+^{m\times n}$, $p \in \mathbb{Z}$, $p < \min(m, n)$, and $T \in \mathbb{Z}$

**Output**: $U \in \mathbb{R}_+^{m\times p}$ and $V \in \mathbb{R}_+^{p\times n}$

5.1 **begin**

5.2 Randomly generate a $m \times p$ matrix $U$;

5.3 **for** $i \leftarrow 1$ **to** $T$ **do**

`(Given the matrix` $U$`, find the matrix` $V$ `to minimize` $f(U, V)$`)`

5.4 Suitably rescale columns of $U$;

5.5 Calculate the proximity map of $\vartheta(A)$ onto $\text{conv}([0, U])$;

5.6 Formulate the matrix $V$ from the proximity map of $\vartheta(A)$ onto $\text{conv}([0, U])$ ;

`(Given the matrix` $V$`, find the matrix` $U$ `to minimize` $f(U, V)$`)`

5.7 Suitably rescale columns of $V^\top$;

5.8 Calculate the proximity map of $\vartheta(A^\top)$ onto $\text{conv}([0, V^\top])$;

5.9 Formulate the matrix $U$ from the proximity map of $\vartheta(A^\top)$ onto $\text{conv}([0, V^\top])$ ;

5.10 **end**

5.11 **end**

unique and optimal simplicial combination coefficients $\Phi \in \mathbb{R}^{p\times m}$, the optimal $U$ is given by

$$U = \left(\sigma(V^\top)^{-1}\Phi\sigma(Y^\top)\right)^\top. \tag{5.54}$$

Again this process is similar to Lee and Seung's multiplicative update,

$$U^+ := U. * (YV^\top)./(UVV^\top), \tag{5.55}$$

except that our $U$ obtained through (5.54) is a global minimizer for $f(U, V)$ from a fixed $V$ which is calculated in (5.50). The numerical approach for solving NMF could be illustrated via Algorithm 5.

Finally, programming details for our NMF algorithm are listed in its following MATLAB code. We note that the determination of a suitable low rank $p$ is itself an important question which, thus far, has not been addressed. In the current code, the value of $p$ with $p < \min\{m, n\}$ is assumed to be given. Note also that, although our scheme guarantees that each sweep of the ADI gives rise to a descent objective value, we have not furnished a comprehensive convergence analysis of the scheme. Our viewpoint is that because an absolute optimizer is often preferred but not absolutely needed in NMF applications, an improved approximate solution is generally acceptable. Instead of a rigorous stopping criterion, we specify a maximal allowable number $T$ of iterations for the ADI in the code.

In conclusion, suppose $A \in \mathbb{R}_+^{m\times n}$. Let $p$ be an integer with $p < \min\{m, n\}$ and $T$ be max-

imal allowable number of iterations. We want to apply the Sekitani and Yamamato technique alternatively to compute the proximity maps of $\vartheta(A)$ onto the simplicial cones of $U \in \mathbb{R}^{m\times p}$ or the proximity maps of $\vartheta(A^\top)$ onto the simplicial cones of $V \in \mathbb{R}^{p\times n}$. The resulting nonnegative matrices $U$ and $V$ therefore progress toward minimizing $f(U,V)$. The MATLAB code are listed below for the convenience of explanation.

---

```
%
% Input:
%
%   Z = points to be aprroximated
%   T = maximal allowable number of iterations
%
% Output:
%
%   U = vertices of new polytope
%   V = convex combination coefficients
%

SigmaA = sum(A); Z = A*diag(1./SigmaA);     % pull back of A
SigmaAt = sum(A'); Zt =A'*diag(1./SigmaAt); % pull back of A'
alpha = 2*sqrt(m);                          % scaling factor

U = rand(m,p);                              % initialization of U

for iter = 1:T                              % loop of ADI T times
    sigmaU = sum(U);
    sigmaU(find(sigmaU~=0))=1./sigmaU(find(sigmaU~=0));
    U = U*diag(sigmaU);                     % pull-back of U

    Utilde = [zeros(m,1),alpha*U];          % simplicial cone of U

    S = []; C = [];
    for i = 1:n
        A = Z(:,i);
        U0 = Utilde - A*ones(1,p+1);
        active = 1:p+1;
        [s,c] = sekitani(U0,active);
```

```
        S = [S,s+A];
        C = [C,c];
    end

    W = alpha*C(2:p+1,:);

    V = W*diag(SigmaA);                    % optimal V, given U
    V = V.*(V>eps*10);

    if iter == T
       return                              % end of iteration
    end

    sigmaVt = sum(V');
    sigmaVt(find(sigmaVt~=0))=1./sigmaVt(find(sigmaVt~=0));
    Vt = V'*diag(sigmaVt);

    Vtilde = [zeros(n,1),alpha*Vt];        % simplicial cone of V

    S = []; C = [];
    for i = 1:m
        A = Zt(:,i);
        V0 = Vtilde - A*ones(1,p+1);
        active = 1:p+1;
        [s,c] = sekitani(V0,active);
        S = [S,s+A];
        C = [C,c];
    end

    U = (alpha*C(2:p+1,:)*diag(SigmaAt))';  % optimal U (w/t scaling), given V
    U = U.*(U>eps*10);
end
```

---

Both matrices $W$ in (5.42) (and hence $V$) and $\Phi$ in (5.53) (and hence $U$) are readily available from the routine SEKITANI which we stress again solves repeatedly nonnegative least squares problems without any matrix inversion. More importantly, in each alternating direction, our

method guarantees to compute the unique global minimizer, which is remarkable advance compared to most with the conventional procedures.

We do have to point out two possible drawbacks of our algorithm. Firstly, it is hard to predict beforehand the depth of recursion needed for the calculation in the Sekitani and Yamamato algorithm. When $p$ is large, the recursion might take more steps to complete the process in each direction and thus drives up the overhead. Fortunately, in most NMF applications, $p$ is required to be quite small. In this case, only a few vertices of the polytopes, namely, the truncated simplicial cones of either $U$ or $V^\top$, need to be checked for support hyperplanes. The recursion then can be accomplished reasonably fast. Secondly, while computing the proximity map by SEKITANI, our procedure thus far deals with one column of $A$ a time. This is inefficient when $m$ or $n$ is large, which unfortunately is the case for NMF applications. On the other hand, since each column can be computed independently of each other, the computation is embarrassingly parallelizable. It should require no extra programming effort to load the work on a parallel computer. To put it differently, we should be able to compute the proximity maps of multiple columns simultaneously. We should be able to generalize the Sekitani and Yamamoto algorithm so as to handle multiple columns at the same time or formulate the computations it in terms of BLAS3 operations. For example, akin to the scalar equation (5.23), we could conveniently summarize $n$ hyperplanes in $\mathbb{R}^m$ by a single matrix equation

$$N^\top \mathbf{x} = \mathbf{c}, \tag{5.56}$$

where each column of $N \in \mathbb{R}^{m\times n}$ represents a normal vector. Many of the inequalities in Algorithm 2 should have their counterparts. We are currently exploring this possibility and will report our investigation more formally somewhere else.

## 5.5 Numerical experiments

In this section, we demonstrate the working of our algorithms by providing several numerical experiments.

**Example 5.5.1** *This first example is to demonstrate the the polytope approximation on the probability simplex graphically. Even though the theory works for general $m$, for the purpose of illustration, we focus our attention on $\mathbb{R}^3$, that is, $m = 3$, so that we can see the actions. We randomly generate $n = 18$ points on the simplex. In general, for low rank polytope approximation, we would desire $p < \min\{m, n\}$. But $m$ is so low that we choose $p = 3$. The polytope approximation becomes a problem of seeking a "triangle" to enclose the 18 given points [78]. Needless to say, the probability simplex $\mathcal{D}_3$ itself satifies the condition. Our task is to discover other triangles that encircle these points.*

*Applying Algorithm 3 followed by Algorithm 4 iteratively 100 times, we see in the left drawing of Figure 5.5 that a "minimal" triangle with vertices on $\partial\mathcal{D}_3$ is found. The triangle is minimal in the sense that three points reside on the three different sides of the triangle individually. Thus, there is no more room for reducing the triangle further without leaving some points outside. The dynamics of adjustment by our algorithm on the triangles with vertices on $\partial\mathcal{D}_3$ is illustrated in the right drawing of Figure 5.5.*

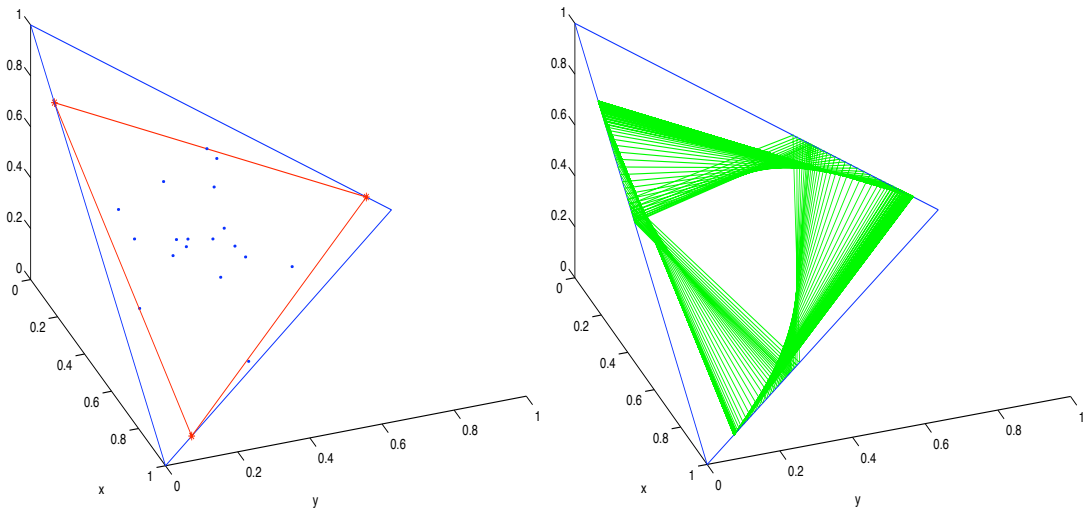


Figure 5.5: Triangle (polytope) enclosing a prescribed set of points on $\mathcal{D}_3$.

*Just like most iterative methods, the limiting behavior of our method depends on the initial values. It is possible that the iteration ends up with a triangle which does not include all given points to its interior. In this case, the method has reached a local solution.*

**Example 5.5.2** *We need to stress that Algorithm 5 is a method which finds global minimizer in each iteration. To demonstrate this point, we design our experiment as follows. Let $Y \in \mathbb{R}^{m\times p}$ and $Z \in \mathbb{R}^{p\times n}$ with $p < \min\{m, n\}$ be two randomly generated nonnegative matrices. Define $A = YZ$ which will be our target data matrix. We want to compare the objective values $f(U, V)$ per iteration with the popular Lee-Seung multiplicative update algorithm by starting from the same initial value $(U_0, V_0)$.*

*Given a test case with $m = 100$, $n = 50$, and $p = 5$ in 100 iterations, Figure 5.6 demonstrates that our algorithm consistently produces much closer approximation to $A$ per iteration than the Lee-Seung algorithm. We report two random cases. In the case on the left side of Figure 5.6, the*

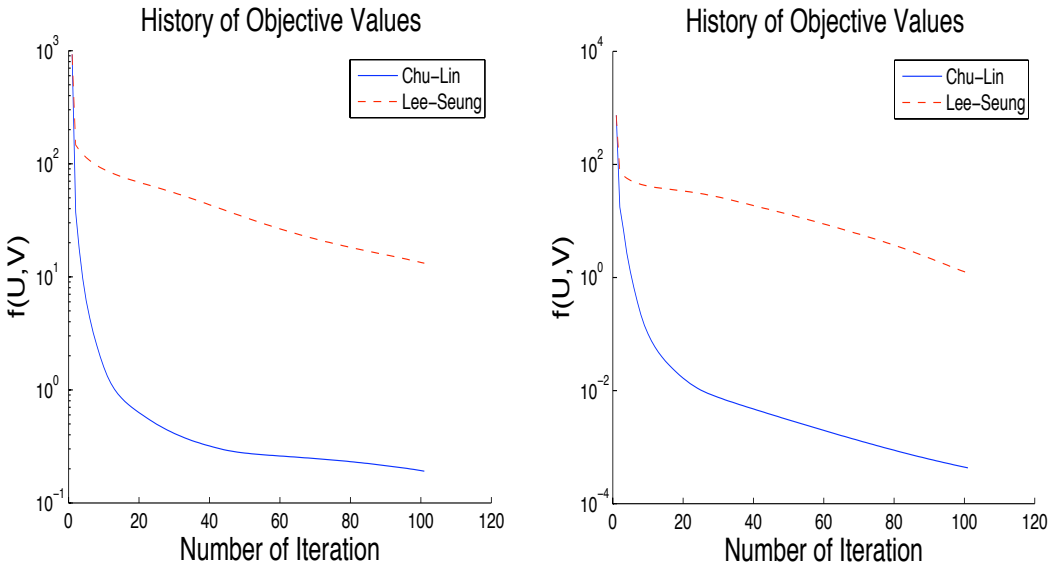


Figure 5.6: Objective values $f(U, V)$ per iteration by the Chu-Lin algorithm versus the Lee-Seung algorithm.

*objective value is reduced to* 0.1909 *by our method in comparison with* 13.1844 *obtained by the Lee-Seung method, although these values continue to be improved with more iterations. In the case on the right side of Figure 5.6, the objective value attained by our method is* $3.3035 \times 10^{-4}$ *in comparison with* 1.1989 *achieved by Lee-Seung. In addition to these two experiments, we consistently discover that our method always produces an impressive improvement by multiple orders.*

**Example 5.5.3** *To further demonstrate the effectiveness of our method compared with the Lee-Seung algorithm, we randomly generate a* $50 \times 30$ *matrix* $Y$ *of rank* 20*. We systematically compute its nonnegative matrix approximation of rank* $p$*,* $p = 1, \ldots, 15$*, but allowing only 20 iterations. Each point in Figure 5.7 represents the objective value* $f(U, V)$ *after 20 iterations of the corresponding method. It shows that our method decreases the objective values* $f(U, V)$ *more rapidly than the Lee-Seung algorithm. Also, our method decreases the objective value as* $p$ *increases while the Lee-Seung method does not seem to be sensitive to* $p$ *during the first 20 iterations.*

*Other than showing the results after 20 iterations, the continuous curve in the middle of Figure 5.7 represents the objective values computed by the Lee-Seung algorithm after 100 iterations. We choose not to draw the objective values obtained by our method after 100 iterations because they are essentially the same as those after 20 iterations except the mild differences at the second decimal digit. This experiment evidences that not only our method yields a better*

*factorization than the Lee-Seung method, but also our algorithm converges more rapidly to an optimal solution in its early iterations.*

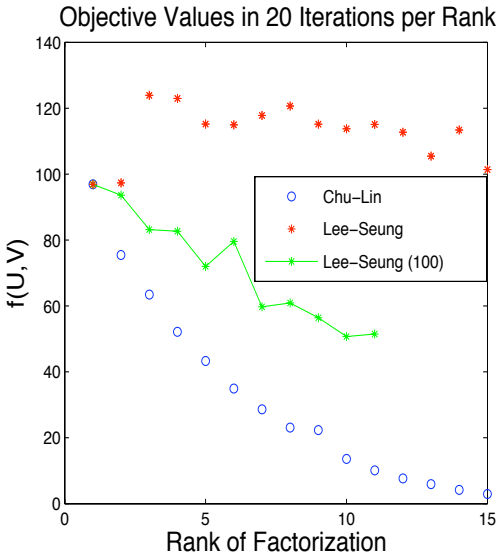


Figure 5.7: Objective values $f(U, V)$ in 20 iterations per specified rank by the Chu-Lin and the Lee-Seung algorithms.

*It is premature to conclude the complexity of our algorithm at this stage because we have not yet fully exploited the implementation. Algorithm 5 is only a rough working code. Figure 5.8 depicts the CPU time per sweep in ADI of our algorithm. It seems to suggest consistently that the complexity of our algorithm is exponential in $p$ for the same $50 \times 30$ matrix $Y$ of rank 20. Recall that the essential cost in Algorithm 5 is the repeated calling of the routine* SEKITANI. *The exponential growth of complexity in $p$ seems to result from the recursive nature of the Sekitani and Yamamoto algorithm. Even though our algorithm could be sped up by parallelization and vectorization, we could expect only either the downward shift or the decreasing slope of the curve, but the exponential dependence on $p$ should be the same.*

**Example 5.5.4** *The "Swimmer" data set proposed in [56] is meant to be the exemplary and ultimate test bed for any NMF algorithms. It consists of a set of black-and-white stick figures satisfying the so called separable factorial articulation criteria. In particular, each figure, placed in a frame of $32 \times 32$ pixels, is made of a "torso" of 12 pixels in the center and four "limbs" of six pixels. With limbs pointing towards one of four artificial directions, there are a total of 256 figures in the collection. A subset of these images from the Swimmer data set are depicted*

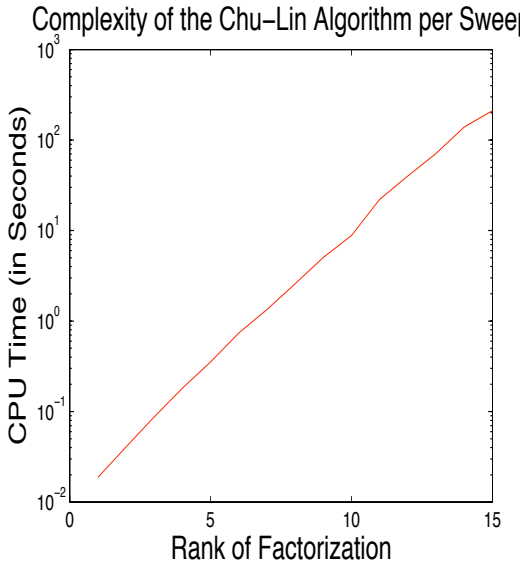


Figure 5.8: Complexity of the Chu-Lin algorithm measured in CPU time per sweep of the alternating optimization.

*in Figure 5.9 [140] .*

*To start with out test, we first vectorize each image into a column such that our target matrix $A$ is of size $1024 \times 256$. The original data matrix contains only two integer values, $0$ and $255$, which we convert without loss of generality to binaries. It should be stressed that the numerical rank of $A$ is only $13$, but we shall decompose it with $p = 17$ with the hope of retrieving the $17$ standard parts. It has been speculated ever since the NMF concept has been conceived that the NMF should be able to extract the standard parts embedded in an articulation model, so it remains to see whether we can recover the $17$ basic parts that make up these swimmers. The effect of the overestimating the rank of $A$ in the NMF is something that deserves further study. This actually becomes the issue of nonnegative rank which will be discussed in Chapter 8.*

*Figure 5.10 depicts the $17$ intrinsic "parts" obtained by our algorithm with only $10$ iterations. Observe that not all limbs are separated from the torso, but all positions are clearly identified. The gray region indicates a drawback by using NMF in general for this type of data because the calculation brings in fractional values whereas the model should involve only binaries. Note that frame $16$ contains entirely zero information, perhaps having something to do with the overestimation of the rank in the NMF.*

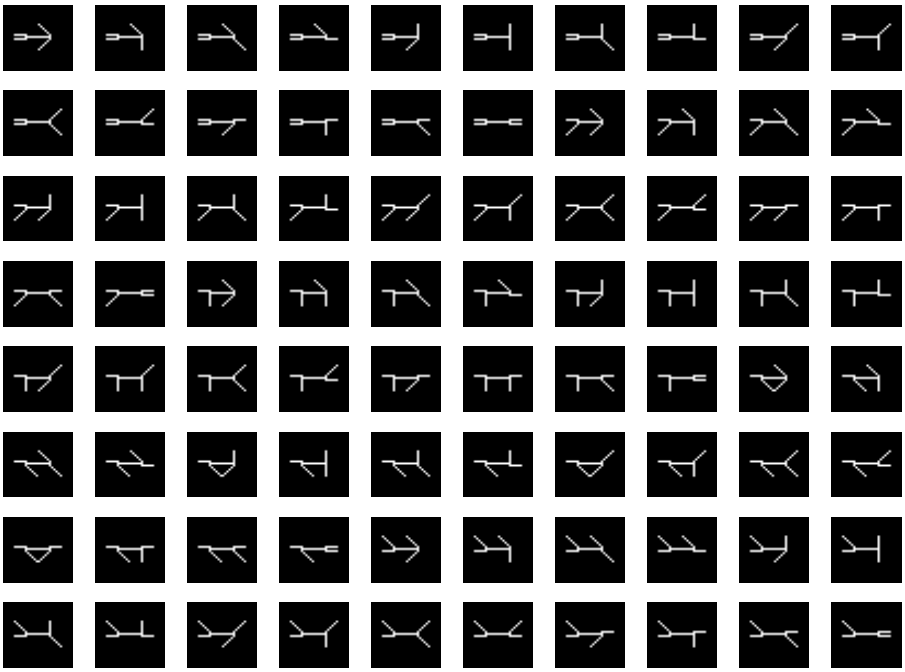

Figure 5.9: 80 sample images from the swimmer database.

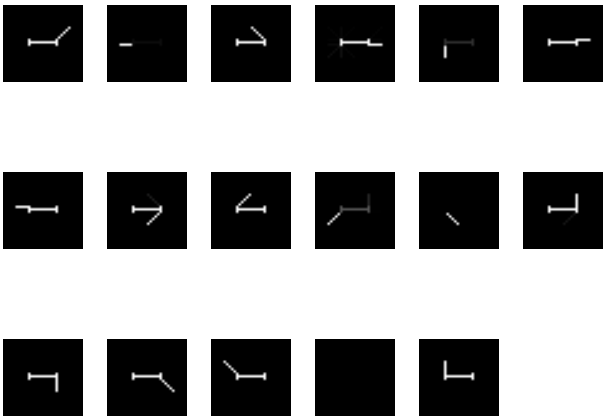

Figure 5.10: 17 "parts" decomposed from the swimmer database by Algorithm 5 in 10 iterations.

## 5.6 Conclusion

Low rank factorization is another type of inverse problem where the information in the observed data is to be reconstructed, approximated, or retrieved via simpler representations. It is often the case that the low rank representations are structured, such as nonnegativity, which thus

imposes difficulties. This chapter studies the NMFs from the viewpoint of geometric approximation.

For a given polytope in the first orthant, we propose the idea of pulling the polytope back to the probability simplex. This action has the advantages of not only transforming polytopes into a more manageable compact set but also facilitating the calculation of the proximity map which identifies the unique nearest point on the polytope to an arbitrarily given point in finitely many steps. Two kinds of low dimensional polytope approximations are investigated in the discussion. The first focuses entirely on the probability simplex whereas the second on the truncated simplicial cones.

Our main thrust in this investigation is to apply the low dimensional polytope approximation to the nonnegative matrix factorization problem. We calculate the best approximation by using the proximity maps to compute the unique global minimization on the desired polytope in each alternating direction. Our derivation is driven by geometry whereas the theory is supported by the Hahn-Banach separation theorem. Since the NMF is a nonlinear optimization to begin with, there is no guarantee that our algorithm solves the original problem to its absolute optimality. Numerical results, nonetheless, seem to suggest that our method produces much smaller residual errors in the factorization than the popular Lee-Seung multiplicative updating scheme.

The main engine for computing the proximity map in our method is currently conducted on a column by column basis. As such, the calculation is less competitive in speed with the Lee-Seung algorithm which can be executed under matrix-to-matrix operations. We are currently working on the prospects to compute the proximity map for multiple columns simultaneously. It is is realizable, it would be an added power to our method. This would be a major topic worthy for the next phase of study.

# Chapter 6

# Integer Matrix Factorization via Rank-one Approximation

## 6.1 Overview

In Section 3.1 we have already explained the fundamental importance of matrix factorization in modern sciences and technology. For that purpose, we investigate in this chapter the notion of factorization with entries restricted to integers or binaries, where the "integer" could be either the regular ordinal integers or just some nominal labels. When the entries are chosen from the set $\mathbb{Z}_2 := \{0, 1\}$, i.e., binaries, a non-orthogonal binary decomposition by recursive partitioning has recently been proposed in [111]. The associated code PROXIMUS, written in the *C* language with considerable thoughts given to efficient data structure, storage, and movement within the system, has been distributed widely in the field. In this research, we generalize principles to general integer matrices.

Being discrete in nature, such a factorization or approximation cannot be accomplished by conventional techniques. Built upon a basic scheme of rank-one approximation, an approach that recursively splits (approximates) the underlying matrix into a sum of rank-one matrices with discrete entries is proposed. The mechanism presented in this chapter can handle multiple types of data. For demonstration purposes, the subsequent discussion concentrates mainly on binary-integer factorizations. But the notion is readily generalizable with slight modifications to, for example, integer-integer factorizations.

As is always the case in approximation, we need a metric $d$ to measure nearness or similarity. Because we could be dealing with a basket of different data types in a single application, the metric itself might be a combination of disparate measures, one for each category in the data array. For ordinal data, we also need to decide whether the spacing between the categorial values signifies all levels of the variables or not. If the values are not equally spaced, using integers to

represent these values has the advantage of numerically scaling in the spacing disparities. The commonly used *Hamming distance* or *Euclidean distance* can be refined to reflect the different situations. We limit our discussion in this study to these two metrics, although our concept can easily be generalized to other means of measurement.

We shall demonstrate several interesting applications of the IMF in this presentation. But more importantly, it is the details on the computational aspects of IMF that deserve our attention. To make our approach effectual, we need to analyze several essential components separately first before putting them to work in conjunction. We thus organize this chapter as follows: We begin in Section 6.2 with a basic scheme that does ADI search for rank-one approximation. We show that in each alternating direction an optimal solution is attainable in closed form. This scheme serves as a building block by which we construct in Section 6.3 a divide-and-conquer strategy that ultimately factorize the given matrix $A$. The method gradually breaks down the data matrix to submatrices in a recursive way. Although it seems that we are dealing with numerals, our method actually can handle categorical variables. The splitting mechanism is initially based a one-way comparison with a selected, but not necessarily justified, representative. A suitable criterion for assessing the validity of the proposed splitting, therefore, must also be addressed. A prototype code is sketched in Algorithm 8 to demonstrate how these components should be assembled together. In practical application, the code might need to be modified to reflect the different nature of the underlying data and the metric, but the general concept is the same. Of particular interest and significance in our investigation is the discovery of the ordering for rank one approximations which, in a sense, is the analogy for discrete data to the ordering of singular values for continuous data. With such a perspicacity in hand, a truncated low rank factorization (for discrete data) analogous to the truncated singular value decomposition is also obtained. We have experimented our method with quite a few real-world data sets from cluster analysis and pattern discovery. We select to report only a few in Section 6.4. Our numerical experiences seem to suggest that our approach is easy to program, converges
considerably fast even with a large data matrix, and can handle multiple types of data.

## 6.2 Basic rank-one approximation

The optimal procedure in our IMF algorithm is made of a sequence of rank-one approximations that minimize the difference of a rank-one matrix from a given matrix. In this section, we discuss how such an approximation is achieved. Given a suitable metric $d$ corresponding the data type, the general idea for computing the rank-one approximation is to employ the notion of ADI as in Algorithm 6.

Note that we consider only the case where $\mathbf{u}$ is binary. It is obviously that such iterations

**Algorithm 6**: Rank-one approximation with general metric $d$

**Input**: Matrix $A \in \mathbb{Z}^{m\times n}$ and initial vector $\mathbf{v} \in \mathbb{Z}^{1\times n}$
**Output**: Vectors $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$ and $\mathbf{v} \in \mathbb{Z}^{1\times n}$ that minimize $d(A, \mathbf{uv})$

6.1 **begin**
6.2 **repeat**
6.3 For the fixed $\mathbf{v}$, find $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$ to minimize $d(A, \mathbf{uv})$;
6.4 For the fixed $\mathbf{u}$, find $\mathbf{v} \in \mathbb{Z}^{1\times n}$ to minimize $d(A, \mathbf{uv})$;
6.5 **until** *convergence* ;
6.6 **end**

can start with an initial vector $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$, instead of $\mathbf{v} \in \mathbb{Z}^{1\times n}$. Since Algorithm 6 is a descent method by finding out an optimal solution at each step and there are only finitely many choices for $\mathbf{u}$, the procedure is supposed to terminate within finite iterations. The real issue at stake is to derive a scheme to pick up the minimizer at each step. We analyze below the best choice with respect to the two commonly used metrics.

### 6.2.1 Approximation with Hamming metric

The *Hamming metric* (*Hamming distance*) between two arrays of the same length is defined as the minimum number of substitutions required to transform one array into the other. We certainly can use the very same metric to count the difference between two matrices. For convenience, we introduce three operators, $\mathsf{match}(\mathbf{x}, \mathbf{y})$, $\mathsf{zeros}(\mathbf{x})$ and $\mathsf{mode}(\mathbf{x})$, to count the number of matching elements between vectors $\mathbf{x}$ and $\mathbf{y}$, the number of zero entries in the vector $\mathbf{x}$, and the most frequent entry of the vector $\mathbf{x}$, respectively. Also, we adopt the notation $A(i,:)$ for the $i$th row of a matrix $A$. Given a vector $\mathbf{v}$ (or $\mathbf{u}$), the following two results show that the optimal solution $\mathbf{u}$ (or $\mathbf{v}$) for minimizing $d(A, \mathbf{uv})$ can be found in closed form with respect to the Hamming metric.

**Theorem 6.2.1** *Given $A \in \mathbb{Z}^{m\times n}$ and $\mathbf{v} \in \mathbb{Z}^{1\times n}$, then among all possible $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$, the measure $d(A, \mathbf{uv})$ is minimized at $\mathbf{u}^*$ whose ith entry $u_i^*$, $i = 1, \ldots, m$, is defined by*

$$u_i^* := \begin{cases} 1, & \text{if } \mathsf{match}(A(i,:), \mathbf{v}) \geq \mathsf{zeros}(A(i,:)), \\ 0, & \text{otherwise.} \end{cases} \tag{6.1}$$

**Proof.** For any $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$, observe that

$$d(A, \mathbf{uv}) = \sum_{i=1}^{m} d(A(i,:), u_i\mathbf{v}). \tag{6.2}$$

Thus minimizing $d(A, \mathbf{uv})$ is equivalent to minimizing each individual term $d(A(i,:), u_i\mathbf{v})$ for $i = 1, 2, \cdots, m$. From the relationships that $\mathsf{match}(A(i,:), \mathbf{v}) = n - d(A(i,:), \mathbf{v})$ whereas $\mathsf{zeros}(A(i,:)) = n - d(A(i,:), 0)$, it is clear that the choice in (6.1) is optimal. □

**Theorem 6.2.2** *Given $A \in \mathbb{Z}^{m\times n}$ and $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$ with $\mathbf{u} \neq 0$, then among all $\mathbf{v} \in \mathbb{Z}^{1\times n}$, the measure $d(A, \mathbf{uv})$ is minimized at $\mathbf{v}^*$ whose $j$th entry $v_j^*$, $j = 1, \ldots, n$, is defined by*

$$v_j^* := \mathsf{mode}(A_2(:, j)), \tag{6.3}$$

*where $A_2$ is the submatrix composed of all $A(i,:)$ whose corresponding $u_i$ is 1.*

**Proof.** For a given vector $\mathbf{u}$, we may assume without loss of generality that

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \end{bmatrix}$$

with $\mathbf{u}_1 = [0, \cdots, 0]_{1\times s}^\top$ and $\mathbf{u}_2 = [1, \cdots, 1]_{1\times(m-s)}^\top$. Accordingly, partition $A$ into two parts

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}$$

with $A_1 \in \mathbb{Z}^{s\times n}$ and $A_2 \in \mathbb{Z}^{(m-s)\times n}$. For any vector $\mathbf{v} \in \mathbb{Z}^{1\times n}$, observe that

$$\begin{aligned} d(A, \mathbf{uv}) &= d(A_1, \mathbf{u}_1\mathbf{v}) + d(A_2, \mathbf{u}_2\mathbf{v}) \\ &= d(A_1, 0\mathbf{v}) + \sum_{j=1}^{n}\sum_{i=1}^{m-s} d(A_2(i,j), v_j), \end{aligned}$$

Minimizing $d(A, \mathbf{uv})$ is equivalent to minimizing $\sum_{i=1}^{m-s} d(A_2(i,j), v_j)$ for each $j = 1, \ldots, n$. The optimal choice for $v_j$ is certainly the most frequently occuring entry in the $j$th column of $A_2$. □

Note that in both theorems above, the optimal solutions may not be unique if there is a tie. For instance, assigning $u_i^* = 0$ instead of 1 when $\mathsf{match}(A(i,:), \mathbf{v}) = \mathsf{zeros}(A(i,:))$ will not affect the optimal objective value $d(A, \mathbf{u}^*\mathbf{v})$ in Theorem 6.2.1. Likewise, multiple choices for $v_j^*$ happen in Theorem 6.2.2 when there are more than one most frequent entries in the $j$th column of $A_2$.

Equipped with the closed-form solutions characterized in Theorems 6.2.1 and 6.2.2, the rank-one approximation with the Hamming metric is now taking shape in the form of Algorithm 7. We name this algorithm VOTE because the choice of $\mathbf{u}$ (and $\mathbf{v}$ if $A$ is a binary matrix) is a matter of majority rule between the dichotomy of zeros and matches. A function similar to the

**Algorithm 7**: Rank-one factorization with Hamming metric: $[\mathbf{u}, \mathbf{v}] = \text{VOTE}(A)$

**Input**: Matrix $A \in \mathbb{Z}^{m \times n}$

**Output**: Vectors $\mathbf{u} \in \mathbb{Z}_2^{m \times 1}$ and $\mathbf{v} \in \mathbb{Z}^{1 \times n}$ such that $d(A, \mathbf{uv})$ is minimized

```
7.1  begin
7.2      initialization;
7.3      v ← randomly selected from one row of matrix A;
7.4      z ← numbers of zeros in A per row;
7.5      repeat
7.6          vold ← v;
7.7          m ← numbers of matches between v and each row of A;
7.8          if m_i ≥ z_i then
7.9              u_i ← 1;
7.10         else
7.11             u_i ← 0;
7.12         end
7.13         if u = 0 then
7.14             v ← 1;
7.15         else
7.16             ind ← find(u);
7.17             v_i ← mode(A(ind, i));
7.18         end
7.19     until vold = v ;
7.20 end
```

command "find" in Matlab, where **ind** = find(**x**) locates all nonzero elements of array **x** and returns the linear indices of those elements in the vector **ind**, proves handy in the coding.

It is interesting to note that, if $A$ is a binary matrix, then the definition (6.3) $\mathbf{v}^*$ is exactly the same as (6.1).

**Lemma 6.2.1** *Given $A \in \mathbb{Z}_2^{m \times n}$ and $\mathbf{u} \in \mathbb{Z}_2^{m \times 1}$, then among all $\mathbf{v} \in \mathbb{Z}_2^{1 \times n}$, the measure $d(A, \mathbf{uv})$ is minimized at $\mathbf{v}^*$ whose $j$th entry $v_j^*$ is given by*

$$v_j^* = \begin{cases} 1, & \textit{if } \mathit{match}(A(:,j), \mathbf{u}) \geq \mathit{zeros}(A(:,j)), \\ 0, & \textit{otherwise.} \end{cases} \tag{6.4}$$

**Proof.** We present an alternative argument by bringing forth the relationships that

$$\begin{aligned} \mathsf{match}(A(:,j), \mathbf{u}) &= \mathsf{match}(A_1(:,j), 0) + \mathsf{match}(A_2(:,j), \mathbf{1}), \\ \mathsf{zeros}(A(:,j)) &= \mathsf{zeros}(A_1(:,j)) + \mathsf{zeros}(A_2(:,j)), \end{aligned}$$

where obviously $\mathsf{match}(A_1(:,j),0) = \mathsf{zeros}(A_1(:,j))$. Thus the choice of $\mathsf{mode}(A_2(:,j))$ in Theorem 6.2.2 is again a majority rule between $\mathsf{match}(A_2(:,j),\mathbf{1})$ and $\mathsf{zeros}(A_2(:,j))$. □

We claim that our method VOTE generalizes the existent code Proximus [111]. In particular, when applied to binary data, our selection mechanism is theoretically equivalent to the code PROXIMUS. To see the equivalence, recall that the basic rank-one approximation in PROXIMUS is based on the idea of minimizing the Euclidean distance $\|A - \mathbf{u}\mathbf{v}\|_F^2$. Upon rewriting

$$\|A - \mathbf{u}\mathbf{v}\|_F^2 = \|A\|_F^2 - 2\mathbf{u}^\top A\mathbf{v}^\top + \|\mathbf{u}\|_2^2\|\mathbf{v}\|_2^2,$$

we see that the functional

$$f(\mathbf{u},\mathbf{v}) = 2\mathbf{u}^\top A\mathbf{v}^\top - \|\mathbf{u}\|_2^2\|\mathbf{v}\|_2^2 \tag{6.5}$$

needs to be maximized. The rule adopted by Proximus is that, given a binary vector $\mathbf{v}$, the optimal entries $\mathbf{u}$ must be defined by

$$u_i := \begin{cases} 1, & \text{if } \ 2(A\mathbf{v}^\top)_i - \|\mathbf{v}\|_2^2 \geq 0, \\ 0, & \text{otherwise .} \end{cases} \tag{6.6}$$

Similarly, given a binary vector $\mathbf{u}$, the optimal entries $\mathbf{v}$ is given by

$$v_i := \begin{cases} 1, & \text{if } \ 2(\mathbf{u}^\top A)_i - \|\mathbf{u}\|_2^2 \geq 0, \\ 0, & \text{otherwise .} \end{cases} \tag{6.7}$$

Observe that

$$\begin{aligned} &\mathsf{match}(A(i,:),\mathbf{v}) - \mathsf{zeros}(A(i,:)) \\ &= \{n - (\mathbf{v} - A(i,:))(\mathbf{v} - A(i,:))^\top\} - \{n - A(i,:)A(i,:)^\top\} \\ &= -\mathbf{v}\mathbf{v}^\top + 2A(i,:)\mathbf{v}^\top \\ &= 2(A\mathbf{v}^\top)_i - \|\mathbf{v}\|_2^2, \end{aligned} \tag{6.8}$$

and

$$\begin{aligned} &\mathsf{match}(A(:,i),\mathbf{u}) - \mathsf{zeros}(A(:,i)) \\ &= \{m - (\mathbf{u} - A(:,i))^\top(\mathbf{u} - A(:,i))\} - \{m - A(:,i)^\top A(:,i)\} \\ &= -\mathbf{u}^\top\mathbf{u} + 2\mathbf{u}^\top A(i,:) \\ &= 2(\mathbf{u}^\top A)_i - \|\mathbf{u}\|_2^2. \end{aligned} \tag{6.9}$$

Formulas (6.8) and (6.9) justify that VOTE and PROXIMUS are employing exactly the same procedure at each step in rank-one approximation when dealing with binary data. Note, however,

that our method is computationally cheaper than PROXIMUS because VOTE avoids matrix-vector multiplications needed in (6.6) and (6.7). Furthermore, our approach VOTE can handle more general polychotomous data.

### 6.2.2 Approximation with Euclidean metric

In many applications, it is essential to differentiate the true discrepancies among variable values. That is, the values that a variable takes signifies levels of priority, weight, or worth. If the data are somehow represented in the Euclidean space, then the real distance between two points is meaningful and makes a difference in the interpretation. Under such a setting, we discuss in this section the rank-one approximation when the Euclidean metric is used as the measurement for nearness.

Given $\mathbf{v}$, even if it is not binary, the definition (6.6) for $\mathbf{u}$ remains to be the optimizer. For completion, we restate the following theorem, but provide a slightly different proof.

**Theorem 6.2.3** *Given $A \in \mathbb{Z}^{m\times n}$ and $\mathbf{v} \in \mathbb{Z}^{1\times n}$, then among all $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$, the minimal value of $\|A - \mathbf{uv}\|_F^2$ is attained at $\mathbf{u}^*$ whose ith entry is defined by*

$$u_i = \begin{cases} 1, & \text{if } 2(A\mathbf{v}^\top)_i - \|\mathbf{v}\|_2^2 \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{6.10}$$

**Proof.** Since

$$\|A - \mathbf{uv}\|_F^2 = \sum_{i=1}^{m} \|A(i,:) - u_i\mathbf{v}\|_2^2,$$

it is clear that in order to minimize each individual term in the summation we should choose

$$u_i := \begin{cases} 1, & \text{if } \|A(i,:)\|_2^2 \geq \|A(i,:) - \mathbf{v}\|_2^2, \\ 0, & \text{otherwise ,} \end{cases}$$

which is equivalent to (6.10). □

Let $\mathsf{round}(\mathbf{x})$ denote the operator that rounds every entry of $\mathbf{x}$ to its nearest integer. Given $\mathbf{u}$, the next result nicely generalizes the selection criterion in PROXIMUS for general integer vector $\mathbf{v}$ under the Euclidean metric.

**Theorem 6.2.4** *Given $A \in \mathbb{Z}^{m\times n}$ and $\mathbf{u} \in \mathbb{Z}_2^{m\times 1}$ with $\mathbf{u} \neq 0$, then among all $\mathbf{v} \in \mathbb{Z}^{1\times n}$, the minimal value of $\|A - \mathbf{uv}\|_F^2$ is attained at $\mathbf{v}^*$ defined by*

$$\mathbf{v}^* := \mathit{round}\left(\frac{\mathbf{u}^\top A}{\|\mathbf{u}\|_2}\right). \tag{6.11}$$

**Proof.** We rewrite (6.5) as

$$2\mathbf{u}^\top A\mathbf{v}^\top - \|\mathbf{u}\|_2^2\|\mathbf{v}\|_2^2 = \sum_i \left[ -\|\mathbf{u}\|_2^2 \left( v_i - \frac{(\mathbf{u}^\top A)_i}{\|\mathbf{u}\|_2} \right)^2 + \left( \frac{(\mathbf{u}^\top A)_i}{\|\mathbf{u}\|_2} \right)^2 \right]. \tag{6.12}$$

It is now clear that the only option for $\mathbf{v}$ to minimize $\|A - \mathbf{uv}\|_F^2$ while keeping $\mathbf{v}$ an integer vector is the definition (6.11). □

Since it takes only a slight modification in the code VOTE to reflect the Euclidean metric, we shall skip the particulars of the coding here. Be cautious though that if the data range is restricted to only a subset of integers rather than $\mathbb{Z}$, the operator round might return a value outside $\mathbb{Z}$. In this case, we can use (6.12) to define a proper value $v_i \in \mathbb{Z}$ that is nearest to $\frac{(\mathbf{u}^\top A)_i}{\|\mathbf{u}\|_2}$.

We conclude this section with two remarks. Firstly, in the event that $\mathbf{u} = 0$, we can simply assign $\mathbf{v}$ arbitrarily, say, $\mathbf{v} = \mathbf{1}$, without affecting the iteration. Secondly and most importantly, the reason we insist on restricting $\mathbf{u}$ to $\mathbb{Z}_2^{m\times 1}$ is only for the purpose of establishing mutually exclusive clusters, as we shall see in the next section. As a consequence, the resulting matrix $U$ automatically has mutually orthgonal columns. Our current emphasis is on the binary-integer matrix factorization, but ideas in the proof of Theorem 6.2.4 certainly can be generalized to $\mathbf{u}$ if $\mathbf{u}$ is to be an integer vector. In this event, the optimal integer vector $\mathbf{u}$ for minimizing $\|A - \mathbf{uv}\|_F^2$ with a fixed $\mathbf{v}$ is given by

$$\mathbf{u}^* := \mathsf{round}\left( \frac{A\mathbf{v}^\top}{\|\mathbf{v}\|_2} \right). \tag{6.13}$$

Continuing this way, we construct a sequence of integer rank-one approximations to $A$ and obtain a factorization with integer entries.

## 6.3 Algorithms of integer matrix factorization

We now describe the procedure for constructing the binary-integer factorization of a given integer matrix $A$. The main idea is to assemble $A$ via a sequence of rank-one approximations, but the superposition is not done in the traditional sense of *additive* manner as we shall explain in this section. Three essential issues must be resolved before we can move this idea forward. These are — how to assess the quality of an approximation, how to recursively deflating the matrix, and how to determine the optimal rank. We address each issue separately in the sequel.

For the convenience of reference, after a rank-one approximation $\mathbf{uv}$ for $A$ is established, we say collectively that those rows $A(i,:)$ of $A$ corresponding to $u_i = 1$ are *active* and call the corresponding $\mathbf{v}$ the *pattern vector* or *representative* of these active rows.

### 6.3.1 Communal rule

Algorithm 6 guarantees the undertaking of the "global" minimizer at each iteration for each fixed vector $\mathbf{u}$ or $\mathbf{v}$. The outcome from the one-sided minimization does not necessarily give rise to the global minimizer for the two-sided objective function. The quality of a rank-one approximation $A \approx \mathbf{uv}$ depends highly on the initial value. This dependence limits the ADI to find only local interpretable patterns. It is therefore reasonable to set up a checking criterion to decide whether active rows are adequately represented by the pattern vector $\mathbf{v}$.

So far the assignment of $u_i$ for each $i = 1, \ldots, m$ is solely on a comparison of $A(i,:)$ individually with $\mathbf{v}$. Since the differences between the pattern vector and corresponding active rows of $A$ might vary immensely, merely qualifying $A(i,:)$ into the active group by self-justification has the danger of dissimilarity among active rows. To remedy this situation, we must provide some criteria to ensure the homogeneity within the collective of all active rows. Since $\mathbf{v}$ is not necessarily the mean of all active rows, every single active row, though preliminarily counted as active, could be far away from $\mathbf{v}$. The conventional approach by fixing a neighborhood around $\mathbf{v}$ is not effective to exclude outliers. Instead, we propose to check the communality of the active rows more dynamically by using a simple statistical rule.

Firstly, let the vector $\mathbf{r}$ denote the collection of "distances" between active rows of $A$ and $\mathbf{v}$, that is,

$$r_j = d(A(i_j,:), \mathbf{v}), \tag{6.14}$$

whenever $u_{i_j} = 1$. Secondly, we calculate the mean $\mu$ and standard deviation $\sigma$ of $\mathbf{r}$. For a fixed $\beta > 0$, we adopt a communal rule that whenever

$$|r_j - \mu| > \beta\sigma, \tag{6.15}$$

the membership of $A(i_j,:)$ is rejected from the cluster by resetting $u_{i_j} = 0$. It is possible that all active rows will be rejected based on (6.15), leading to the so called *cluster death* and causing the algorithm to break down. To avoid such a situation, we keep $u_{i_\ell} = 1$ whenever $r_\ell = \min \mathbf{r}$.

The parameter $\beta$ in (6.15) serves as a threshold which can affect the ultimate partitions of data matrix. The larger the threshold $\beta$ is, the more inclusive the cluster become, and the less the number of partitions of $A$ will be. For data with large noises, for instance, increasing $\beta$ lead to fewer representatives for the rows of $A$ which, in turn, might rid of some of the unwanted substances in the data. On the other hand, if pairwise matching is more important than pairwise distance, we might decrease the $\beta$ value to increase the uniformity. In our algorithm, the user can adjust the parameter value $\beta$ according to the need.

**Algorithm 8**: Integer matrix factorization by VOTE: $[U, V] = \text{IMFVOTE}(A, active, \beta)$

**Input**:

| | | |
|---|---|---|
| $A$ | = | matrix in $\mathbb{Z}^{m \times n}$ to be decomposed |
| $active$ | = | array of indices identifying submatrices of $A$ being analyzed |
| $\beta$ | = | threshold for communal rule |

**Output**: Matrices $U \in \mathbb{Z}_2^{m \times p}$ and $V \in \mathbb{Z}^{p \times n}$ for some integer $p$ such that $A \approx UV$

8.1 **begin**
8.2 $A \leftarrow A(active, :)$;
8.3 $\mathbf{u}, \mathbf{v} \leftarrow \text{VOTE}(A)$;
8.4 % Check the communality;
8.5 $uactive \leftarrow find(\mathbf{u})$;
8.6 **foreach** *row* $A(i, :)$ *of* $A(uactive, :)$ **do**
8.7 **if** $A(i, :)$ *does not meet communal rule* **then**
8.8 $u_{uactive_i} \leftarrow 0$;
8.9 **end**
8.10 **end**
8.11 % Keep decomposing matrix;
8.12 $ZeroCheck \leftarrow find(\mathbf{u} = 0)$;
8.13 **if** $ZeroCheck$ *is not empty* **then**
8.14 $active1 \leftarrow active(find(\mathbf{u} = 1))$;
8.15 **if** $actove1$ *is not empty* **then**
8.16 $\text{IMFVOTE}(A, active1, \beta)$;
8.17 **end**
8.18 $active0 \leftarrow active(ZeroCheck)$;
8.19 $\text{IMFVOTE}(A, active0, \beta)$;
8.20 **else**
8.21 Augment $\mathbf{u}$ and $\mathbf{v}$ in $U$ and $V$ as a column and a row, respectively;
8.22 **end**
8.23 **end**

### 6.3.2 Recursive decomposition

The most important objective of the rank-one approximation is to separate rows of $A$ into two mutually exclusive clusters according to innate attributes of each row while, in the meantime, trying to find a representative for the active rows. The motive of a filtering procedure by the

communal rule is to further refine the active rows to form a tighter cluster. These objectives are repeatedly checked through the following divide-and-conquer procedure:

Step 1. Given a matrix $A$, assess a possible cluster $\mathbf{u}$ of active rows and its representative $\mathbf{v}$.

Step 2. Based on available $\mathbf{u}$, deflate the matrix $A$ into two submatrices composed of all active and inactive rows, respectively.

Step 3. Recursively apply Step 1 to each submatrix in Step 2 until no more splitting is possible or a predesignated number of iteration is reached.

Step 4. Whenever an action in Step 3 is terminated, record the corresponding $\mathbf{u}$ and $\mathbf{v}$ as an extra column and row in the matrix $U$ and $V$, respectively.

Algorithm 8 sketches how a simple integer factorization $A \approx UV$ can be executed through VOTE *without controlling the size* $p$ in the final output matrices $U \in \mathbb{Z}_2^{m\times p}$ and $V \in \mathbb{Z}^{p\times n}$ [1]. Strictly speaking, this is not the IMF we have defined in (1.4) because the ultimate $p$ obtained by Algorithm 8 could be exceedingly large. But, it does represent an exhaustive search for factors. It is possible that in the extreme case that each cluster contains only one member, that is, we have $U = I$ and $V = A$, which of course is of little interest.

A few remarks about Algorithm 8 are worth noting. Observe that the splitting of $A$ according to $\mathbf{u}$ automatically guarantees that the resulting matrix $U$ has mutually orthogonal columns, which also means that each row of $A$ is assigned to one and only one group. Observe also that the code IMFVOTE invokes itself recursively. Such a feature, allowable in most modern programming languages, makes the code particularly efficient. Finally, be aware of the selection mechanism embedded in the code that determines the final membership $\mathbf{u}$ and representative $\mathbf{v}$ through multiple levels of screening.

### 6.3.3 Optimal low rank approximation

In the application of low rank approximations, one of the most challenging issues even to this date is the predetermination of the low rank $k$. Algorithm 8 calculates an approximation $A \approx UV$ without any restriction on the sizes of $U \in \mathbb{Z}_2^{m\times p}$ and $V \in \mathbb{Z}^{p\times n}$. Suppose that such a factorization is now at hand. It is natural to ask whether there is a way to pick up two submatrices $U_k \in \mathbb{Z}_2^{m\times k}$ and $V_k \in \mathbb{Z}^{k\times n}$ from $U$ and $V$ such that $A \approx U_k V_k$ and $k < p$. In this section, we make an interesting observation on how to choose the best pair of submatrices $(U_k, V_k)$ so that $d(A, U_k V_k)$ is minimal among *all* possible submatrices of compatible sizes.

[1] A properly selected low rank $p$ is critically important in practice, but no theory about how this could be done is available. In most case, the choice of $p$ is done on an ad hoc basis. See Section 6.3.3 for our new theory in this regard.

Although this is a postscript to the main computation already done by Algorithm 8, it sheds a remarkable insight into the optimal low rank approximation.

Denote the columns and rows of $U$ and $V$ by

$$U = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_p], \quad V = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_p \end{bmatrix},$$

respectively. Note that these rank-one approximations $\mathbf{u}_i\mathbf{v}_i$ are found, say, by Algorithm 8, successively without any specific ordering. For $\ell = 1, \ldots, p$, define $\mathcal{S}_\ell$ to be the collection

$$\mathcal{S}_\ell := \left\{ (i_1, \ldots, i_\ell) \in \mathbb{Z}^\ell \,\middle|\, d\left(A, U_{i_1,\ldots,i_p} V_{i_1,\ldots,i_p}\right) = \min_{(j_1,\ldots,j_\ell)} d\left(A, U_{j_1,\ldots,j_p} V_{j_1,\ldots,j_p}\right) \right\}, \tag{6.16}$$

where

$$U_{i_1,\ldots,i_p} = \left[\mathbf{u}_{i_1}, \mathbf{u}_{i_2}, \cdots, \mathbf{u}_{i_p}\right], \quad V_{i_1,\ldots,i_p} = \begin{bmatrix} \mathbf{v}_{i_1} \\ \mathbf{v}_{i_2} \\ \vdots \\ \mathbf{v}_{i_p} \end{bmatrix},$$

for some indices $(i_1, \ldots, i_p)$ and $d$ is either the Hamming or the Euclidean metric and the $\ell$-tuple $(j_1, \ldots, j_\ell)$ is made of distinct indices from $\{1, \ldots, p\}$. The folliwng nesting effect among $\mathcal{S}_\ell$'s is rather surprising and of potential significance.

**Theorem 6.3.1** *Suppose that the matrix $A \in \mathbb{Z}^{m\times n}$ has been factorized into $A \approx UV$ with $U \in \mathbb{Z}_2^{m\times p}$ and $V \in \mathbb{Z}^{p\times n}$ by Algorithm 8. Then every element in $\mathcal{S}_s$ must appear as a segment in some element of $\mathcal{S}_t$, if $s < t$.*

**Proof.** It suffices to prove the assertion for the case $s = 1$ and $t = 2$. The following argument can be generalized to other cases. Suppose that there exists an integer $i_1 \in \mathcal{S}_1$ but $\{i_1, i_2\} \notin \mathcal{S}_2$, for any $i_2 \in \{1, 2, \cdots, p\}$. Given any $\{j_1, j_2\} \in \mathcal{S}_2$, we have

$$d\left(A, [\mathbf{u}_{j_1}, \mathbf{u}_{j_2}] \begin{bmatrix} \mathbf{v}_{j_1} \\ \mathbf{v}_{j_2} \end{bmatrix}\right) < d\left(A, [\mathbf{u}_{i_1}, \mathbf{u}_{j_2}] \begin{bmatrix} \mathbf{v}_{i_1} \\ \mathbf{v}_{j_2} \end{bmatrix}\right). \tag{6.17}$$

We want to prove that a contradiction arises.

Note that the numeral "1" appears at mutually disjoint positions within the vectors $\mathbf{u}_{i_1}$, $\mathbf{u}_{j_1}$ and $\mathbf{u}_{j_2}$. Simultaneously permuting rows if necessary, we may assume without loss of generality

that rows of $A$ have been divided into four blocks

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} \tag{6.18}$$

where rows of $A_1$, $A_2$ and $A_3$ correspond to 1's in $\mathbf{u}_{i_1}$, $\mathbf{u}_{j_1}$ and $\mathbf{u}_{j_2}$, respectively, and $A_4$ corresponds to the common zeros of all $\mathbf{u}_{i_1}$, $\mathbf{u}_{j_1}$ and $\mathbf{u}_{j_2}$. Then it becomes clear that

$$\begin{aligned} d(A, \mathbf{u}_{i_1}\mathbf{v}_{i_1} + \mathbf{u}_{j_2}\mathbf{v}_{j_2}) &= d(A_1, \mathbf{v}_{i_1}) + d(A_2, 0) + d(A_3, \mathbf{v}_{j_2}) + d(A_4, 0), \\ d(A, \mathbf{u}_{j_1}\mathbf{v}_{j_1} + \mathbf{u}_{j_2}\mathbf{v}_{j_2}) &= d(A_1, 0) + d(A_2, \mathbf{v}_{j_1}) + d(A_3, \mathbf{v}_{j_2}) + d(A_4, 0), \\ d(A, \mathbf{u}_{i_1}\mathbf{v}_{i_1}) &= d(A_1, \mathbf{v}_{i_1}) + d(A_2, 0) + d(A_3, 0) + d(A_4, 0), \\ d(A, \mathbf{u}_{j_1}\mathbf{v}_{j_1}) &= d(A_1, 0) + d(A_2, \mathbf{v}_{j_1}) + d(A_3, 0) + d(A_4, 0). \end{aligned}$$

Upon substitution, it follows from (6.17) that

$$d(A, \mathbf{u}_{j_1}\mathbf{v}_{j_1}) < d(A, \mathbf{u}_{i_1}\mathbf{v}_{i_1}),$$

which contradicts with respect to the assumption that $i_1 \in \mathcal{S}_1$. □

The preceding observation has an important consequence in that it enables us to assess and gauge the quality of low rank approximations of $A$. The application is characterized in the following theorem.

**Theorem 6.3.2** *Suppose that the matrix $A \in \mathbb{Z}^{m\times n}$ has been factorized into $A \approx UV$ with $U \in \mathbb{Z}_2^{m\times p}$ and $V \in \mathbb{Z}^{p\times n}$ by Algorithm 8. Sort through the rank-one approximations and rearrange the rows if necessary, assume that*

$$d(A, \mathbf{u}_1\mathbf{v}_1) \le d(A, \mathbf{u}_2\mathbf{v}_2) \le \cdots \le d(A, \mathbf{u}_p\mathbf{v}_p). \tag{6.19}$$

*Then, for $k = 1, 2, \cdots, p$, the product $U_kV_k$ where $U_k$ and $V_k$ are submatrices of $U$ and $V$ given by*

$$U_k := [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_k], \quad V_k = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_k \end{bmatrix}, \tag{6.20}$$

*is the best possible approximation to $A$ in the sense that the $k$-tuple $(1, 2, \cdots, k)$ is in $\mathcal{S}_k$.*

**Proof.** We prove the assertion by induction on $k$.

Obviously, the case $k = 1$ is already done due to the rearrangement specified in (6.19). Assume therefore that $(1, 2, \cdots, k-1) \in \mathcal{S}_{k-1}$. We want to show that $(1, 2, \cdots, k) \in \mathcal{S}_k$. The following argument is essentially parallel to that in Theorem 6.3.1, except that we work on blocks.

By Theorem 6.3.1, there exists an integer $q \in \{k, k+1, \cdots, p\}$ such that the $k$-tuple $(1, 2, \cdots, k-1, q)$ is an element in $\mathcal{S}_k$. If $q = k$, then we are done. Assume, by contradiction, that $q \neq k$. Consider the pair of submatrices

$$\tilde{U}_k := [U_{k-1}, \mathbf{u}_q], \quad \tilde{V}_k := \begin{bmatrix} V_{k-1} \\ \mathbf{v}_q \end{bmatrix}.$$

Still, columns of $\tilde{U}_k$ are mutually exclusive. Because $(1, 2, \cdots, k-1, k) \notin \mathcal{S}_k$, we know

$$d(A, \tilde{U}_k \tilde{V}_k) < d(A, U_k V_k),$$

which is equivalent to

$$d(A, U_{k-1}V_{k-1} + \mathbf{u}_q \mathbf{v}_q) < d(A, U_{k-1}V_{k-1} + \mathbf{u}_k \mathbf{v}_k). \tag{6.21}$$

Again, without loss of generality, we may partition $A$ into blocks such as that in (6.18) where rows of $A_1$, $A_2$ and $A_3$ correspond to 1's in $U_{k-1}$, $\mathbf{u}_q$ and $\mathbf{u}_k$, respectively, and $A_4$ corresponds to the common zeros of all $U_{k-1}$, $\mathbf{u}_q$ and $\mathbf{u}_k$. Then, clearly we have the following expressions:

$$\begin{aligned}
d(A, U_{k-1}V_{k-1} + \mathbf{u}_q \mathbf{v}_q) &= d(A_1, V_{k-1}) + d(A_2, \mathbf{v}_q) + d(A_3, 0) + d(A_4, 0), \\
d(A, U_{k-1}V_{k-1} + \mathbf{u}_k \mathbf{v}_k) &= d(A_1, V_{k-1}) + d(A_2, 0) + d(A_3, \mathbf{v}_k) + d(A_4, 0), \\
d(A, \mathbf{u}_q \mathbf{v}_q) &= d(A_1, 0) + d(A_2, \mathbf{v}_q) + d(A_3, 0) + d(A_4, 0), \\
d(A, \mathbf{u}_k \mathbf{v}_k) &= d(A_1, 0) + d(A_2, 0) + d(A_3, \mathbf{v}_k) + d(A_4, 0).
\end{aligned}$$

It follows from (6.21) that

$$d(A, \mathbf{u}_q \mathbf{v}_q) < d(A, \mathbf{u}_k \mathbf{v}_k),$$

which contradicts the assumption that

$$d(A, \mathbf{u}_q \mathbf{v}_q) \geq d(A, \mathbf{u}_k \mathbf{v}_k),$$

for $q \in \{k+1, \cdots, p\}$. □

Even though we still do not know how to find the overall optimal rank, the above discussion

**Algorithm 9**: Low_Rank_Optimization: $[U, V] = \text{LowRankApprox}(A, k)$

**Input**: Matrix $A \in \mathbb{Z}_2^{m\times n}$ and integer $k$
**Output**: Low rank factors $U \in \mathbb{Z}_2^{m\times k}$ and $V \in \mathbb{Z}^{k\times n}$

```
9.1 begin
9.2     U, V ← IMFvote(A, active, β);
9.3     forall i from 1 to p do
9.4         r ← [r; match(A, U(:, i)V(i, :))];
9.5     end
9.6     index_rankings ← sort(r);
9.7     U ← U(:, index_rankings(1 : k));
9.8     V ← V(index_rankings(1 : k), :);
9.9 end
```

suggests that something meaningful can be done after we have "completely" factorize $A$ as $A \approx UV$. That is, by ranking the rank-one approximations as in (6.19), we can build some lower rank approximations in a controlled way. This byproduct is interesting enough that we summarize it in Algorithm 9.

## 6.4 Numerical experiments

In this section, we demonstrate some interesting applications of our IMF techniques. Our test data are collected from requisitions in association analysis, cluster analysis, latent pattern discover, and random simulation. Some of the data are too large to be listed in this presentation. In that case, we give references of the sources where the data can be downloaded.

### 6.4.1 Association analysis

Many supermarkets, as well as companies that provide online content based on popular Web searches, keep detailed records of their daily transactions not so much for the sake of financial bookkeeping but more so for the hidden information about consumers' shopping behavior. In the interest of generating more profits, the merchants try to predict customers' behavior by mining some association rules and adjust their inventories accordingly. Demonstrated in Table 6.1 is a much simplified example of the so called *market basket transactions* which typically is a huge database with many customers and available items. Like the discovery investigated in [110, 111], we can convert the transactions into a binary matrix by associating each row with one transaction and each column with one item. See Table 6.2. The value of an entry in the matrix is 1 if the item is included in the transaction and 0 otherwise. Upon applying the IMF to

Table 6.1: An transaction example with 5 transactions and 6 items

| | |
|---|---|
| $T_1$ | {Milk, Diapers, Eggs} |
| $T_2$ | {Milk, Diapers, Eggs, Beer} |
| $T_3$ | {Diapers, Eggs, Beer} |
| $T_4$ | {Bread, Chips, Beer} |
| $T_5$ | {Bread, Milk, Chips} |

Table 6.2: A binary representation of the transaction example

| | Bread | Milk | Diapers | Eggs | Chips | Beer |
|---|---|---|---|---|---|---|
| $T_1$ | 0 | 1 | 1 | 1 | 0 | 0 |
| $T_2$ | 0 | 1 | 1 | 1 | 0 | 1 |
| $T_3$ | 0 | 0 | 1 | 1 | 0 | 1 |
| $T_4$ | 1 | 0 | 0 | 0 | 1 | 1 |
| $T_5$ | 1 | 1 | 0 | 0 | 1 | 0 |

the binary data in Table 6.2, we can approximate the original binary data by two representatives $[0, 1, 1, 1, 0, 1]$ and $[1, 0, 0, 0, 1, 1]$ as follows

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \approx \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}. \tag{6.22}$$

The factorization (6.22) is a rank-two approximation, suggesting that the original five transactions probably can be simplified by two representative transactions $T_1'$ and $T_2'$ defined in Table 6.3 and that three out of five customers are likely to go with the first kind of buying pattern.

Table 6.3: Two representative transactions $T_1'$ and $T_2'$

| | |
|---|---|
| $T_1'$ | {Milk, Diapers, Eggs, Beer} |
| $T_2'$ | {Bread, Chips, Beer} |

In mining for association rules, it is known that the number of rules grows exponentially with the number of transactions [88]. Typically, the number of transactions are of very large scale. This example typifies the significance of IMF application because now we can concentrate analyzing the associate rules among the smaller approximate transaction set which may manifest the information more easily. The IMF application clearly alleviate the load of performing association analysis on large scale data. More importantly, the discovered patterns in the original data might include some spurious results related to certain casual events. The IMF serves a filter to screen out those casual or redundant events. Only a few representative patters are sufficient to predict the market demand. Through this factorization, not only are we able to characterize the main features in the original patterns, but we are also able to purify some dubious correlations among them.

### 6.4.2 Cluster analysis

The general purpose of cluster analysis is to partition a set of observations into subsets, called clusters, so that observations in the same cluster share some common features. The most difficult part in cluster analysis is the interpretation of the resulting clusters. It is important but not our primary purpose in this investigation to compare the performance of our method with the many cluster analysis algorithm already developed in the literature. Rather, we merely want to demonstrate that IMFVOTE naturally produces clusters and their corresponding integer representatives. If needed, we can also employ LOWRANKAPPROX to pick up the most relevant low rank representation of original data set, which we think is an additional attraction of our method.

We report experiments on two real data sets, mushroom and Wisconsin breast cancer (original), from the *Machine Learning Repository* maintained by the Computer Science Department at the University of California at Irvine [5]. To gauge the effectiveness of our algorithm, we employ four parameters, RowErrorRate, CompressionRatio, Precision, and Recall whose meanings are outline below [31].

RowErrorRate, defined by

$$\mathsf{RowErrorRate} := \frac{d(A, UV)}{m}, \tag{6.23}$$

refers to the average difference per row between the retrieved data $UV$ and the original data $A$. This value reflects how effective row vectors in $V$ are representing the original matrix $A$.

CompressionRatio [47], defined by

$$\begin{aligned} \mathsf{CompressionRatio} &:= \frac{\sharp \text{ of entries in matrices } U \text{ and } V}{\sharp \text{ of entries in matrices } A} \\ &= \frac{(m+n)k}{mn} \end{aligned} \tag{6.24}$$

measures how efficiently the original data $A$ has been compressed by the low rank representation $UV$. It is hoped that through the compression, less relevant data or redundant information is removed.

Precision and Recall, defined by

$$\mathsf{Precision} := \frac{\text{relevant data} \cap \text{retrieved data}}{\text{retrieved data}}, \tag{6.25}$$

$$\mathsf{Recall} := \frac{\text{relevant data} \cap \text{retrieved data}}{\text{relevant data}}, \tag{6.26}$$

computes the percentages of the retrieved data that are relevant and relevant data that are retrieved, respectively. In the context of information retrieval, "relevant data" usually refer to documents that are apposite to a specified inquiry. Since our task at present is simply to divide observations into disjoint clusters based on innate attributes, we do not know the exact meaning of each cluster. In other words, we do have at hand a representative for each cluster, but we do not have an interpretation of the representative. Still, considering points in a cluster as retrieved data relative to their own representative, it might be interesting to compare the corresponding Precision and Recall with some known training data. In this way, a high correlation between a particular cluster and the training data might suggest an interpretation for the cluster. We have not pursued this direction in this investigation

**Mushroom Data Set.** This Agaricus and Lepiota Family data set consists of biological for 8124 hypothetical species of gilled mushroom. Each species is characterized by 23 attributes such as its cap shape, cap surface, odor, and so on. Each attribute has different *nominal* values. For example, the cap shape may be belled, conical, convex, flat, knobbed, or sunken, while the cap surface may be fibrous, grooved, scaly, or smooth. To fit into our scheme, we convert the attribute information into a list of integer values. These integers should not be regarded as numerals, but only labels. Since attributes are independent of each other, the same integer for different attributes has different meanings. Our input data $A$ is an $8124 \times 23$ integer matrix. Of particularly noticeable is its first column which is dichotomous and indicates whether the mushroom is edible or poisonous. The Hamming metric is more suitable than the Euclidean metric for this problem.

In our first experiment, we simply want to investigate how different $\beta$ values affect the quantities of approximation. As is expected, a smaller $\beta$ value would lead to a larger number $p$ of rows in the matrix $V$ after a complete factorization. For mushroom data set, we find

that corresponding to $\beta = 1, 2, 3$ the numbers of rows in $V$ are $p = 1418, 15, 1$, respectively. For each $\beta$, construct $V_k$ according to Theorem 6.3.2 after sorting the corresponding $V$. We plot in Figure 6.1 the value RowErrorRate versus $k$. Note the rapidly decreasing behavior, especially in the cases $\beta = 1, 2$, suggesting that the sorting in LowRankApprox is capable of identifying the first few most important clusters and their representatives. A comparison with the CompressionRatio$\approx 0.0436k$ in this case is also worth noting. To achieve RowErrorRate$=$ 9.6128, we just need one pattern vector with $\beta = 3$, but we will need many more representatives with $\beta = 1$ or 2. On the other hand, with $\beta = 3$ we cannot possibly improve the RowErrorRate, but with more restrictive $\beta$ values there is a chance to improve the RowErrorRate.

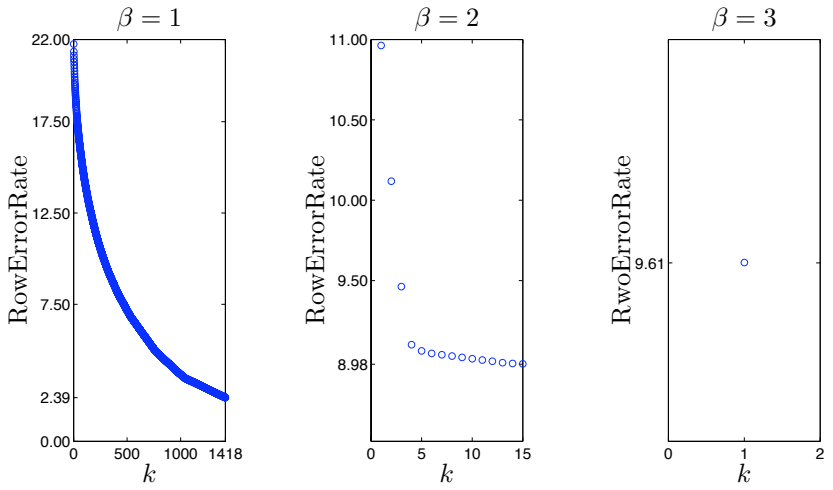


Figure 6.1: Performance of algorithms on the mushroom data set

In our second experiment, we apply LowRankApprox to the $8124 \times 22$ submatrix after removing the first column from $A$. Rather surprisingly, by deleting merely one column of $A$, the resulting IMF behaves very differently. With $\beta = 2$, for example, the complete factorization returns only three clusters with RowErrorRate$=$ 9.4623. In an attempt to provide some meaning to these clusters, we construct the so called *confusion matrix* in Table 6.4 with respect to the attribute of whether the mushroom is edible or poisonous. As is seen, the first cluster represented by $\mathbf{v}_1$ contains large amounts of mushrooms in both kinds, resulting in relative high Recall rates. But the Precision values by $\mathbf{v}_1$ are about the same as the distribution rates in the original data, indicating that the cluster by $\mathbf{v}_1$ does not differentiate edible mushrooms from poisonous ones. The exact meaning of these clusters is yet to be understood.

**Wisconsin Breast Cancer Data Set.** In a similar way, we experiment our method

Table 6.4: Precision and Recall of edible or poisonous mushrooms

| | Retrieved data set | | |
|---|---|---|---|
| | $\mathbf{v}_1$ | $\mathbf{v}_2$ | $\mathbf{v}_3$ |
| Edible | 4128 | 80 | 0 |
| Poisonous | 3789 | 119 | 8 |
| Cardinality of $\mathbf{v}_i$ | 7917 | 199 | 8 |
| Precision of edible mushrooms | 0.5214 | 0.4020 | 0 |
| Recall of edible mushrooms | 0.9810 | 0.0190 | 0 |
| Precision of poisonous mushrooms | 0.4786 | 0.5980 | 1 |
| Recall of poisonous mushrooms | 0.9676 | 0.0304 | 0.0020 |

the Breast Cancer Wisconsin Data. This data set $A$ contains 699 samples, with 458 (65.5%) benign and 241 (34.5%) malignant cases. Each sample is characterized by 11 attributes. The first attribute is dichotomous with labels 2 or 4, indicating benign and malignant tumors. The remaining ten attributes are of integer values ranging from 1 to 10, but some entries in this part of $A$ have missing values for which we assign the value 0 in our computation. It is commonly known that aspects such as the thickness of clumps or the uniformity of cell shape affect the prognosis. Thus the ordinal of values in attributes matters. It is more appropriate to use the Euclidean metric.

Again, we try out three different communal rules by varying $\beta$. With CompressionRatio$\approx 0.1125k$, meaning a reduction of 11% memory space per one less cluster, we plot RowErrorRate versus $k$ in Figure 6.2. It is seen for this breast cancer data that increasing $k$ would improve RowErrorRate only modestly. Since the RowErrorRate is already low to begin with, a low rank IMF should serve well in approximating the original $A$.

We then apply LowRankApprox with $\beta = 3$ to the $699 \times 10$ submatrix by removing the first column of $A$. A complete factorization returns 11 pattern vectors with most samples being included in the first two clusters. We construct the confusion matrix with respect to the first attribute of $A$ in Table 6.5. Judging from the Recall and Precision values, we have high level of confidence that the two patterns $\mathbf{v}_1$ and $\mathbf{v}_2$ should be good indicators of whether the tumor is malignant or benigh, respectively. Such a classification of the original 699 samples into 2 major indicators will be extremely useful in medical science.

### 6.4.3 Pattern discovery

Most modern image processing techniques treats monochromatic images as a two-dimensional array composed of pixels. A digital image is an instruction of how to color each pixel. In a gray scale image, for example, the instruction for every element is an integer between 0 and

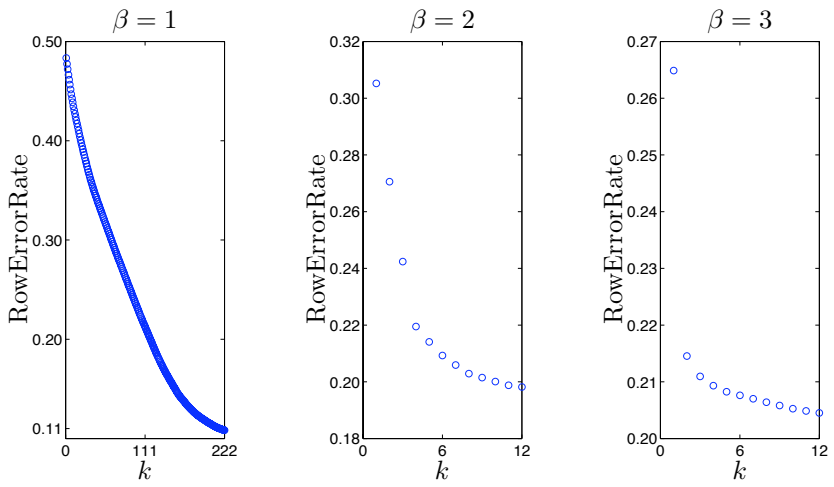


Figure 6.2: Performance of algorithms on Wisconsin breast cancer data set

255 (or a nonnegative decimal number between $[0, 1]$ which requires more storage and often is converted to integers) indicating the intensity of brightness at the corresponding pixel. In this section, let $A \in \mathbf{Z}^{m\times n}$ denote a collection of $n$ images each of which is represented by a column of $m$ pixels. Consider the scenario that images in this library are composite objects of many basic parts which are latent at present. The factorization $A = UV$ then might be suggested as a way to identify and classify those "intrinsic" parts that make up the object being imaged by multiple observations. More specifically, columns of $U$ are the basis elements while each row of $V$ can be thought of as an identification sequence representing the corresponding image in $A$ in terms of the basis elements. This idea has been extensively exploited by NMF techniques. See the many references mentioned earlier. The point to make, nonetheless, is that the NMF techniques cannot guarantee the resulting images to have integer-valued pixels whereas our IMF can.

Needless to say, the same idea can be applied to an extended field of applications, such as the quantitative structure-activity relationship (QSAR) discovery in chemoinformatics. Due to space limitation, we shall illustrate the pattern discovery ability of IMF by using two special data sets.

**Swimmer Data Set.** The "Swimmer" data set as described in Example 5.5.4 contains 17 basic parts that make up these swimmers. Taking everything into account, we should also expect one additional part for the background. Such an expectation of $U \in \mathbf{Z}_2^{1024\times 18}$ appears to be problematic because the original matrix $A$ has numerical rank of only 13. In this context, the notion of "low" rank approximation to $A$ really is not appropriate.

After using the Hamming metric and $\beta = 0.5$ in our code IMFVote to carry out a complete

Table 6.5: Precision and Recall of benign or recall patients

| | Retrieved data set | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\mathbf{v}_1$ | $\mathbf{v}_2$ | $\mathbf{v}_3$ | $\mathbf{v}_4$ | $\mathbf{v}_5$ | $\mathbf{v}_6$ | $\mathbf{v}_7$ | $\mathbf{v}_8$ | $\mathbf{v}_9$ | $\mathbf{v}_{10}$ | $\mathbf{v}_{11}$ |
| Benign | 33 | 411 | 1 | 5 | 3 | 0 | 1 | 1 | 1 | 1 | 1 |
| Malignant | 225 | 13 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Cardinality of $\mathbf{v}_i$ | 258 | 424 | 1 | 5 | 4 | 1 | 2 | 1 | 1 | 1 | 1 |
| Precision of benign cancer | 0.1279 | 0.9693 | 1 | 1 | 0.75 | 0 | 0.5 | 1 | 1 | 1 | 1 |
| Recall of benign cancer | 0.0721 | 0.8974 | 0.0022 | 0.0109 | 0.0066 | 0 | 0.0022 | 0.0022 | 0.0022 | 0.0022 | 0.0022 |
| Precision of malignant cancer | 0.8721 | 0.0307 | 0 | 0 | 0.25 | 1 | 0.5 | 0 | 0 | 0 | 0 |
| Recall of malignant cancer | 0.9336 | 0.0539 | 0 | 0 | 0.0041 | 0.0041 | 0.0041 | 0 | 0 | 0 | 0 |

factorization, we reshape the 18 columns of the resulting binary $U$ into $32 \times 32$ matrices. We recover all 16 limbs, one torso, plus the background as depicted in Figure 6.3. Note that these 17 recovered "body" parts are completely disjointed from each other and suffer from no blurring at all — a result that cannot be accomplished by NMF techniques. In fact, the factors $U$ and $V$ returned from our IMF satisfy $A = UV$ exactly.

**Block Matrix Data Set.** To demonstrate that our method can recognize patterns more complicated than one-dimensional sticks, consider a $5 \times 5$ block matrix with each block of size $5 \times 5$. Randomly select 2 out of the 25 blocks and assign the value 1 to their entries while keeping all other entries 0. Border this $25 \times 25$ matrix with 4 pixels on each side and call the resulting $33 \times 33$ matrix an image. Totally there are 300 images. Collect these images into the $1089 \times 300$ binary matrix $A$ with each column representing a vectorized $33 \times 33$ image. Apply IMFvote with Euclidean metric and $\beta = 1$ to $A$. The resulting complete factorization returns 27 clusters shown in Figure 6.4. It is interesting to note that the 25 basic patterns containing exactly one $5 \times 5$ block are completely discovered. Additionally, one pattern representing almost the entire border except the point on the upper left corner is found.

### 6.4.4 Random performance test

To further test the capability of the IMF in recovering random clusters or patterns, for a given triplet $(m, n, k)$ of integers we randomly generate $W \in \mathbb{Z}_2^{m \times k}$ and $H \in \mathbb{Z}_{14}^{k \times n}$, where columns of $W$ are kept mutually exclusive and $Z_{14} = \{0, 1, \ldots, 13\}$. Define $A = WH$ and apply IMFvote with Euclidean metric and $\beta = 1$ to $A$.

Let $(U, V)$ denote the pair of factors returned by our calculation. We wonder how likely $(U, V)$ would be the same as the original $(W, H)$ after some permutations. When this happens,

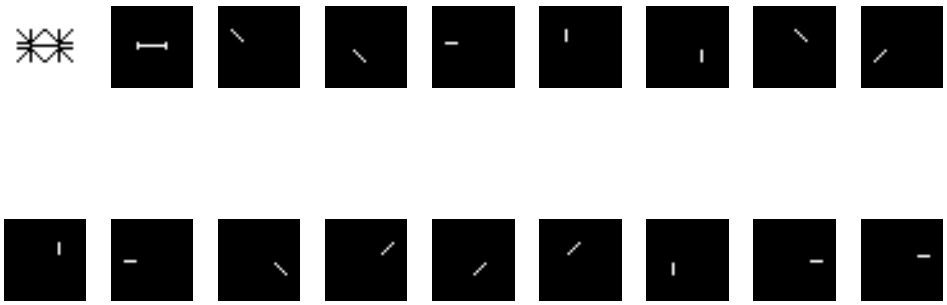

Figure 6.3: Basic elements recovered from the swimmer data set by the IMF.

we say that our method has reached its *optimal performance*. Recall that the basic rank one approximation in our scheme is only a local minimizer. We expect that pushing our algorithm, foredoomed just like most optimization techniques for nonlinear problems, to reach its optimal performance would be an extremely challenging task.

For each given $(m, n, k)$, we repeat the above-described experiment 1000 times and tally the rate of success, denoted by OptRate, in reaching the optimal performance. We also measure the CPU time needed for each experiment on a PC running Windows XP and Matlab R2009a with Intel(R)Core(TM)2 Duo CPU T8300@2.4GHz and 3.5GB of RAM. The average CPU time, denoted by AvgTime, then serves as an across-the-board reference for the computational overhead. Test results for a few selected triplets are summarized in Table 6.6. It seems possible to draw a few general rules from this table. For problems of the same size $(m, n)$, larger $k$ means more complexity and deeper recursion which, in turn, reduce OptRate and cost more AvgTime. For problems of the same $(m, k)$, increasing $n$ costs only AvgTime, but has modest effect on OptRate. For problem of the same $(n, k)$, increasing $m$ also costs only AvgTime and effects little on OptRate. The overall speed of our method seems reasonable, even though our code is yet to be further polished for efficiency by taking into account data structure, storage, and movement within the system.

Following from these results, we establish IMFvote as efficient tool for mining the inheritance patterns of given large datasets.

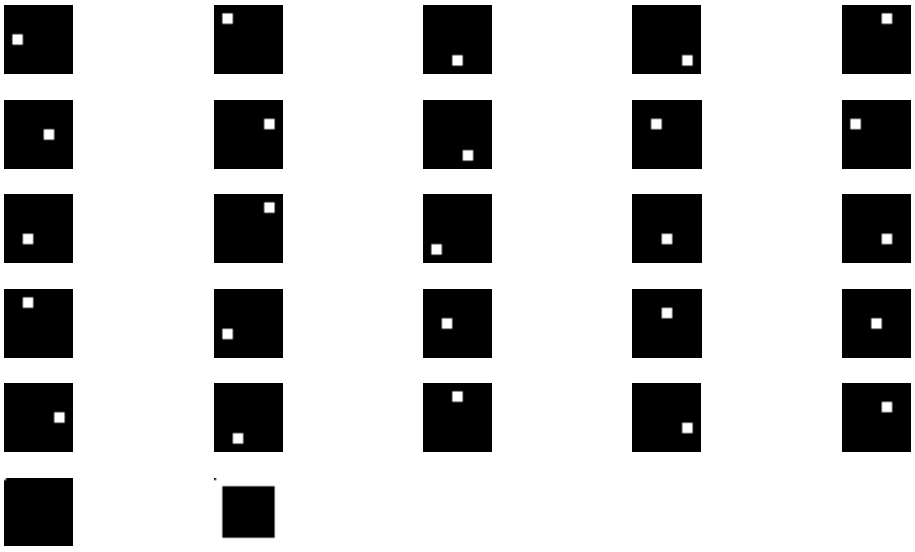

Figure 6.4: Bases elements recovered from block matrix data set by the IMF.

Table 6.6: Global convergence rate and average time per factorzation (in seconds) on randomly generated data sets.

| $m$ | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 500 | 1000 | 1000 | 1000 | 1000 | 1000 | 1000 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $n$ | 30 | 40 | 50 | 30 | 40 | 50 | 60 | 80 | 100 | 30 | 40 | 50 | 60 | 80 | 100 |
| $k$ | 5 | 6 | 7 | 10 | 12 | 14 | 5 | 6 | 7 | 10 | 12 | 14 | 5 | 6 | 7 |
| OptRate | .5530 | 0.3200 | .1820 | 0.0480 | .0110 | .0010 | .5280 | .3130 | .1450 | .0470 | .0100 | .0020 | .5440 | .3520 | .1740 |
| AvgTime | .0133 | .0158 | .0178 | .0167 | .0196 | .0224 | .0177 | .0213 | .0252 | .0468 | .0524 | .0575 | .0529 | .0627 | .0724 |

## 6.5 Conclusions

Matrix factorization has many important applications. In this chapter, we investigate the notion of factorization with entries restricted to integers or binaries. Being discrete in nature, such a factorization or approximation cannot be accomplished by conventional techniques. Built upon a basic scheme of rank one approximation, we propose an approach that recursively splits (or more correctly, approximates) the underlying matrix into a sum of rank one matrices with discrete entries. We carry out a systematic discussion to address the various computational issues involved in this kind of factorization. The ideas are implemented into an algorithm using either the Hamming or the Frobenius metric, but using other types of metrics is possible.

If the underlying data are binary, our idea is in line with the existing code PROXIMUS. But our formulation is readily generalizable to other types of data. For application purposes we have mainly concentrated on binary-integer factorizations, where the "integer" could be either the regular ordinal integers or just some nominal labels. With little effort, we can modify the mechanism to perform integer-integer factorizations.

Of particular interest is the result in Theorem 6.3.2 where we show how an optimal lower rank can be selected after a simple sorting. We think, in a remote sense, the ordering in (6.19) for discrete data is analogous to the ordering of singular values for continuous data. Similarly, the truncated product $U_k V_k$ defined in (6.20) is analogous to the truncated singular value decomposition.

Five different testing data are used to demonstrate the working of our IMF algorithm. We hope that our discussion in this investigation offers a unified and effectual avenue of attack on more general factorization problem. There is plenty of room for future research, including a refinement of our algorithm for more efficient data management and a generalization to more complex data types.

# Chapter 7

# Nonnegative Rank Factorization via Rank Reduction

## 7.1 Overview

The notion of "rank" for a matrix really should be dependent upon its ambient space, that is, the fields, rings, or semirings where its entries come from. A fundamental issue in dealing with the important class of nonnegative matrices is the difference between its algebraic rank over the real numbers and nonnegative rank over nonnegative number, which is main concern in this chapter.

Given any nonnegative matrix $A \in \mathbb{R}^{m\times n}$, it is always possible to express $A$ as the product $A = UV$ for some nonnegative matrices $U \in \mathbb{R}^{m\times k}$ and $V \in \mathbb{R}^{k\times n}$ with $k \leq \min\{m, n\}$. One trivial choice is $U = A$ and $V = I_n$. The smallest $k$ that makes such a factorization possible is called the *nonnegative rank* of A. For convenience, the nonnegative rank of $A$ is denoted by $\text{rank}_+(A)$. Note that the factorization $A = UV$ can always be expressed as the sum of a series of rank-one matrices

$$A = \sum_{i=1}^{k} \mathbf{u}_i \mathbf{v}_i^\top \tag{7.1}$$

where $\mathbf{u}_i$ and $\mathbf{v}_i^\top, i = 1, \cdots, k$, are column and row vectors of matrices $U$ and $V$, respectively. Some theoretical discussion about estimating the nonnegative rank of $A$ can be found in [44, 76]. In particular, we know

$$\text{rank}(A) \leq \text{rank}_+(A) \leq \min\{m, n\}. \tag{7.2}$$

The challenge has been to determine the exact nonnegative rank for a nonnegative matrix. It

is know that such an ascertainment is NP-hard [161].

Different from and more challenging than the conventional nonnegative matrix factorization, NMF, that we have studied in Chapter 5, the task we are facing is a complete factorization of matrix $A$ with its product equal to $A$. In this chapter, our focus is on a special subclass of nonnegative matrices,

$$\mathfrak{R}(m,n) := \left\{ A \in \mathbb{R}_+^{m\times n} \mid \operatorname{rank}(A) = \operatorname{rank}_+(A) \right\}. \tag{7.3}$$

Note that we have assumed that the nonnegative rank is known for $A \in \mathfrak{R}(m,n)$. Our goal is to determine two nonnegative matrices $U$ and $V$ of sizes $m \times \operatorname{rank}(A)$ and $\operatorname{rank}(A) \times n$, respectively, such that

$$A = UV.$$

We call this factorization a *nonnegative rank factorization* (NRF) of $A$. By representing the matrix $A$ as the sum of a series of nonnegative rank-one matrices, we notice a special feature in that the residual after subtracting each rank-one matrix from $A$ remains nonnegative but has rank one less than that of the original $A$.

Every nonnegative matrix has a nonnegative factorization. But, not every nonnegative matrix could be factorized in the form of NRF [44]. A few sufficient conditions for constructing nonnegative matrices *without* NRF have been given in [102, 147]. The simplest and well studied example is the $4 \times 4$ matrix

$$\mathscr{C} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \tag{7.4}$$

which has $\operatorname{rank}(\mathscr{C}) = 3$ and $\operatorname{rank}_+(\mathscr{C}) = 4$.

Needless to say, given any $A \in \mathbb{R}_+^{m\times n}$ , the first question should be to determine whether $A \in \mathfrak{R}(m,n)$, which we have already pointed out is an NP-hard problem. One of our contributions in this work is to show that if it is known that $\operatorname{rank}_+(A) = k$, with probability one we should have $\operatorname{rank}(A) = k$. The converse is not true. Deciding the conditional probability of $\operatorname{rank}_+(A) = k$, given $\operatorname{rank}(A) = k$ is still an open question. On the other hand, a necessary and sufficient condition for discerning whether a nonnegative matrix has an NRF is given in [156], but the algebraic operations of checking this qualification for a given matrix are numerically formidable. Other known results for checking the existence of an NRF are more restricted to certain subclasses of matrices such as the weakly monotone nonnegative matrices

[101], $\lambda$-monotone [100], or matrices with nonnegative 1-inverse [25].

The main crux in our investigation is to develop a numerical procedure for computing the NRF, if it exists. Certainly the precision of the resulting NRF is subject to the floating-point arithmetic errors. A perturbation analysis for the NRF is itself an interesting subject deserving further investigation, but no advance has been made thus far in the literature. Our current destination is simply to provide a heuristic algorithm that can serve as an experimental tool for studying the NRF. Our establishment is still preliminary, but we think our numerical approach is innovative. To facilitate the subsequent discussion, we introduce two basic terms below. First, any nonnegative matrix whose subtraction from a given nonnegative matrix $A$ remains nonnegative is called a *nonnegative component* (NC) of $A$. Second, any rank-one NC of a nonnegative matrix $A$ whose subtraction from $A$ reduce the rank of $A$ by one is called a *nonnegative element* (NE) of $A$. The fundamental difference between an NC and an NE is significant and could be illustrated through the following example.

**Example 7.1.1** *The matrix $\mathscr{C}$ defined in (7.4) has many NCs, but has no NE at all. Recall that* $\mathrm{rank}(\mathscr{C}) = 3$ *and* $\mathrm{rank}_+(\mathscr{C}) = 4$. *If there were an NE for $A_1 = \mathscr{C}$, then the residual matrix $A_2$ after its removal from $A_1$ would be nonnegative and of rank 2. It follows from [44, Theorem 4.1] that the matrix $A_2$ would automatically have nonnegative rank 2, implying the matrix $\mathscr{C}$ would have nonnegative rank 3. This is a contradiction.*

In [122], Levin describes a numerical algorithm for computing the "maximum" rank-one NC of any given nonnegative matrix. However, the subtraction of the NC characterized in Levin's algorithm cannot guarantee to lower the rank of the resulting matrix. In other words, the NC found by Levin is not an NE. Continuing this process might give rise to an infinite loop of finding nonnegative rank one matrices, which totally deviates the minimal constraint on NRF. We do not think Levin's algorithm is capable of computing the NRF for a given matrix in $\mathfrak{R}(m, n)$.

Unlike Levin's method, we propose an NRF algorithm based on the greedy rank-one reduction. Our idea is to gradually factorize $A$ over a sequence of NEs each of which are not only NCs, but will also reduce the rank by one. Our method is motived by the Wedderburn rank reduction formula [35, 163]. This formula provides the unique recipe on reducing the rank by one in each iteration. We modify the notion in our greedy method by finding one NE a time in each iteration. If $A \in \mathfrak{R}(m, n)$, then using our method should be able to find the NRF in at most $\mathrm{rank}(A)$ many iterations in exact arithmetic. On the other hand, a termination of our algorithm with nonzero residual suggests, but only heuristically, that the underlying nonnegative matrix is not in the class of $\mathfrak{R}(m, n)$.

This chapter is organized as follows. In Section 7.2, we briefly review the fundamental Wedderburn rank reduction formula. For our desirable rank-one reduction, we must take into

account the nonnegativity constraints during the reduction process. In Section 7.3, we recast the NRF as a constrained optimization problem and propose to solve the NRF by the available optimization techniques. Our approach is readily generalized in Section 7.4 to the even harder problem of decomposing the so called completely positive matrices where the two factors $U$ and $V$ must satisfy $U = V^\top$. We think this approach is new in this regard. In the case $A \notin \mathfrak{R}(m,n)$, we propose the notion of maximal nonnegative rank splitting of a nonnegative matrix in section 7.5. In Section 7.6, we outline a geometric meaning of the NRF which relate it to the classical Sylvester's problem. In Section 7.7, some empirical testing results are reported by our algorithm.

## 7.2 Wedderburn rank reduction formula

The Wedderburn rank reduction formula appears as a modest statement in Wedderburn's 1934 book [163] and is listed as an exercise in Householder's 1964 book [94]. In the review article [35], however, Chu, Funderlic, and Golub point out that perhaps all matrix factorizations can be expressed through this seemingly intuitive expression. In [97, Figure 2.1], Hubert, Meulman, and Heiser chronicle an interesting timeline for the appearance of generalized rank reduction results in the numerical linear algebra as well as the applied statistics and psychometrics literature. We review two basic facts below.

The first theorem characterizes a necessary and sufficient condition for rank subtraction by a rank-one matrix. This result plays a vital role in our application.

**Theorem 7.2.1** *Let $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$. Then the matrix*

$$B := A - \sigma^{-1}\mathbf{u}\mathbf{v}^\top \tag{7.5}$$

*satisfies the rank subtractivity* $\operatorname{rank}(B) = \operatorname{rank}(A) - 1$ *if and only if there are vectors $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^m$ such that*

$$\mathbf{u} = A\mathbf{x}, \quad \mathbf{v} = A^\top\mathbf{y}, \quad \sigma = \mathbf{y}^\top A\mathbf{x}. \tag{7.6}$$

Cline and Funderlic [43] then generalize the rank-one reduction formula (7.5) almost verbosely to simultaneous multiple rank reduction.

**Theorem 7.2.2** *Suppose $U \in \mathbb{R}^{m\times k}$, $R \in \mathbb{R}^{k\times k}$, and $V \in \mathbb{R}^{n\times k}$. Then*

$$\operatorname{rank}(A - UR^{-1}V^\top) = \operatorname{rank}(A) - \operatorname{rank}(UR^{-1}V^\top)$$

*if and only if there exist $X \in \mathbb{R}^{n\times k}$ and $Y \in \mathbb{R}^{m\times k}$ such that*

$$U = AX, \quad V = A^\top Y, \quad \text{and} \quad R = Y^\top AX. \tag{7.7}$$

The formula (7.5) provides a clue to the factorization of a matrix as the sum of a series of rank-one matrices. The basic idea is that starting with $A_1 = A$, define a sequence $\{A_k\}$ of matrices via the formula

$$A_{k+1} := A_k - (\mathbf{y}_k^\top A_k \mathbf{x}_k)^{-1} A_k \mathbf{x}_k \mathbf{y}_k^\top A_k. \tag{7.8}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ and $\mathbf{y}_k \in \mathbb{R}^m$ are properly chosen so that $\mathbf{y}_k^\top A_k \mathbf{x}_k \neq 0$. Since $\text{rank}(A_k)$ is reduced by one at each iteration, the sequence $\{A_k\}$ must be finite. The original matrix $A$ is thus broken down by the finite series of rank-one matrices. Different choices of $\mathbf{x}_k$ and $\mathbf{y}_k$ lead to different matrix decompositions, such as $LU$, $QR$, or SVD, used in applications. Further details of this discussion could be found in [35].

For our application, we want to break down a nonnegative matrix by taking away one NE at a time. Hence both the rank-one matrix in the Wedderburn form $(\mathbf{y}_k^\top A_k \mathbf{x}_k)^{-1} A_k \mathbf{x}_k \mathbf{y}_k^\top A_k$ and the resulting $A_{k+1}$ must remain nonnegative for some properly chosen vectors $\mathbf{x}_k \in \mathbb{R}^n$ and $\mathbf{y}_k \in \mathbb{R}^m$. If these nonnegativity constraints can be satisfied, the rank of $A_{k+1}$ is guaranteed to be one less than that of $A_k$. We then repeat the process until either that the rank is reduced to zero, implying that an NRF has been found, or that the process terminates prematurely, implying that the original matrix might not have an NRF. (See Example 7.3.1.) If $A_k$ is a nonzero and nonnegative matrix, then nonnegative rank-one matrices in the Wedderburn form are easy to find. A premature termination thus means that it is impossible to keep the residual $A_{k+1}$ nonnegative by *any* nonnegative rank-one reduction. There are two probable causes for this to happen. One is that the matrix $A$ itself is not a matrix in $\mathfrak{R}(m, n)$. This leads to the notion of maximal nonnegative rank splitting of $A$. The other is that some bad initial points for the optimization process overshoot the subtraction and inadvertently shut down the branching activity for $A_k$. A possible remedy for the latter cause is to restart the problem. In either cases, because we only have a heuristic algorithm in hand, we must carefully assess the situation before come to a conclusion about whether $A$ has an NRF or not.

## 7.3 Nonnegative rank factorization

We now explain how to recast the Wedderburn rank reduction formula for the NRF application. Note that the rank-one matrix $\mathbf{u}\mathbf{v}^\top$ is nonnegative if and only if all entries of $\mathbf{u}$ and $\mathbf{v}$ have same signs. Since the the scalar $\mathbf{y}_k^\top A_k \mathbf{x}_k$ in the Wedderburn formula (7.8) is either positive or negative, we may assume without loss of generality that $A_k \mathbf{x}_k \geq 0$, $\mathbf{y}_k^\top A_k \geq 0$, and $\mathbf{y}_k^\top A_k \mathbf{x}_k > 0$. Furthermore, we can scale the vectors such that $\mathbf{y}_k^\top A_k \mathbf{x}_k = 1$. These conditions are referred to as our "*nonnegativity constraints*". If these constraints are satisfied, then $A_k \mathbf{x}_k \mathbf{y}_k^\top A_k$ serves as a candidate of nonnegative rank-one matrix to be subtracted from $A_k$. The concern is whether

**Algorithm 10**: NRF $[U, V, p, \text{Iflag}] = \text{NRF}(A, \epsilon, Gmax, Lmax)$

**Input**:

| | | |
|---|---|---|
| $A$ | = | matrix in $\mathbb{R}_+^{m\times n}$ to be factorized |
| $\epsilon$ | = | threshold for machine zero |
| $Gmax, Lmax$ | = | maximal allowable numbers for retries |

**Output**:

| | | |
|---|---|---|
| $p$ | = | an integer, is the numerical $\text{rank}_+(A)$ if Iflag $= 0$ |
| Iflag | = | $\begin{cases} 0, & \text{An NRF is found with } \|A - UV\|_F < \epsilon \\ 1, & \text{Failed to find an NRF} \end{cases}$ |
| $U \in \mathbb{R}_+^{m\times p}$ and $V \in \mathbb{R}_+^{p\times n}$ | | |

10.1 **begin**
10.2 $Gstart \leftarrow 0$; $B \leftarrow A$;
`initialization`
10.3 $Gstart \leftarrow Gstart + 1$; $Lstart \leftarrow 0$; $U \leftarrow []$; $V \leftarrow []$; $p \leftarrow 0$;
10.4 **if** $\|B\|_F \geq \epsilon$ **then**
10.5 **if** $Gstart \leq Gmax$ **then**
10.6 $\mathbf{x}, \mathbf{y} \leftarrow$ feasible random starting points;
10.7 $[\mathbf{x}, \mathbf{y}, \text{ObjValue}] \leftarrow$ Solve (7.9) with respect to $B$ by available optimization routines;
10.8 **if** ObjValue $\geq 0$ **then**
10.9 $p \leftarrow p + 1$; $U \leftarrow [U, B\mathbf{x}]$; $V \leftarrow [V; \mathbf{y}^\top B]$; $B \leftarrow B - B\mathbf{x}\mathbf{y}^\top B$;
10.10 **else**
10.11 **if** $Lstart \leq Lmax$ **then**
10.12 $Lstart \leftarrow Lstart + 1$; Go to line 10.6;
10.13 **else**
10.14 Go to line 10.2;
10.15 **end**
10.16 **end**
10.17 **else**
10.18 Report that an NE is not found after retries; Iflag $= 1$; return;
10.19 **end**
10.20 **else**
10.21 Iflag $= 0$;
10.22 **end**
10.23 **end**

this nonnegative rank-one matrix can serve as an NE for $A_k$.

To search for a rank-one NE of $A_k$, we formulate a maximin problem as follows:

$$\begin{aligned} \max_{\mathbf{x}_k \in \mathbb{R}^n, \mathbf{y}_k \in \mathbb{R}^m} \quad & \min \left[ A_k - A_k \mathbf{x}_k \mathbf{y}_k^\top A_k \right], \\ \text{subject to} \quad & A_k \mathbf{x}_k \geq 0, \\ & \mathbf{y}_k^\top A_k \geq 0, \\ & \mathbf{y}_k^\top A_k \mathbf{x}_k = 1, \end{aligned} \tag{7.9}$$

where the minimum is taken over all entries of the matrix. By nonnegativity, $A_k - A_k \mathbf{x}_k \mathbf{y}_k^\top A_k$ is always less than or equal to $A_k$. The maximizer of $\min \left[ A_k - A_k \mathbf{x}_k \mathbf{y}_k^\top A \right]$ always exists. A nonnegative optimal value of (7.9) implies that $A_{k+1} \geq 0$. We have the rank-one matrix $A_k \mathbf{x}_k \mathbf{y}_k^\top A_k$ is intrinsically as a feasible NE for $A_k$ and we can repeat the process to search for the next NE of $A_{k+1}$. The prototype of our NRF calculation is sketched in Algorithm 10.

Since (7.9) is a nonlinear problem, most maximin algorithms are only able to find a local solution. But a local solution is enough for our application, if a nonnegative optimal value is obtained. In this event, a valid NE has been found. On the other hand, a negative optimal value indicates only that an NE has not been found yet. In this case, we can remedy this situation by trying another starting value with the hope that maybe another nonnegative optimal value could be found. Such a strategy has been applied twice in Algorithm 10. Line 10.12 is to restart the optimization solver with the current $A_k$ and Line 10.14 is to restart the entire NRF process with $A_0$. At present, we do not have any smart tactics for initiating the starting points, so we terminate the iteration after a fixed number of failures. Strictly speaking, our algorithm is inconclusive when it fails. But when it does perform the NRF, as we have experienced for matrices in R(m,n), our method seems to be the first of its kind to actually find the factorization.

**Example 7.3.1** *Consider the* $4 \times 5$ *matrix* $A = [\mathscr{C}; \mathbf{c}]$*, where* $\mathbf{c}$ *is a randomly chosen nonnegative vector from* $\mathbb{R}^4_+$ *such that this matrix* $A$ *is of full row rank and* $\operatorname{rank}_+(A) = \operatorname{rank}(A)$*. This particular matrix* $A$ *does have an NRF as we only need to factorize* $A$ *corresponding to its rows. On the other hand, if we start with these specific initial values* $\widehat{\mathbf{x}} = [0, 0, 0, 0, 1]^\top$ *and* $\widehat{\mathbf{y}} = [\alpha, -\alpha, -\alpha, \alpha]^\top$*, where* $\alpha = 1/(c_1 - c_2 - c_3 + c_4)$*, it can be checked that* $(\widehat{\mathbf{x}}, \widehat{\mathbf{y}})$ *is already a local maximizer to (7.9) and the nonnegative rank-one matrix* $\Delta := A\widehat{\mathbf{x}}\widehat{\mathbf{y}}^\top A == [0_4, \mathbf{c}]$ *is an NE. But, the remaining nonnegative matrix* $A - \Delta = [\mathscr{C}; \mathbf{0}]$*, which is of rank 3, does not have an NRF anymore as we have proved in Example 7.1.1. In other words, starting from these initial values* $(\widehat{\mathbf{x}}, \widehat{\mathbf{y}})$*, the matrix* $\Delta$ *would be the first NE extracted by our algorithm. But then our iteration would be stranded at a "dead end" because there is no more NE contained in* $\mathscr{C}$*. This situation is what we referred to earlier as "overshooting the subtraction and branching into the wrong direction". For circumstances like this, we suggest in Algorithm 10 by Line 10.14 to*

*restart the process entirely and it does fix the problem.*

The above example indicates that a "bad" NE can causes a break-down for Algorithm 10, even if the NRF does exists. With the built-in restart mechanism, we generally achieve an NRF for generic matrices in $\mathfrak{R}(m,n)$. Our extensive numerical experiments seem to validate that Algorithm 10, despite the fact that the accuracy of the numerical results depends highly on the stopping criteria set out in the underlying optimization solver, is generally robust.

For computation, Algorithm 10 is sufficient as a working procedure for the construction of a nonnegative rank-one approximation. From a theoretical point of view, Theorem 7.2.2 brings forth the idea of reducing multiple ranks of the matrix $A_k$ simultaneously in one iteration. Again, the role of the matrix $R^{-1}$ in Theorem 7.2.2 is immaterial. A natural generalization of the optimization problem (7.9) is in the form

$$\begin{aligned} \max_{X_k\in\mathbb{R}^{m\times r},Y_k\in\mathbb{R}^{n\times r}} \quad & \min\left[A_k - A_kX_kY_k^\top A_k\right], \\ \text{subject to} \quad & A_kX_k \geq 0, \\ & Y_k^\top A_k \geq 0, \\ & Y_k^\top A_kX_k = I_{r\times r}, \end{aligned} \tag{7.10}$$

where $I_{r\times r}$ is the identity matrix of rank $r$.

The following theorem describes an interesting fact that if the rank of the matrix $A_k$ can be reduced by $r$ in one iteration via (7.10), then the same result could be achieved via $r$ rank-one reduction steps through (7.9).

**Theorem 7.3.1** *If a nonnegative matrix has a nonnegative rank-r reduction, then it must have r nonnegative rank-one reductions.*

**Proof.** Suppose the nonnegative matrix $A$ has a nonnegative rank-$r$ reduction. By Theorem 7.2.2, we know there are matrices $X \in \mathbb{R}^{n\times r}$ and $Y \in \mathbb{R}^{m\times r}$ satisfying

$$AX \geq 0, \quad Y^\top A \geq 0, \quad Y^\top AX = I_{r\times r}$$

and making the matrix $B := A - AXY^\top A$ nonnegative with $\operatorname{rank}(B) = \operatorname{rank}(A) - r$. Denote the columns of $X$ and $Y$ as

$$X := [\mathbf{x}_1, \ldots, \mathbf{x}_r], \quad Y := [\mathbf{y}_1, \ldots, \mathbf{y}_r].$$

then we have

$$A\mathbf{x}_i \geq 0, \quad \mathbf{y}_i^\top A \geq 0, \quad \mathbf{y}_i^\top A\mathbf{x}_j = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases}$$

We now show that each pair $(\mathbf{x}_i, \mathbf{y}_i)$, $i = k, \ldots, r$, has the desirable effect of rank-one reduction on $A_k$ defined successively by (7.8), starting with $A_1 = A$, and that $A_{r+1} = B$.

The case $k = 1$ is trivial. It is obvious that for $1 \leq i, j \leq r$, we have

$$A_1\mathbf{x}_i = A\mathbf{x}_i \geq 0, \quad \mathbf{y}_i^\top A_1 = \mathbf{y}_i^\top A \geq 0, \quad \mathbf{y}_i^\top A_1\mathbf{x}_j = \mathbf{y}_i^\top A\mathbf{x}_j = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases}$$

Assume the statement that

$$A_k\mathbf{x}_i \geq 0, \quad \mathbf{y}_i^\top A_k \geq 0, \quad \mathbf{y}_i^\top A_k x_j = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j, \end{cases}$$

is true for all $k \leq i, j \leq r$. Recall that $A_{k+1} = A_k - A_k\mathbf{x}_k\mathbf{y}_k^\top A_k$. It follows that for all $k + 1 \leq i, j \leq r$, we have

$$\begin{aligned} A_{k+1}\mathbf{x}_i &= (A_k - A_k\mathbf{x}_k\mathbf{y}_k^\top A_k)\mathbf{x}_i = A_k x_i \geq 0, \\ \mathbf{y}_i^\top A_{k+1} &= \mathbf{y}_i^\top (A_k - A_k\mathbf{x}_k\mathbf{y}_k^\top A_k) = \mathbf{y}_i^\top A_k \geq 0, \\ \mathbf{y}_i^\top A_{k+1}\mathbf{x}_j &= \mathbf{y}_i^\top (A_k - A_k\mathbf{x}_k\mathbf{y}_k^\top A_k)\mathbf{x}_j = \mathbf{y}_i^\top A_k\mathbf{x}_j = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} \end{aligned}$$

By the mathematical induction, we conclude that the very same matrix $B$ can be achieved by $r$ rank-one reductions via the sequence of vectors $\{(\mathbf{x}_k, \mathbf{y}_k)\}_{k=1}^r$. □

Theorem 7.3.1 delivers an important message that no matrix can have a nonnegative rank-$r$ reduction by a rank-$r$ matrix without first having a nonnegative rank-one reduction by a rank-one matrix. Additionally, nonnegative rank-one reduction is preferable from the standpoint of computational complexity. For rank-one reduction, it involves $n + m$ variables in the problem (7.9) and need to solve the optimization problem repeatedly $r$ times (if no restart). For rank-$r$ reduction, it involves $r(n + m)$ variables in the problem (7.10), but it only need to solve the problem once. As $r$ grows larger, the complexity involved in solving the optimization problem (7.10) even by the same optimization solver would grow far more rapidly than the total complexity of $r$ applications to the problem (7.9).

## 7.4 Completely positive matrices

A nonnegative semi-definite matrix $A \in \mathbb{R}_+^{n\times n}$ is said to be *completely positive* (CP) if and only if $A$ can be factorized as

$$A = BB^\top, \tag{7.11}$$

where $B$ is nonnegative [15]. The matrix $B$ is not necessarily square. The smallest number of columns of $B$ satisfying the factorization (7.11) is called the *cp-rank* of the matrix $A$, denoted by $\text{rank}_{cp}(A)$. If $A$ is CP, then clearly $\text{rank}_+(A) \leq \text{rank}_{cp}(A)$. There has been considerable interest in the CP matrices and their properties. An upper bound estimate for the cp-rank, for example, can be found in [8, 84],

$$\text{rank}_{cp}(A) \leq \frac{\text{rank}(A)(\text{rank}(A)+1)}{2} - 1,$$

provided $\text{rank}(A) > 1$. Some sufficient conditions under which $\text{rank}_{cp}(A) = \text{rank}(A)$ can be found in [149]. Of particular intrigue is the case that if $A$ is generated by a Soules matrix, then $\text{rank}_{cp}(A) = \text{rank}(A)$ [150]. General properties and some applications of CP matrices are discussed in the book [15].

The determination of whether a given nonnegative semi-definite matrix is CP is an open question. Computing the CP factorization of a CP matrix is another very challenging task [14]. The symmetric form demanded in (7.11) seems to make the CP factorization more difficult than the NRF problem It turns out that with slight modification our NRF algorithm can determine heuristically whether $\text{rank}_{cp}(A) = \text{rank}(A)$ and, if it is affirmative, we can compute the factor $B$ numerically.

It is almost obvious how the Wedderburn formula should be modified to assure symmetric rank-one reduction. In order to satisfy the symmetric condition, we choose the vector $\mathbf{x}$ to be the same as $\mathbf{y}$ in the rank reduction formula [36]. Starting with $A_1 = A$, the optimization problem in (7.9) is reformulated as follows:

$$\begin{aligned} \max_{\mathbf{x}_k \in \mathbb{R}^n} \quad & \min\left[A_k - A_k\mathbf{x}_k\mathbf{x}_k^\top A_k\right] \\ \text{subject to} \quad & A_k\mathbf{x}_k \geq 0 \\ & \mathbf{x}_k^\top A_k \mathbf{x}_k = 1 \end{aligned} \tag{7.12}$$

with $A_{k+1} := A_k - A_k\mathbf{x}_k\mathbf{x}_k^\top A_k$. If the optimal value at a local maximizer is nonnegative, then a symmetric NE has been found for the matrix $A_k$. Otherwise, a treatment similar to the NRF problem can be applied to the CP problem until no more symmetric NE could be found. Since

the rank is reduced by one in each step, the process could only be repeated at most rank($A$) iterations in exact arithmetic. In the event that $\text{rank}_{cp}(A) = \text{rank}(A)$, our algorithm offers a numerical procedure to discover the nonnegative factor

$$B = (A_1\mathbf{x}_1, \ldots, A_{\text{rank}(A)}\mathbf{x}_{\text{rank}(A)})$$

heuristically. Our approach perhaps represents only a modest advance toward the CP problem, but so far as we know, there is no other way capable of computing the factorization of CP matrix in the literature.

## 7.5 Maximal nonnegative rank splitting

Due to the rank reduction nature, our process cannot extract more than rank($A$) rank-one NEs. To our knowledge, currently there is simply no techniques to handle nonnegative rank factorization for the cases $\text{rank}_+(A) > \text{rank}(A)$ (or cp-rank factorization for the case $\text{rank}_{cp}(A) > \text{rank}(A)$). Still, we can employ our algorithm to address one interesting issue on the so called maximal nonnegative rank splitting (MNRS) defined below.

**(MNRS)**: Given a nonnegative matrix $A$, find a splitting

$$A = B + C, \tag{7.13}$$

where both $B$ and $C$ are nonnegative matrices satisfying

$$\begin{aligned} \text{rank}(B) &= \text{rank}_+(B), \\ \text{rank}(A) &= \text{rank}(B) + \text{rank}(C), \end{aligned}$$

and rank($B$) is maximized.

If $A \in \mathfrak{R}(m,n)$, then trivially $B = A$ and $C = 0$. Considering $A \notin \mathfrak{R}(m,n)$, we know that $A$ has no NRF. However, it is plausible that $A$ still has a few NEs. Repeatedly apply our method until it has to be terminated (after many retries), say, at $A_k$. If we trust that $A_k$ has no more NE, then $B := A - A_k$ and $C := A_k$ form an MNRS for $A$.

We mention few examples to demonstrate the notion of MNRS.

**Example 3.** The matrix $\mathscr{C}$ defined in (7.4) has zero $B$ component in its MNRS.

**Example 4.** Consider the $8\times 8$ nonnegative matrix $A := [W, H]$, where $W := \begin{bmatrix} \mathscr{C}^\top & 0_4 \end{bmatrix}^\top \in$

$\mathbb{R}^{4\times 8}$ and $H \in \mathbb{R}^{8\times 4}$ is made of the product $H = H_1 H_2$ with

$$H_1 := \begin{bmatrix} 0.2917 & 0.3109 & 0.2026 \\ 0.4665 & 0.2558 & 0.9396 \\ 0.9439 & 0.1048 & 0.2107 \\ 0.0943 & 0.2903 & 0.9670 \\ 0.0119 & 0.4985 & 0.6356 \\ 0.3723 & 0.8205 & 0.4252 \\ 0.3542 & 0.3074 & 0.2262 \\ 0.0820 & 0.7715 & 0.9325 \end{bmatrix} \quad \text{and} \quad H_2 := \begin{bmatrix} 0.7426 & 0.2143 & 0.0907 & 0.1922 \\ 0.5133 & 0.8007 & 0.8121 & 0.0639 \\ 0.5417 & 0.6280 & 0.0968 & 0.4969 \end{bmatrix}.$$

Since $3 = \text{rank}(H) \leq \text{rank}_+(H)$, it is clearly that $\text{rank}(H) = \text{rank}_+(H) = 3$. Hence, $\text{rank}(A) = 6$. Does $A$ have an NRF, or can we retrieve an MNRS of $A$?

Apply Algorithm 10 to the matrix $A$. The farthest we can go is a splitting $A = B + C$ with

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0.17825 & 0.36892 & 0.29782 & 0.08302 \\ 0 & 0 & 0 & 0 & 0.56400 & 0.47683 & 0.34027 & 0.18382 \\ 0 & 0 & 0 & 0 & 0.38505 & 0.33387 & 0.19112 & 0.16500 \\ 0 & 0 & 0 & 0 & 0.14403 & 0.42702 & 0.33791 & 0.09387 \\ 0 & 0 & 0 & 0 & 0.60902 & 0.80086 & 0.46744 & 0.34997 \\ 0 & 0 & 0 & 0 & 0.92796 & 1.00378 & 0.74126 & 0.33527 \\ 0 & 0 & 0 & 0 & 0.54335 & 0.46409 & 0.30366 & 0.20012 \\ 0 & 0 & 0 & 0 & 0.96204 & 1.22092 & 0.72424 & 0.52842 \end{bmatrix}$$

and

$$C = \begin{bmatrix} 1 & 1 & 0 & 0 & 0.30770 & 0.06976 & 0.00073 & 0.09358 \\ 1 & 0 & 1 & 0 & 0.42271 & 0.41803 & 0.00073 & 0.38908 \\ 0 & 1 & 0 & 1 & 0.48382 & 0.08464 & 0 & 0.12781 \\ 0 & 0 & 1 & 1 & 0.59883 & 0.43291 & 0 & 0.42331 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

where for the ease of running text we have displayed all numerals in 5 digits only. Since the result is derived from the Wedderburn rank reduction process, it is guaranteed by theory that $\text{rank}(B) = \text{rank}_+(B) = 3$ and $\text{rank}(C) = \text{rank}(A) - \text{rank}(B) = 3$. However, the special matrix $\mathscr{C}$ is still embedded at the upper left corner of $C$. This forces the fact that $\text{rank}_+(C) = 4$ and that $C$ has no more NE. Hence, this splitting is an MNRS of $A$.

## 7.6 Geometric meaning of nonnegative rank

This section is to briefly investigate a geometric meaning of the nonnegative rank. This meaning not only relates the notionof NRF to the classical Sylvester's problem, but also brings forth an interesting conclusion on the conditional probability of the NRF.

For a given matrix $A \in \mathbb{R}_+^{m\times n}$, denote its columns by $A = [\mathbf{a}_1, \ldots, \mathbf{a}_n]$. Assume that the matrix $A$ can be factorized as $A = UV$, where $U \in \mathbb{R}_+^{m\times p}$ and $V \in \mathbb{R}_+^{p\times n}$. Following the discussion in Section 5.2, let

$$\vartheta(A) := A\sigma(A)^{-1}, \tag{7.14}$$

with

$$\sigma(A) := \operatorname{diag}\left\{\|\mathbf{a}_1\|_1, \ldots, \|\mathbf{a}_n\|_1\right\}, \tag{7.15}$$

denote the projection of $A$ onto the probability simplex $\mathcal{D}_m$. Assume without loss of generality, that $\sigma(U) = I_n$. Then, we have

$$\begin{aligned} \vartheta(A) &= \vartheta(U)\vartheta(V), & (7.16)\\ \sigma(A) &= \sigma(V). & (7.17) \end{aligned}$$

Consequently, we have the following result immediately.

**Lemma 7.6.1** *Given a nonnegative matrix $A \in \mathbb{R}^{m\times n}$, then* $\operatorname{rank}_+(A) = \operatorname{rank}_+(\vartheta(A))$.

**Proof.** Suppose that $\operatorname{rank}_+(A) = p$, without lose of generality, we might factorize $A$ as the sum of $p$ rank-one matrices, $\mathbf{u}_i\mathbf{v}_i^\top$ for vectors $\mathbf{u}_i \in \mathbb{R}^m$, $\mathbf{v}_i \in \mathbb{R}^n$ and $i = 1, \cdots p$ such that

$$A = \sum_{i=1}^{p} \mathbf{u}_i\mathbf{v}_i^\top \tag{7.18}$$

We know that $A = \vartheta(A)\sigma(A)$. Hence

$$\begin{aligned} A\sigma^{-1}(A) &= (\textstyle\sum_{i=1}^{p} \mathbf{u}_i\mathbf{v}_i^\top)\sigma^{-1}(A)\\ &= \textstyle\sum_{i=1}^{p} \mathbf{u}_i(\mathbf{v}_i^\top\sigma^{-1}(A)) \qquad (7.19)\\ &= \vartheta(A) \end{aligned}$$

Following from formula (7.19), we get that $\operatorname{rank}_+(\vartheta(A)) \leq \operatorname{rank}_+(A)$. In a similar way, we could prove the other direction of the inequality. □

It thus suffices to consider the geometric meaning of $\operatorname{rank}_+(\vartheta(A))$ on the simplex $\mathcal{D}_m$. By the relationship (7.16), we have the following interpretation of the nonnegative rank.

**Lemma 7.6.2** *The integer* $\operatorname{rank}_+(A)$ *represents the minimal number of vertices on $\mathcal{D}_m$ so that the resulting convex polytope encloses all columns of the pull-back $\vartheta(A)$.*

In other words, a nonnegative matrix $A$ has an NRF if and only if the matrix $\vartheta(A)$ could be enclosed by the the minimal convex polytope with exactly rank$(A)$ many vertices on $\mathcal{D}_m$. Using this notion, we are now able to explain why the matrix $\mathscr{C}$ in (7.4) does not have an NRF by geometry.

Since each point $\mathbf{a} = [a_1, a_2, a_3, a_4]^\top \in \mathcal{D}_4$ satisfies $a_1+a_2+a_3+a_4 = 1$, it suffices to represent the 4-dimensional vector $\mathbf{a}$ by its first three entries $[a_1, a_2, a_3]^\top$. Through this transformation, the probability simplex $\mathcal{D}_4$ can easily be visualized through the unit tetrahedron $S$ in the first octant of $\mathbb{R}^3$. In particular, columns of $\vartheta(\mathscr{C})$ can be interpreted as points $A_1, A_2, A_3, A_4$ depicted in Figure 8.2. Also, the four points $A_1, A_2, A_3, A_4$ are coplanar because rank$(\vartheta(\mathscr{C})) = 3$. On

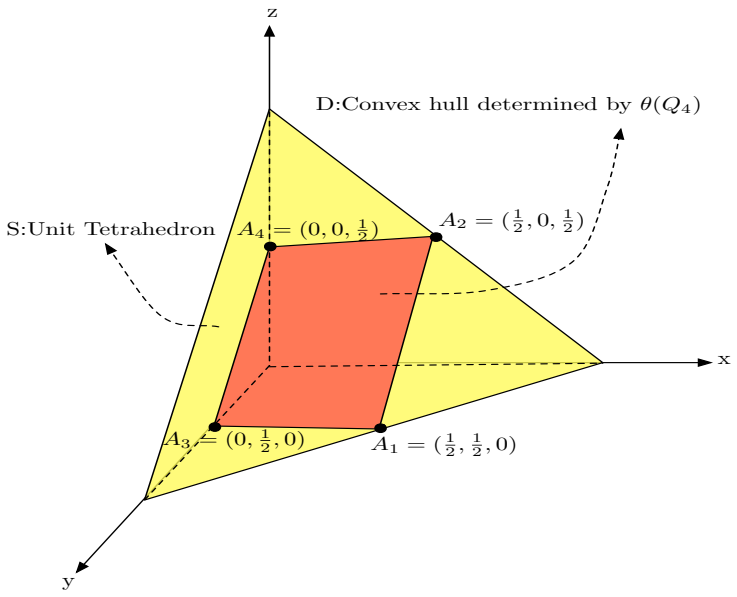


Figure 7.1: A geometric representation of the matrix $\vartheta(\mathscr{C})$.

the other hand, these four points of the convex hull $D$ sits on four separate "ridges" of the tetrahedron, which cannot be enclosed by any three-vertex convex set in the tetrahedron. The minimum number of vertices for a convex set in the unit tetrahedron to enclose $D$ is four, hence rank$_+(\mathscr{C}) = 4$. The point to make is that the columns of $\mathscr{C}$ are sitting at very strategic positions in the unit tetrahedron. An interesting question to ask is how often this can happen. Or, to put it differently, we might want to ask the following two questions about geometric probability:

($\mathbf{R2R}_+$) : Given an arbitrary nonnegative 4 by 4 matrix of rank 3, what is the probability that its nonnegative rank is 3?

($\mathbf{R_+2R}$) : Given an arbitrary nonnegative 4 by 4 matrix of nonnegative rank 3, what is the probability that its rank is 3?

To answer $R2R_+$, it suffices to consider matrices in the set

$$\mathfrak{E} := \left\{ A \in R_+^{4\times 4} \mid \text{rank}(A) = 3,\ A = \vartheta(A) \right\}.$$

For $A \in \mathfrak{E}$, the four corresponding points obtained from deleting the last row of $A$ must be in a plane that intersects the unit tetrahedron. Hence there are two mutually exclusive cases for this $R2R_+$. First, the cross-section of the plane in the unit tetrahedron is a triangle. Naturally this triangle encloses the four points. In this case, the matrix has nonnegative rank 3. Then what is the probability of a randomly selected plane in the unit tetrahedron to have a triangular cross-section? Second, the cross-section is a quadrilateral. Then what is the conditional probability that the four points representing $A$ in the unit tetrahedron are enclosed in a triangle within the quadrilateral?

Both questions are not easy to be answered. Especially, we must reconcile the first question concerning the basic geometric probability with how the plane cuts through the unit tetrahedron [109]. It is known that for this question alone, different definitions of randomness will lead to different answers. For the second question, it is related to the well known Sylvester's four-point problem which asks for the probability, denoted by $p(4, K)$, of four random, independent, and uniform points from a compact set $K$ such that none of them lies in the triangle formed by the other three [7]. Hence, the conditional probability of the event that $\text{rank}_+(A) = 3$, given $A \in \mathfrak{E}$ is in some compact quadrilateral $K$, is greater than or equal to $1 - p(4, K)$. Nevertheless, it is a well known story that $p(4, K)$ itself has a number of different answers [141]. Thus far, no one could proclaim an exact solution for this problem and Sylvester has to proclaim in [155] that, "This problem does not admit of a determinate solution!" For this reason, we currently do not have a definitive answer to the seemingly simple problem $R2R_+$.

The flipped side question of $R2R_+$ is also interesting. It turns out we have a surprising answer even for the general cases. We think this result is important and new.

**Theorem 7.6.1** *Given $k < \min\{m, n\}$, let $R_+(k)$ denote the manifold of nonnegative matrices in $\mathbb{R}_+^{m\times n}$ with nonnegative rank $k$. Then the conditional probability of* $\text{rank}(A) = k$, *given $A \in R_+(k)$, is one.*

**Proof.** Since $\text{rank}(A) = \text{rank}(\vartheta(A))$ and $\text{rank}_+(A) = \text{rank}_+(\vartheta(A))$, we may, without loss of generality, assume $A = \vartheta(A)$. Note that the subspace orthogonal to columns of $A$ is of dimension $m - \text{rank}(A)$ and the column sum for $A$ is always 1. Together, columns of $A$ satisfy $m - \text{rank}(A) + 1$ independent linear equations. Thus these vertices should reside on an affine subspace of dimension $\text{rank}(A) - 1$.

If $A \in R_+(k)$, then $\text{rank}_+(A) = k$. It is known that the probability of $k$ distinct points in $\mathbb{R}^m$ to be in an affine subspace of dimension strictly less than $k-1$ is zero. Hence the conditional probability of $\text{rank}(A) < k$, given $\text{rank}_+(A) = k$, is zero. That is, if $\text{rank}_+(A) = k$, then with probability one we have $\text{rank}(A) = k$. □

## 7.7 Numerical experiments

In general, a nonnegative matrix may not have an NRF. In this case, we have already described in Section 7.5 how to calculate the MNRS. In this section, we demonstrate the working of our algorithm by computing the NRF of a few nontrivial matrices. For illustration purpose, we employ the MATLAB routine FMINIMAX as our optimization solver in all our computation and choose $\epsilon = 10^{-10}$ as the threshold for machine zero in Algorithm 10.

To ensure a given $m \times n$ matrix to be in $\mathfrak{R}(m, n)$, we rely on Theorem 7.6.1 to generate test data.

**Example 7.7.1** *Randomly generate two nonnegative matrices $W \in \mathbb{R}_+^{5\times 3}$ and $H \in \mathbb{R}_+^{3\times 5}$*

$$W = \begin{bmatrix} 0.9708 & 0.2140 & 0.4120 \\ 0.9901 & 0.6435 & 0.7446 \\ 0.7889 & 0.3200 & 0.2679 \\ 0.4387 & 0.9601 & 0.4399 \\ 0.4983 & 0.7266 & 0.9334 \end{bmatrix}, \quad H = \begin{bmatrix} 0.6833 & 0.2071 & 0.4514 \\ 0.2126 & 0.6072 & 0.0439 \\ 0.8392 & 0.6299 & 0.0272 \\ 0.6288 & 0.3705 & 0.3127 \\ 0.1338 & 0.5751 & 0.0129 \end{bmatrix}^\top,$$

*and define $A = WH$. By construction, the matrix $A$ clearly has an NRF. Then we use our method to discover a new NRF for the matrix $A = UV$ such that*

$$U = \begin{bmatrix} 0.02556251462152 & 0.00563995049828 & 0.05020141897100 \\ 0.02170073447446 & 0.00833461311952 & 0.10441556394589 \\ 0.01809150541560 & 0.01018126666230 & 0.05049608516928 \\ 0 & 0.01848758106428 & 0.11124821245836 \\ 0.00844625518446 & 0 & 0.11504734974715 \end{bmatrix}$$

*and*

$$V = \begin{bmatrix} 22.65196335171072 & 0 & 6.26680882859984 \\ 2.42501879200814 & 7.93355352839419 & 4.93382971188032 \\ 21.86596055059507 & 15.79385557484764 & 6.22840681607381 \\ 18.88971723342108 & 4.21210726740658 & 6.21364289968766 \\ 0 & 7.37487007847031 & 4.31631898597720 \end{bmatrix}^\top,$$

*where, for convenience, we have transcribed all digits of the computed result.*

**Example 7.7.2** *To demonstrate the NRF of a CP matrix, we randomly generate a nonnegative matrix $W$*

$$W = \begin{bmatrix} 0.3840 & 0.0158 & 0.6315 & 0.3533 \\ 0.6831 & 0.0164 & 0.7176 & 0.1536 \\ 0.0928 & 0.1901 & 0.6927 & 0.6756 \\ 0.0353 & 0.5869 & 0.0841 & 0.6992 \\ 0.6124 & 0.0576 & 0.4544 & 0.7275 \\ 0.6085 & 0.3676 & 0.4418 & 0.4784 \end{bmatrix},$$

*and define $A = WW^\top$. By construction, $A$ is CP. Our algorithm shows that the matrix $A$ has a nonnegative decomposition $A = BB^\top$ with*

$$B = \begin{bmatrix} 0.58354630329629 & 0.35203758768455 & 0.44908808470853 & 0.07198556063105 \\ 0.58153426004785 & 0.21962336143449 & 0.78677873223851 & 0 \\ 0.67910036260539 & 0.70358647639519 & 0.14987548760103 & 0.04842113133372 \\ 0 & 0.91741547294584 & 0 & 0 \\ 0.45303579486590 & 0.65652500722113 & 0.57125648786549 & 0.38921284477200 \\ 0.28134324531544 & 0.66368746544619 & 0.63911349062806 & 0.03680601340055 \end{bmatrix}.$$

**Example 7.7.3** *For a general matrix $R$ with negative entries, it cannot be guaranteed that the product $RR^\top$ has an NRF. However, if $R$ is a Soules matrix, by definition, the product $RDR^\top$ is nonnegative and $\operatorname{rank}_{cp}(RDR^\top) = \operatorname{rank}(RDR^\top)$ for every nonnegative diagonal matrix $D$ with nonincreasing diagonal elements [150]. The matrix*

$$R = \begin{bmatrix} 0.1348 & 0.1231 & 0.1952 & 0.3586 & 0.8944 \\ 0.2697 & 0.2462 & 0.3904 & 0.7171 & -0.4472 \\ 0.4045 & 0.3693 & 0.5855 & -0.5976 & 0 \\ 0.5394 & 0.4924 & -0.6831 & 0 & 0 \\ 0.6742 & -0.7385 & 0 & 0 & 0 \end{bmatrix}$$

*has negative entries, but is a Soules matrix [58]. With this $R$ and with*

$$D = \operatorname{diag}([0.7, 0.5, 0.4, 0, 0]),$$

*We can define a nonnegative matrix*

$$A = RDR^\top = \begin{bmatrix} 0.035537749 & 0.071084934 & 0.106614875 & 0.027868556 & 0.018162837 \\ 0.071084934 & 0.142188747 & 0.213258065 & 0.055774870 & 0.036372868 \\ 0.106614875 & 0.213258065 & 0.319849520 & 0.083670750 & 0.054535705 \\ 0.027868556 & 0.055774870 & 0.083670750 & 0.511545776 & 0.072745736 \\ 0.018162837 & 0.036372868 & 0.054535705 & 0.072745736 & 0.590873073 \end{bmatrix}.$$

*By theory, $A$ is a CP matrix. Also, because of the two zeros in $D$, we know that* $\operatorname{rank}(A) = 3$.

*By using our method, we find* $\operatorname{rank}_{cp}(A) = 3$ *as is expected and obtain a decomposition* $A = BB^\top$ *with*

$$B = \begin{bmatrix} 0.18851458564260 & 0 & 0 \\ 0.37707922576755 & 0.00005556024891 & 0.00003751809670 \\ 0.56555239286434 & 0.00006079293882 & 0.00008502728571 \\ 0.14783235952195 & 0.07671246735655 & 0.69556205102811 \\ 0.09634711785325 & 0.76262068283226 & 0 \end{bmatrix}.$$

## 7.8 Conclusion

Different from the NMF where the factorization is done approximately, the NRF insists on the holding of equality to the original matrix by the product of its factors. The NRF is a much harder problem both in theory and in computation. Our contribution in this investigation is to propose a numerical method that is able to detect whether a given nonnegative matrix does have an NRF and find such a factorization, if it exists. Although at present our method is still heuristic and can break down, empirical results seem to strongly suggest that our algorithm can handle the situation reasonable well for generic nonnegative matrices. Even for a matrix without NRF, we can also study its MNRS. Our method also can be applied to explore the CP factorization for a CP matrix. All of these would have been extremely difficult, if not possible, in the literature.

The most important feature in our algorithm is the employment of the Wedderburn rank reduction formula. At each iteration, we uncover an NE that not only reduces the rank by one, but also maintains a nonnegativity residual.

# Chapter 8

# On the Nonnegative Rank of Euclidean Distance Matrices

## 8.1 Overview

This short chapter exemplifies the rich field of many unanswered questions relevant to our study. We have pointed out earlier that generically a matrix $A$ with $\text{rank}_+(A) = k$ should have $\text{rank}(A) = k$. But what are the exceptions? In this chapter we study the analysis of the nonnegative rank of a very important class of nonnegative matrices, the so called *Euclidian distance matrices* (EDM). The notion of distance geometry, initiated by Menger and Schoenbeng in the 1930s, has been an area of active research because of its many important applications, including molecular conformation problems in chemistry [48], multidimensional scaling in behavioral sciences [53, 129], and multivariate analysis in statistics [130]. The article [74] is an excellent reference that expounds the framework of Euclidean distance geometry in general. More extensive discussion on the background and applications of distance geometry can be found in [48]. Our contribution here is to show via a geometric argument that while the EDMs always have low algebraic ranks, their nonnegaive rank usually is full. Specifically, we show that the EDM of $n$ distinct points in $\mathbb{R}$ has nonnegative rank $n$ and conjecture that same is true for points in higher dimensional spaces. In short, the EDMs have no NRF in general. This work represents perhaps only a modest advance in the field, but it should be of interest to confirm the precise rank and nonnegative rank of a distance matrix.

## 8.2 Rank condition and standard form

We first briefly review the rank condition of EDMs. For this particular of nonnegative matrices, we also introduce the notion of standard form of matrix factorization Corresponding to any given

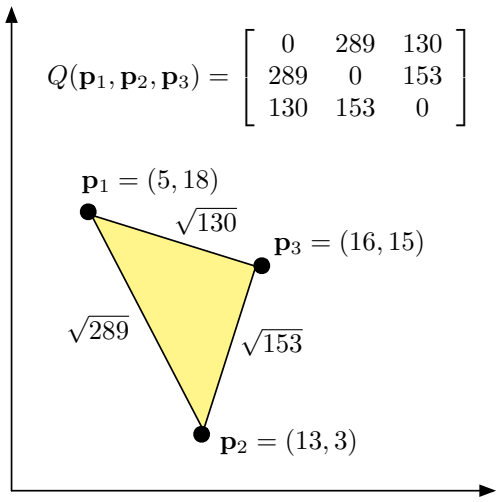


Figure 8.1: Euclidean matrix of three points in $\mathbb{R}^2$.

$n$ points $\mathbf{p}_1, ..., \mathbf{p}_n$ in the space $\mathbb{R}^r$, the EDM is an $n \times n$ symmetric and nonnegative matrix $Q(\mathbf{p}_1, \ldots, \mathbf{p}_n) = [q_{ij}]$ whose entry $q_{ij}$ is defined by

$$q_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|^2, \quad i, j = 1, \ldots, n, \tag{8.1}$$

where $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^r$. That is, the EDM in $\mathbb{R}^{n\times n}$ is an exhaustive representation of the relative distance between any two points of $\{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ in $\mathbb{R}^r$. An example for the case $n = 3$ and $r = 2$ is sketched in Figure 8.1. It is easy to see that this matrix is symmetric with zero diagonal.

Among the many properties of EDMs, perhaps the following rank condition of $Q(\mathbf{p}_1, \ldots, \mathbf{p}_n)$ is most peculiar. There are many different ways in the literature to establish this result. For the sake of connecting the matrix factorization, we adopt the following constructive and straightforward proof whose simplicity is worth noting.

**Theorem 8.2.1** *Suppose $\mathbf{p}_1, \cdots, \mathbf{p}_n \in \mathbb{R}^r$. Then for any $n \geq r + 2$, the rank of $Q(\mathbf{p}_1, \ldots, \mathbf{p}_n)$ is no greater than $r + 2$ and is generically $r + 2$.*

**Proof.** It is true that [10]

$$Q(\mathbf{p}_1,\ldots,\mathbf{p}_n) = \underbrace{\begin{bmatrix} \|\mathbf{p}_1\|^2 & 1 & -2\mathbf{p}_1^\top \\ \vdots & \vdots & \vdots \\ \|\mathbf{p}_i\|^2 & 1 & -2\mathbf{p}_i^\top \\ \vdots & \vdots & \vdots \\ \|\mathbf{p}_n\|^2 & 1 & -2\mathbf{p}_n^\top \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 1 & \ldots & 1 & \ldots & 1 \\ \|\mathbf{p}_1\|^2 & \ldots & \|\mathbf{p}_j\|^2 & \ldots & \|\mathbf{p}_n\|^2 \\ \mathbf{p}_1 & & \mathbf{p}_j & & \mathbf{p}_n \end{bmatrix}}_{V}. \tag{8.2}$$

where $U \in \mathbb{R}^{n\times(r+2)}$ and $V \in \mathbb{R}^{(r+2)\times n}$. Hence, the rank of matrices $U$ and $V$ are generically equal to $r+2$ unless the points $\mathbf{p}_1,\ldots,\mathbf{p}_n$ satisfy some particular algebraic conditions such as $\|\mathbf{p}_\ell\| = 1$ for all $\ell = 1,\ldots n$. □

It is important to note that the rank of an EDM is independent of the number $n$ of points given, but depends only on the dimension of the ambient space. In other words, the size of an EDM can be arbitrarily large, but its rank remains very low and generically is a constant. The rank deficient property of the EDM implies that many entries in the matrix are redundant and could be replaced by the other entries. It is nature to ask the question of whether $\text{rank}_+(Q(\mathbf{p}_1,\ldots,\mathbf{p}_n))$ has similar property. The two factors $U$ and $V$ in the decomposition certainly cannot be nonnegative simultaneously. The (minimum) nonnegative factorization of $Q(\mathbf{p}_1,\ldots,\mathbf{p}_n)$ is yet to be analyzed.

Suppose that a given nonnegative matrix $A$ has two nonnegative factorizations, $A = BC$ and $A = FG$, We say that these two factorizations are *equivalent* if there exist a permutation matrix $P$ and a diagonal matrix $D$ with positive diagonal elements such that $BDP = F$ and $P^\top D^{-1}C = G$ [10]. With this notion in mind, we can rewrite every nonnegative factorization of an EDM into the following "standard form".

**Lemma 8.2.1** *Given $n$ distinct points $\mathbf{p}_1,\ldots,\mathbf{p}_n$ in $\mathbb{R}^r$ with $n \geq r+2 \geq 3$, then any nonnegative factorization of $Q(\mathbf{p}_1,\ldots,\mathbf{p}_n)$ is equivalent to the form*

$$Q(\mathbf{p}_1,\ldots,\mathbf{p}_n) = \begin{bmatrix} 1 & 0 & * & * & \ldots \\ * & 1 & 0 & * & \ldots \\ 0 & * & 1 & * & \ldots \\ * & * & * & * & \\ \vdots & & & & \end{bmatrix} \begin{bmatrix} 0 & * & + & * & \ldots \\ + & 0 & * & * & \\ * & + & 0 & * & \\ * & * & * & * & \\ \vdots & & & & \end{bmatrix} \tag{8.3}$$

*where $*$ stands for some undetermined nonnegative numbers and $+$ stands for three undetermined positive numbers.*

**Proof.** Let $Q(\mathbf{p}_1,\ldots,\mathbf{p}_n) = UV$ be a nonnegative factorization with $Q(\mathbf{p}_1,\ldots,\mathbf{p}_n) = [q_{ij}]$,

$U = [u_{ij}]$ and $V = [v_{ij}]$. Since $\mathbf{p}_1$ and $\mathbf{p}_3$ are distinct (i.e., $q_{13} > 0$), there must exist an index $1 \leq k_1 \leq n$ such that $u_{1k_1}v_{k_13} > 0$. Place $u_{1k_1}$ at the $(1,1)$ position and $v_{k_13}$ at the $(1,3)$ position by permuting both the first and the $k_1$-th columns of $U$ and the first and the $k_1$-th rows of $V$ simultaneously. This permutation will not affect the product. After scaling $u_{1k_1}$ to unit, rename without causing ambiguity the permuted matrices as $U$ and $V$, respectively. As $q_{11} = 0$, the corresponding $v_{11}$ in the new $V$ must be zero.

Since $\mathbf{p}_1$ and $\mathbf{p}_2$ are distinct (i.e., $q_{12} > 0$), there must exist an index $2 \leq k_2 \leq n$ such that $u_{2k_2}v_{k_21} > 0$. Similarly, place $u_{2k_2}$ at the $(2,2)$ position and $v_{k_21}$ at the $(2,1)$ position by permuting the second and the $k_2$-th columns of $U$ and the second and the $k_2$-th rows of $V$ simultaneously. It follows that the permutation will neither affect the product nor alter the first column of $U$ or the first row of $V$. Again, after scaling $u_{2k_2}$ to unit and renaming the permuted matrices as $U$ and $V$. Since $q_{33} = q_{22} = 0$, it must be $u_{31} = v_{22} = 0$.

Finally, since $\mathbf{p}_2$ and $\mathbf{p}_3$ are distinct (i.e., $q_{23} > 0$), there exists an index $3 \leq k_3 \leq n$ such that $u_{3k_3}v_{k_32} > 0$. Permuting the third and the $k_3$-th columns and rows and scaling $u_{3k_3}$ to unit will give rise to the structure specified in the lemma. □

It is important to note that the procedure described above cannot be continued to the fourth or other rows or columns. With this in mind, we refer to (8.3) as the *standard* nonnegative factorization of $Q(\mathbf{p}_1, \ldots, \mathbf{p}_n)$.

## 8.3 Nonnegative rank factorization for linear Euclidean distance matrices

In a recent paper [10], it is shown that for a nonnegative matrix of rank 3 to have nonnegative rank 10, we would need a matrix of order at least 252. The result is this section clearly indicates that the actual order could be much lower. We use linear EDMs to make our points.

To begin this, given a permutation $\tau$ of the set $\{1, 2, \ldots, n\}$, define the permutation matrix $P_\tau := [\delta_{i\tau(j)}]$ where $\delta_{st}$ denotes the Kronecker delta function. It is easy to see that

$$P_\tau^\top Q(\mathbf{p}_1, \ldots, \mathbf{p}_n) P_\tau = Q(\mathbf{p}_{\tau(1)}, \ldots \mathbf{p}_{\tau(n)}), \tag{8.4}$$

since the matrices $P_\tau^\top$ and $P_\tau$ permute the rows and the columns of the matrix $Q(\mathbf{p}_1, \ldots, \mathbf{p}_n)$, respective. It follows that any two EDMs constructed from the same set of $n$ given points are orthogonally equivalent to each other by permutation matrices. In particular, they have the same nonnegative rank.

In one dimensional case (i.e., $r = 1$), we may assume, after permutations if necessary, that the point are arranged in ascending order, $\mathbf{p}_1 < \ldots < \mathbf{p}_n$. Define $s_i := \mathbf{p}_{i+1} - \mathbf{p}_i$, $i = 1, \ldots, n-1$. Entries in the linear EDM are arranged in a special ordering pattern— be

complexed when radiating away from the diagonal per column and row, i.e.

$$Q(\mathbf{p}_1,\ldots,\mathbf{p}_n)=\begin{bmatrix} 0 & s_1^2 & (s_1+s_2)^2 & (s_1+s_2+s_3)^2 & \ldots \\ s_1^2 & 0 & s_2^2 & (s_2+s_3)^2 & \ldots \\ (s_1+s_2)^2 & s_2^2 & 0 & s_3^2 & \ldots \\ (s_1+s_2+s_3)^2 & (s_2+s_3)^2 & s_3^2 & 0 & \\ \vdots & & & & \end{bmatrix} \tag{8.5}$$

Such a specific ordering could provide us with an illuminating insight into the nonnegative rank of the EDM. Unless mentioned otherwise, the subsequent discussion is for the case $r = 1$. Since the reference to the points $\mathbf{p}_1,\ldots,\mathbf{p}_n$ is not crucial, for convenience, we express $Q(\mathbf{p}_1,\ldots,\mathbf{p}_n)$ as $Q_n$.

We start the analysis of the nonnegative rank condition with the case $n = 4$. This is the case where we can actually "see" the nonnegative rank. The insight thus obtained will be generalized to draw general conclusions. Denote the columns of $Q_4$ by $Q_4 = [\mathbf{q}_1,\ldots,\mathbf{q}_4]$. By Lemma 7.6.1, it suffices to consider the pull-back $\vartheta(Q_4)$ on the probability simplex $\mathcal{D}_4$. Recall that the probability simplex $\mathcal{D}_4$ can easily be represented via the unit tetrahedron $S_3$ in the first octant of $\mathbb{R}^3$. Columns of $\vartheta(Q_4)$ are represented by points $\tilde{\mathbf{q}}_1,\tilde{\mathbf{q}}_2,\tilde{\mathbf{q}}_3,\tilde{\mathbf{q}}_4$ in $S_3$. Assume the generic condition that $\text{rank}(Q_4) = 3$. Then the four points $\tilde{\mathbf{q}}_1,\tilde{\mathbf{q}}_2,\tilde{\mathbf{q}}_3,\tilde{\mathbf{q}}_4$ are coplanar.

A simple layout is sketched in Figure 8.2. It can be argued that this common plane is either parallel to the $y$-axis or has the $y$-intercept gaiven by

$$\frac{s_1 s_2}{s_1 s_2 - s_3(s_1+s_2+s_3)}$$

which is either negative or positive with value greater than 1. In any of these cases, the plane intersects the tetrahedron as a quadrilateral. The first three points reside on three separate "ridges" of the quadrilateral and hence cannot be enclosed by any triangle within the quadrilateral except the one with vertices at these three points. Clearly, if $\text{rank}_+(Q_4) < 4$, then $\tilde{\mathbf{q}}_4$ must be inside this triangle and hence be a convex combination of $\tilde{\mathbf{q}}_1,\tilde{\mathbf{q}}_2,\tilde{\mathbf{q}}_3$, which deduces that the vector $\mathbf{q}_4$ must be a nonnegative combination of $\mathbf{q}_1,\mathbf{q}_2,\mathbf{q}_3$. This, of course, is impossible because $q_{44} = 0$. Thus $\text{rank}_+(Q_4) = 4$.

Similarly, for the case $n > 4$, identify any $n$-dimensional vector $\mathbf{x} \in \mathcal{D}_n$ by its first $n-1$ entries $[x_1,\ldots,x_{n-1}]^\top$. Columns of $\vartheta(Q_n)$ are represented by $\tilde{\mathbf{q}}_1,\ldots,\tilde{\mathbf{q}}_n$ as $n$ points residing within the unit polyhedron $\mathcal{S}_{n-1}$ in the first orthotant of $\mathbb{R}^{n-1}$. The geometry related to nonnegative rank property of $Q_4$ could be generalized as follows:

**Theorem 8.3.1** *Suppose that the linear EDM $Q_n$ is of rank 3. Then* $\text{rank}_+(Q_n) = n$.

**Proof.** Because $\text{rank}(Q_n) = 3$, its columns reside on a 3-dimensional subspace of $\mathbb{R}^n$. The pull-

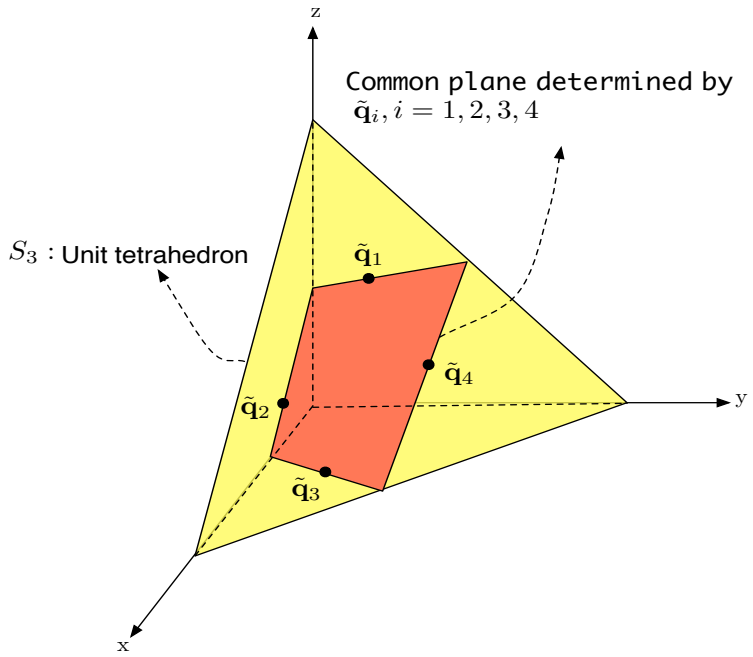


Figure 8.2: A geometric representation of the matrix $\vartheta(Q_4)$ when $r = 1$.

back map $\vartheta$ can be considered as the intersection of this subspace and the hyperplane defined by $\sum_{i=1}^{n} x_i = 1$. Columns of $\vartheta(Q_n)$ therefore are "coplanar" whereas by their common plane we refer to a 2-dimensional affine subspace in $\mathbb{R}^n$. Identifying any $n$-dimensional vector $\mathbf{x} \in \mathcal{D}_n$ by its first $n-1$ entries $[x_1, \ldots, x_{n-1}]^\top$, we thus are able to "see" columns $\vartheta(\mathbf{q}_1), \ldots, \vartheta(\mathbf{q}_n)$ as $n$ points residing within the unit polyhedron $\mathcal{S}_{n-1}$ in the first orthotant of $\mathbb{R}^{n-1}$. These points remain to be coplanar. (Indeed, the 2-dimensional affine subspace can be identified by a fixed point, say, $\vartheta(\mathbf{q}_1)$, and two coordinate axes, say, $\mathbf{v}_1 := \vartheta(\mathbf{q}_2) - \vartheta(\mathbf{q}_1)$ and $\mathbf{v}_2 := \vartheta(\mathbf{q}_3) - \vartheta(\mathbf{q}_n)$, where all points in the 2-dimensional affine subspace can be represented as $\vartheta(\mathbf{q}_1) + \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2$ with scalars $\alpha_1$ and $\alpha_2$. The drawing in Figure 8.2, therefore, is still relatively instructive.)

Now we show that each point $\tilde{\mathbf{q}}_i$ could not be a nonnegative combination of the others. For $1 \le i \le n-1$, it is clear that $\tilde{\mathbf{q}}_i$ cannot possibly be a convex combination of any other $\tilde{\mathbf{q}}_j$ because of the unique zero at its $i$th entry. We claim further that $\tilde{\mathbf{q}}_n$ cannot possibly be in the convex hull spanned by $\tilde{\mathbf{q}}_1, \ldots, \tilde{\mathbf{q}}_{n-1}$. Assume otherwise, then we would have

$$\tilde{\mathbf{q}}_n = \sum_{i=1}^{n-1} c_i \tilde{\mathbf{q}}_i$$

for some $c_i \ge 0$ with $\sum_{i=1}^{n-1} c_i = 1$. Note that $\|\tilde{\mathbf{q}}_n\|_1 = 1$. However, $\|\sum_{i=1}^{n-1} c_i \tilde{\mathbf{q}}_i\|_1 < 1$ because

Table 8.1: Standard nonnegative factorizations of $Q_4$.

| $U$ | $V$ |
|---|---|
| $\begin{bmatrix} 1 & 0 & \frac{{s_1}^2}{s_2^2} & (s_1+s_2+s_3)^2 \\ \frac{{s_2}^2}{(s_1+s_2)^2} & 1 & 0 & (s_2+s_3)^2 \\ 0 & \frac{(s_1+s_2)^2}{{s_1}^2} & 1 & {s_3}^2 \\ \frac{{s_3}^2}{(s_1+s_2)^2} & \frac{(s_1+s_2+s_3)^2}{{s_1}^2} & \frac{(s_2+s_3)^2}{{s_2}^2} & 0 \end{bmatrix}$ | $\begin{bmatrix} 0 & 0 & (s_1+s_2)^2 & 0 \\ {s_1}^2 & 0 & 0 & 0 \\ 0 & {s_2}^2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |
| $\begin{bmatrix} 1 & 0 & 0 & \frac{(s_1+s_2)s_2(s_1+s_2+s_3)}{s_2+s_3} \\ 0 & 1 & 0 & {s_2}^2 \\ 0 & \frac{s_3(s_1+s_2)}{(s_2+s_3)s_1} & 1 & 0 \\ \frac{s_3(s_2+s_3)}{(s_1+s_2)s_1} & 0 & \frac{(s_2+s_3)(s_1+s_2+s_3)}{(s_1+s_2)s_2} & 0 \end{bmatrix}$ | $\begin{bmatrix} 0 & {s_1}^2 & \frac{s_3s_1(s_1+s_2)}{s_2+s_3} & 0 \\ {s_1}^2 & 0 & 0 & \frac{s_3(s_2+s_3)s_1}{s_1+s_2} \\ \frac{(s_1+s_2)s_2(s_1+s_2+s_3)}{s_2+s_3} & {s_2}^2 & 0 & 0 \\ 0 & 0 & 1 & \frac{(s_2+s_3)(s_1+s_2+s_3)}{(s_1+s_2)s_2} \end{bmatrix}$ |
| $\begin{bmatrix} 1 & 0 & 0 & {s_1}^2 \\ \frac{s_2(s_2+s_3)}{(s_1+s_2)(s_1+s_2+s_3)} & 1 & 0 & 0 \\ 0 & \frac{s_3(s_1+s_2)}{(s_2+s_3)s_1} & 1 & 0 \\ 0 & 0 & \frac{(s_2+s_3)(s_1+s_2+s_3)}{(s_1+s_2)s_2} & \frac{s_3(s_2+s_3)s_1}{s_1+s_2} \end{bmatrix}$ | $\begin{bmatrix} 0 & 0 & \frac{s_2(s_1+s_2)(s_1+s_2+s_3)}{s_2+s_3} & (s_1+s_2+s_3)^2 \\ {s_1}^2 & 0 & 0 & \frac{s_3(s_2+s_3)s_1}{s_1+s_2} \\ \frac{s_2(s_1+s_2)(s_1+s_2+s_3)}{s_2+s_3} & {s_2}^2 & 0 & 0 \\ 0 & 1 & \frac{s_3(s_1+s_2)}{(s_2+s_3)s_1} & 0 \end{bmatrix}$ |

$\|\tilde{\mathbf{q}}_i\|_1 < 1$ after chopping away the last row of $\vartheta(Q_n)$. This is a contradiction. The smallest number of vertices for a convex hull to enclose $\tilde{\mathbf{q}}_1, \ldots, \tilde{\mathbf{q}}_n$, therefore, has to be $n$, implying that $\operatorname{rank}_+(Q_n) = n$. □

The expression $Q_4 = UV$ in the standard form (8.3) contains a polynomial system of 22 equations in 23 unknowns whereas one of the nonzero unknowns can be normalized to unit. Other than the trivial factorization $Q_4 = I_4 Q_4$ where $I_4$ is the identity matrix, it can be shown that this nonlinear system contains only three nontrivial nonnegative factorizations which we list in Table 8.1. Clearly, the first set of factorization in the table is equivalent to $Q_4 I_4$. The last two sets of factorizations correspond to the four vertices of the quadrilateral depicted in figure 8.2. This observation also shows that $Q_4$ is not prime [13, 147].

There is a subtle difference between the standard nonnegative factorization of $Q_4$ and that of $Q_n$ when $n \geq 5$. Except for the trivial factorization, both factors $U$ and $V$ in Table 8.1 for $Q_4$ are of rank 3. This is not the case in general.

**Lemma 8.3.1** *Assume that* $\operatorname{rank}(Q_n) = 3$ *and* $n \geq 5$ *and* $Q_n = UV$ *is a standard nonnegative factorization for the matrix* $Q_n$*. Then* $U$ *and* $V$ *cannot be of rank 3 simultaneously.*

**Proof.** Observe first that since $Q_n$ is symmetric, $\operatorname{rank}(Q_n) = 3$ and $n \geq 5$, we can partition $Q_n$ as

$$Q_n = \left[\begin{array}{c|c} Q_3 & Q_3\Phi \\ \hline \Phi^\top Q_3 & \Phi^\top Q_3 \Phi \end{array}\right] \tag{8.6}$$

where $\Phi \in \mathbb{R}^{3\times(n-3)}$ is uniquely determined. Indeed, if we write $\Phi = [\boldsymbol{\phi}_4, \ldots \boldsymbol{\phi}_n]$, then it can be shown that

$$\boldsymbol{\phi}_j = \begin{bmatrix} \frac{\left(\sum_{\ell=2}^{j-1} s_\ell\right)\left(\sum_{\ell=3}^{j-1} s_\ell\right)}{s_1(s_1+s_2)} \\ -\frac{\left(\sum_{\ell=1}^{j-1} s_\ell\right)\left(\sum_{\ell=3}^{j-1} s_\ell\right)}{s_1 s_2} \\ \frac{\left(\sum_{\ell=1}^{j-1} s_\ell\right)\left(\sum_{\ell=2}^{j-1} s_\ell\right)}{s_2(s_1+s_2)} \end{bmatrix}, \quad j = 4, \ldots, n. \tag{8.7}$$

Note that the second entry in $\boldsymbol{\phi}_j$ is always negative.

Assume by contradiction that both $U$ and $V$ of $Q_n$ are of rank 3. As $U$ and $V$ appear in the standard form (8.3), their $3 \times 3$ leading principal submatrices $U_{11}$ and $V_{11}$ are nonsingular.

Thus similar to (8.6), we can partition the nonnegative factors into blocks

$$Q_n = \left[\begin{array}{c|c} U_{11} & U_{11}\Theta \\ \hline \Lambda^\top U_{11} & \Lambda^\top U_{11}\Theta \end{array}\right] \left[\begin{array}{c|c} V_{11} & V_{11}\Gamma \\ \hline \Delta^\top V_{11} & \Delta^\top V_{11}\Gamma \end{array}\right], \tag{8.8}$$

where $\Theta, \Lambda, \Gamma$ and $\Delta$ are real matrices of compatible sizes. Upon comparing (8.8) with (8.6), we see that $\Lambda = \Phi = \Gamma$. Taking a closer look at the product $\Lambda^\top U_{11}$, we find that the signs of its entries are given by

$$\Lambda^\top U_{11} = \begin{bmatrix} + & - & + \\ \vdots & \vdots & \vdots \\ \\ + & - & + \end{bmatrix} \begin{bmatrix} 1 & 0 & * \\ * & 1 & 0 \\ 0 & * & 1 \end{bmatrix} = \begin{bmatrix} * & \boxdot & + \\ \vdots & \vdots & \vdots \\ \\ * & \boxdot & + \end{bmatrix},$$

where, again, $*$ indicates some undetermined nonnegative numbers, $+$ some undetermined positive numbers, and $\boxdot$ some nonnegative numbers which can further be determined. Similarly, the signs for entries of $V_{11}\Gamma$ are given by

$$V_{11}\Gamma = \begin{bmatrix} 0 & * & + \\ + & 0 & * \\ * & + & 0 \end{bmatrix} \begin{bmatrix} + & \dots & + \\ - & \dots & - \\ + & \dots & + \end{bmatrix} = \begin{bmatrix} * & \dots & * \\ + & \dots & + \\ \boxdot & \dots & \boxdot \end{bmatrix}.$$

Since the diagonal entries of $Q_n$ are zero and entries of $U$ and $V$ are nonnegative, it follows that $u_{ij}v_{ji} = 0$ for all indices $i$ and $j$. Accordingly, the $+$'s in the middle row of $V_{11}\Gamma$ must cause the $\boxdot$'s in the middle column of $\Lambda^\top U_{11}$ to become zeros. This implies that the very same $u_{32}$ would have to satisfy the equalities

$$-\frac{\left(\sum_{\ell=1}^{j-1} s_\ell\right)\left(\sum_{\ell=3}^{j-1} s_\ell\right)}{s_1 s_2} + \frac{\left(\sum_{\ell=1}^{j-1} s_\ell\right)\left(\sum_{\ell=2}^{j-1} s_\ell\right)}{s_2(s_1+s_2)} u_{32} = 0,$$

for all $j = 4, \dots n$, which is genetically impossible if $n \geq 5$. □

The computation of the nonnegative factorization of $Q_n$ for $n \geq 5$ is considerably harder. The case $n = 5$, for example, involves a polynomial system of 39 nonlinear equations in 41 unknowns two of which can be normalized. From a geometric point of view, we might want to

generate two nontrivial matrix factorization of $Q_5$ based on the factorization of $Q_4$. Write

$$Q_5 = \left[\begin{array}{c|c} Q_4 & \mathbf{c}_5 \\ \hline \mathbf{c}_5^\top & 0 \end{array}\right], \tag{8.9}$$

with $\mathbf{c}_5 \in \mathbb{R}^{4\times 1}$. Consider the submatrix $[Q_4, \mathbf{c}_5]$ only. In Theorem 8.3.1 we have shown that vectors $\tilde{\mathbf{q}}_1, \ldots, \tilde{\mathbf{q}}_5$ are coplanar and, hence, $\tilde{\mathbf{q}}_5$ is a point in the interior of the quadrilateral drawn in Figure 8.2. In particular, if $Q_4 = UV$ is one of the two nontrivial standard nonnegative factorizations of $Q_4$, i.e., columns of $\vartheta(U)$ (or $\vartheta(V^\top)$) are the four vertices of the quadrilateral, then $\mathbf{c}_5$ is a nonnegative combination of columns of $U$ (or $V^\top$). In this way, two of the nontrivial standard nonnegative factorizations of $Q_5$ are given by

$$Q_5 = \left[\begin{array}{c|c} U & \mathbf{0} \\ \hline \mathbf{0}^\top & 1 \end{array}\right] \left[\begin{array}{c|c} V & \mathbf{w}_5 \\ \hline \mathbf{c}_5^\top & 0 \end{array}\right] = \left[\begin{array}{c|c} U & \mathbf{c}_5 \\ \hline \mathbf{z}_5^\top & 0 \end{array}\right] \left[\begin{array}{c|c} V & \mathbf{0} \\ \hline \mathbf{0}^\top & 1 \end{array}\right], \tag{8.10}$$

respectively, where $\mathbf{w}_5$ and $\mathbf{z}_5$ are some nonnegative vectors satisfying $U\mathbf{w}_5 = V^\top \mathbf{z}_5 = \mathbf{c}_5$. This procedure can be generalized to higher $n$, but there might be other nonnegative factorizations which are not of this particular form specified in (8.10).

## 8.4 A conjecture for higher dimensional Euclidean distance matrices

In higher dimensional vector spaces, points $\mathbf{p}_1, \ldots, \mathbf{p}_n$ cannot be completely arranged in ascending order. Thus, for $r > 1$ and $n \geq r + 2$, the EDM will not enjoy the intrinsic structure indicated in (8.5). However, if we denote $\mathbf{p}_j = [p_{ij}]$, the matrix $Q(\mathbf{p}_1, \ldots, \mathbf{p}_n)$ could be expressed as

$$Q(\mathbf{p}_1, \ldots, \mathbf{p}_n) = \sum_{i=1}^{r} Q(p_{i1}, \ldots, p_{in}).$$

It has been shown earlier that generically $\text{rank}_+(Q(p_{i1}, \ldots, p_{in})) = n$ for each $1 \leq i \leq r$. Note that these $r$ linear EDMs are essentially independent. If we want to get a reduction of rank via the summation (of nonnegative entries) of these $r$ linear EDMs, they must satisfy some delicate algebraic constraints. We thus conjecture that $\text{rank}_+(Q(\mathbf{p}_1, \ldots, \mathbf{p}_n)) = n$ generically for all $r$.

It might be informative to reexamine the geometric representation of the matrix $Q_4$ when $r > 1$. In contrast to the setting in Figure 8.2, columns of $Q_4$ are not coplanar while $\text{rank}(Q_4) =$

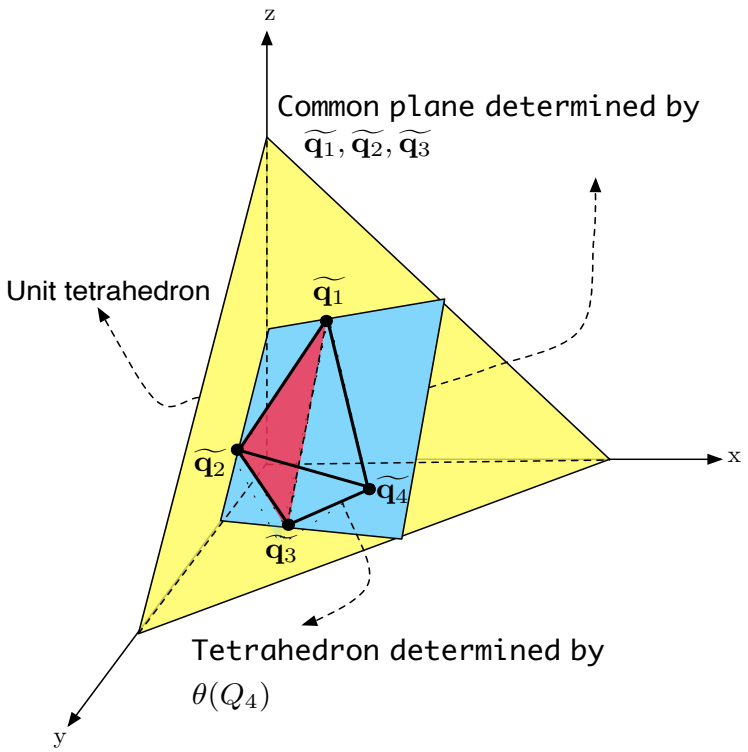


Figure 8.3: A geometric representation of the matrix $\vartheta(Q_4)$ when $r > 1$.

$r + 2 \geq 4$.

The relationship of $\tilde{\mathbf{q}}_1$, $\tilde{\mathbf{q}}_2$, $\tilde{\mathbf{q}}_3$, and $\tilde{\mathbf{q}}_4$ could be depicted as that in Figure 8.3. The vertex $\tilde{\mathbf{q}}_4$ resides on the simplex $\mathcal{D}_3$. How the base plane determined by vertices $\tilde{\mathbf{q}}_1$, $\tilde{\mathbf{q}}_2$, and $\tilde{\mathbf{q}}_3$ intersects the axes characterizes the zero structure of nonnegative factors. Different from the case $r = 1$, there are several possibilities and there is simply no general rules here. The one shown in Figure 8.3 implies that the corresponding $Q_4$ is prime, which is another interesting contrast to the case when $r = 1$.

# Chapter 9

# Future Work

In the course of our research into the inverse problem of matrix data reconstruction, we have made several major breakthroughs and resolved several problems that have been regarded as difficult in the literature. But we also have discovered many additional issues that deserve further investigation. These interesting questions arise out of either theoretical rumination or practical consideration. Some of them are serious enough that would impact various parts of our current investigation, while others are more for mathematical inquisitiveness. In this section, we list without further elaboration some of the most intriguing problems that we have identified in our study.

1. *Impact of eigenvalue structure*

   - In Chapter 2, we propose an IMP approach to the IEPs. In that setting, we point out that only the structure of eigenvalues are needed for the IMPs whereas the actual eigenvalues could be arbitrary. We have numerical evidence, but would like to investigate whether and why an IMP is solvable for arbitrary splitting in the number of real and complex eigenvalues $n = t + r$ and the distribution of prescribed eigenvectors $k = 2k_C + k_P + k_N \leq k_{max}$.
   - We point out in Section 2.5 that eigenvalues of an IMP with real, symmetric, and positive semi-definite $(M, C, K)$ must satisfy the condition that the matrix $XJ^2H^{-1}X^\top$ is negative semi-definite. In this case, the completion process for the eigenvectors and the eigenvalues cannot be done individually. Standard optimization could be employed to solve the model built in (2.31). But we have an idea that the numerical techniques introduced in Chapter 3 and 4 might also be applicable to carry out this task. We think this connection or even some theoretical generalization in this regard is worthy of future research.

2. *Expansion of library modules for IEPs via the truncated QR method*

- In Chapter 3, we have developed a software package Opt4QIEP that employs the notion of the truncated $QR$ method to solve QIEPs with sparsity pattern and inexact data. Our idea is to build a consistent linear inequality system while taking the sparsity pattern embedded in the original system directly into account in the construction process. We believe that this automated structure generation and error correction approach should have significant impact in applications and should be generalized to include as many critical dynamical systems as possible. Module by module, we continue to expand the library according to various physical laws.

3. *Improvement of IEPs via the SDP techniques for IEPs*

   - In Chapter 4, we have demonstrated how the SDP approach can be employed with little effort to tackle various very difficult structured systems IEPs. However, most current general-purpose SDP software suffers from memory limitation on handling large scale problems. The culprit is the interior point method which usually is aimed at solving the primal problem. Our QIEPs under the framework of the SDP is in the dual form. It thus is possible to develop special-purpose numerical approach, solely for the dual problem, for our QIEPs. With the power that SDP methods have already manifested, this achievement would be a boost to our ability in solving QIEPs.

4. *Vectorization and parallelization Parallel computation in solving NMF*

   - Currently our numerical procedure for solving NMF is contingent upon the calculation of the proximity which is being carried out column by column. Despite its yield of a better approximation, the employment of such a strategy make the computation less efficient and competitive than the Lee-Seung algorithm which allows BLAS3 implementation. We have already pointed out in Chapter 4 that the iterative steps are embarrassingly parallelizable, which makes the transporting to a parallel computing environment a very easy task. We are also investigating multi-tasking by vectorization within the current framework, which will allow us to employ BLAS3 implementation as well.

5. *Gneralization of the IMF*

   - The notion in our IMF formulation proposed in Chapter 6 is meant to handle discrete, multivariate datatypes in matrix form. Numerical techniques developed in this dissertation seem to be able to tackle different applications with the same stratagem. Our next approach is to generalize this method by considering *composite*-integer cases and multi-category applications.

6. *Perturbation analysis for NRF*

   Our primary goal in Chapter 7 is simply to propose a numerical procedure that might serve as a computational tool for the exploration of the very hard NRF problem. We have not studied any backward stability analysis of our algorithm nor the perturbation analysis for the general NRF. We repeat the two important questions already raised in Chapter 7.

   - Given a nonnegative matrix $A$ which has an NRF, under what condition will the perturbed nonnegative matrix $A + E$ still have an NRF?
   - Given a nonnegative matrix $A$ which has an NRF, let $U$ and $V$ be the nonnegative factors found by some numerical computation, say, our Algorithm 10, so that $UV$ is a numerical NRF of $A$. Is $UV$ the exact NRF of some perturbed nonnegative matrix $A + E$?

## REFERENCES


[1] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optim.*, 5(1):13–51, 1995.

[2] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton. Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM J. Optim.*, 8(3):746–768 (electronic), 1998.

[3] A. Antoniou and W.-S. Lu. *Practical optimization, Algorithms and engineering applications.* Springer, New York, 2007.

[4] E. Assis and V. Steffen, Jr. Inverse problem techniques for the identification of rotor-bearing systems. *Inverse Problems in Science and Engineering*, 11(1):39–53, 2003.

[5] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[6] Z.-J. Bai, D. Chu, and D. Sun. A dual optimization approach to inverse quadratic eigenvalue problems with partial eigenstructure. *SIAM J. Sci. Comput.*, 29(6):2531–2561 (electronic), 2007.

[7] I. Bárány. Sylvester's question: the probability that $n$ points are in convex position. *Ann. Probab.*, 27(4):2020–2034, 1999.

[8] F. Barioli and A. Berman. The maximal cp-rank of rank $k$ completely positive matrices. *Linear Algebra Appl.*, 363:17–33, 2003. Special issue on nonnegative matrices, $M$-matrices and their generalizations (Oberwolfach, 2000).

[9] M. Baruch. Optimization procedure to correct stiffness and flexibility matrices using vibration data. *AIAA J.*, 16:1208–1210, 1978.

[10] L. B. Beasley and T. J. Laffey. Real rank versus nonnegative rank. *Linear Algebra and its Applications*, 431(12):2330 – 2335, 2009. Special Issue in honor of Shmuel Friedland.

[11] K. P. Bennett and C. Campbell. Support vector machines: Hype or hallelujah? *SIGKDD Explorations*, 2:1–13, 2000.

[12] A. Berman and E. J. Nagy. Improvement of a large analytical model using test data. *AIAA J.*, 21:1168–1173, 1983.

[13] A. Berman and R. J. Plemmons. Matrix group monotonicity. *Proc. Amer. Math. Soc.*, 46:355–359, 1974.

[14] A. Berman and U. G. Rothblum. A note on the computation of the CP-rank. *Linear Algebra Appl.*, 419(1):1–7, 2006.

[15] A. Berman and N. Shaked-Monderer. *Completely positive matrices.* World Scientific Publishing Co. Inc., River Edge, NJ, 2003.

[16] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Comput. Statist. Data Anal.*, 52(1):155–173, 2007.

[17] D. P. Bertsekas. *Nonlinear programming (2nd)*. Belmont, MA: Athena Scientific., 1999.

[18] R. Boisvert, R. Pozo, K. Remington, B. Miller, and R. Lipman. *Matrix Market*. National Institute of Standards and Technology, 2007.

[19] D. L. Boley and G. H. Golub. A survey of matrix inverse eigenvalue problems. *Inverse Problems*, 3(4):595–622, 1987.

[20] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, 2004.

[21] M. Brinkmeier and U. Nackenhorst. An approach for large-scale gyroscopic eigenvalue problems with application to high-frequency response of rolling tires. *Computational Mechanics*, 41(4):503–515, 2008.

[22] R. Bro and S. de Jong. A fast non-negativity-constrained least squares algorithm. *Journal of Chemometrics*, 11(5):393–401, 1997.

[23] J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Sciences of the United States of America*, 101(12):4164–4169, 2004.

[24] Y.-F. Cai, Y.-C. Kuo, W.-W. Lin, and S.-F. Xu. Solutions to a quadratic inverse eigenvalue problem. *Linear Algebra Appl.*, 2008.

[25] S. L. Campbell and G. D. Poole. Computing nonnegative rank factorizations. *Linear Algebra Appl.*, 35:175–182, 1981.

[26] D. H. Carlson and H. Schneider. Inertia theorems for matrices: the semi-definite case. *Bull, Amer. Math. Soc.*, 68:479–483, 1962.

[27] J. B. Carvalho, B. N. Datta, W.-W. Lin, and C.-S. Wang. Symmetry preserving eigenvalue embedding in finite-element model updating of vibrating structures. *J. Sound Vibration*, 290(3-5):839–864, 2006.

[28] L. Chen. New analysis of the sphere covering problems and optimal polytope approximation of convex bodies. *J. Approx. Theory*, 133(1):134–145, 2005.

[29] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 2001.

[30] Z. Chen, A. Cichocki, and T. M. Rutkowski. Constrained non-negative matrix factorization method for eeg analysis in early detection of alzheimer disease. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP2006*, Toulouse, France, 2006.

[31] G. G. Chowdhury. *Introduction to modern information retrieval. 2nd ed.* facet publishing, 2004.

[32] M. T. Chu, B. Datta, W.-W. Lin, and S.-F. Xu. Spillover phenomenon in quadratic model updating. *AIAA J.*, 46(2):420–428, 2008.

[33] M. T. Chu, N. Del Buono, and B. Yu. Structured quadratic inverse eigenvalue problem. I. Serially linked systems. *SIAM J. Sci. Comput.*, 29(6):2668–2685 (electronic), 2007.

[34] M. T. Chu, F. Diele, R. J. Plemmons, and S. Ragni. Optimality, computation and interpretation of nonnegative matrix factorizations. *Available online at* `http://www4.ncsu.edu/~mtchu/Research/Papers/nnmf.ps`, 2005.

[35] M. T. Chu, R. E. Funderlic, and G. H. Golub. A rank-one reduction formula and its applications to matrix factorizations. *SIAM Rev.*, 37(4):512–530, 1995.

[36] M. T. Chu, R. E. Funderlic, and G. H. Golub. Rank modifications of semidefinite matrices associated with a secant update formula. *SIAM J. Matrix Anal. Appl.*, 20(2):428–436 (electronic), 1999.

[37] M. T. Chu and G. H. Golub. *Inverse eigenvalue problems: theory, algorithms, and applications.* Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 2005.

[38] M. T. Chu, Y.-C. Kuo, and W.-W. Lin. On inverse quadratic eigenvalue problems with partially prescribed eigenstructure. *SIAM J. Matrix Anal. Appl.*, 25(4):995–1020 (electronic), 2004.

[39] M. T. Chu and M. M. Lin. Low-dimensional polytope approximation and its applications to nonnegative matrix factorization. *SIAM J. Sci. Comput.*, 30(3):1131–1155, 2008.

[40] M. T. Chu, W.-W. Lin, and S.-F. Xu. Updating quadratic models with no spillover effect on unmeasured spectral data. *Inverse Problems*, 23(1):243–256, 2007.

[41] M. T. Chu and S.-F. Xu. Spectral decomposition of real symmetric quadratic $\lambda$-matrices and its applications. *Math. Comp.*, 78(265):293–313, 2009.

[42] K. L. Clarkson. Algorithms for polytope covering and approximation. In *Algorithms and data structures (Montreal, PQ, 1993)*, volume 709 of *Lecture Notes in Comput. Sci.*, pages 246–252. Springer, Berlin, 1993.

[43] R. E. Cline and R. E. Funderlic. The rank of a difference of matrices and associated generalized inverses. *Linear Algebra Appl.*, 24:185–215, 1979.

[44] J. E. Cohen and U. G. Rothblum. Nonnegative ranks, decompositions, and factorizations of nonnegative matrices. *Linear Algebra Appl.*, 190:149–168, 1993.

[45] A. L. Comrey and H. B. Lee. *A First Course in Factor Analysis.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1992.

[46] J. H. Conway and N. J. A. Sloane. *Sphere packings, lattices, and groups, 3rd ed.* Springer, 1999.

[47] T. M. Cover and J. A. Thomas. *Elements of information theory.* Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, second edition, 2006.

[48] G. M. Crippen and T. F. Havel. *Distance geometry and molecular conformation*, volume 15 of *Chemometrics Series.* Research Studies Press Ltd., Chichester, 1988.

[49] B. N. Datta. Finite element model updating, eigenstructure assignment and eigenvalue embedding techniques for vibrating systems. *Mech. Sys. Signal Processing*, Special Volumn on "Vibration Control", 16:83–96, 2002.

[50] B. N. Datta. Finite element model updating and partial eigenvalue assignment in structural dynamics: recent developments on computational methods. In *Proceedings: 10th International Conference "Mathematical Modelling and Analysis 2005" and 2nd International Conference "Computational Methods in Applied Mathematics"*, pages 15–27. Technika, Vilnius, 2005.

[51] B. N. Datta, S. Elhay, Y. M. Ram, and D. R. Sarkissian. Partial eigenstructure assignment for the quadratic pencil. *J. Sound Vibration*, 230(1):101–110, 2000.

[52] B. N. Datta and D. R. Sarkissian. Theory and computations of some inverse eigenvalue problems for the quadratic pencil. In *Structured matrices in mathematics, computer science, and engineering, I (Boulder, CO, 1999)*, volume 280 of *Contemp. Math.*, pages 221–240. Amer. Math. Soc., Providence, RI, 2001.

[53] J. de Leeuw and W. Heiser. Theory of multidimensional scaling. In *Classification, pattern recognition and reduction of dimensionality*, volume 2 of *Handbook of Statist.*, pages 285–316. North-Holland, Amsterdam, 1982.

[54] C. Ding, X. He, and H. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the Fifth SIAM International Conference on Data Mining*, Newport Beach, CA, 2005.

[55] B. Dong, M. M. Lin, and M. T. Chu. Parameter reconstruction of vibration systems from partial eigeninformation. preprint, North Carolina State University, Raleigh, North Carolina, 2009.

[56] D. Donoho and V. Stodden. When does nonnegative matrix factorization give a correct decomposition into parts? In *Proceedings of 17th Annual Conference Neural Information Processing Systems*, NIPS, Stanford University, Stanford, CA, 2003, 2003.

[57] M. Dugas, S. Merk, S. Breit, and P. Dirschedl. mdclust–exploratory microarray analysis by multidimensional clustering. *Bioinformatics*, 20(6):931–936, 2004.

[58] L. Elsner, R. Nabben, and M. Neumann. Orthogonal bases that lead to symmetric nonnegative matrices. *Linear Algebra Appl.*, 271:323–343, 1998.

[59] K. Elssel and H. Voss. Reducing huge gyroscopic eigenproblems by automated multilevel substructuring. *Archive of Applied Mechanics (Ingenieur Archiv)*, 76(3–4):171–179, 2 2006.

[60] K. Fan, I. Glicksberg, and A. J. Hoffman. Systems of inequalities involving convex functions. *Proc. Amer. Math. Soc.*, 8:617–622, 1957.

[61] M. I. Friswell, D. J. Inman, and D. F. Pilkey. The direct updating of damping and stiffness matrices. *AIAA I.*, 36:491–493, 1998.

[62] M. I. Friswell and J. E. Mottershead. *Finite element model updating in structural dynamics*, volume 38 of *Solid Mechanics and its Applications*. Kluwer Academic Publishers Group, Dordrecht, 1995.

[63] F. R. Gantmacher. *The theory of matrices. Vol. 1.* AMS Chelsea Publishing, Providence, RI, 1998. Translated from the Russian by K. A. Hirsch, Reprint of the 1959 translation.

[64] Y. Gao and G. Church. Improving molecular cancer class discovery through sparse nonnegative matrix factorization. *Bioinformatics*, 21(21):3970–3975, 2005.

[65] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: an SQP algorithm for large-scale constrained optimization. *SIAM Rev.*, 47(1):99–131 (electronic), 2005. See also `http://ccom.ucsd.edu/~peg/Software.html`.

[66] G. M. L. Gladwell. The inverse mode problem for lumped-mass systems. *Quart. J. Mech. Appl. Math.*, 39(2):297–307, 1986.

[67] G. M. L. Gladwell. Inverse vibration problems for finite-element models. *Inverse Problems*, 13(2):311–322, 1997.

[68] G. M. L. Gladwell. *Inverse problems in vibration*, volume 119 of *Solid Mechanics and its Applications*. Kluwer Academic Publishers, Dordrecht, second edition, 2004.

[69] I. Gohberg, P. Lancaster, and L. Rodman. Spectral analysis of selfadjoint matrix polynomials. *The Annals of Mathematics*, 112(1):33–71, 1980.

[70] I. Gohberg, P. Lancaster, and L. Rodman. *Matrix polynomials*. Academic Press Inc. [Harcourt Brace Jovanovich Publishers], New York, 1982. Computer Science and Applied Mathematics.

[71] G. H. Golub. Some modified matrix eigenvalue problems. *SIAM Review*, 15(2):318–334, 1973.

[72] G. H. Golub and C. F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[73] C. Gordon, D. L. Webb, and S. Wolpert. One cannot hear the shape of a drum. *Bull. Amer. Math. Soc. (N.S.)*, 27(1):134–138, 1992.

[74] J. C. Gower. Euclidean distance geometry. *Math. Sci.*, 7(1):1–14, 1982.

[75] R. M. Gray. Vector quantization. *ASSP Magazine, IEEE*, 1:4–29, 1984.

[76] D. A. Gregory and N. J. Pullman. Semiring rank: Boolean rank and nonnegative rank factorizations. *J. Combin. Inform. System Sci.*, 8(3):223–233, 1983.

[77] P. M. Gruber. Volume approximation of convex bodies by inscribed polytopes. *Math. Ann.*, 281(2):229–245, 1988.

[78] P. M. Gruber. Volume approximation of convex bodies by circumscribed polytopes. In *Applied geometry and discrete mathematics*, volume 4 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 309–317. Amer. Math. Soc., Providence, RI, 1991.

[79] V. Guillemin and A. Pollack. *Differential topology.* Prentice-Hall Inc., Englewood Cliffs, N.J., 1974.

[80] L. Guttman. General theory and methods for matric factoring. *Psychometrika*, 9(1):1–16, 1944.

[81] L. Guttman. Multiple group methods for common-factor analysis: Their basis, computation, and interpretation. *Psychometrika*, 17(2):209–222, 1952.

[82] L. Guttman. A necessary and sufficient formula for matrix factoring. *Psychometrika*, 22(1):79–81, 1957.

[83] P. Hall and B. A. Turlach. On the estimation of a convex set with corners. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 21:225–234, 1999.

[84] J. Hannah and T. J. Laffey. Nonnegative factorization of completely positive matrices. *Linear Algebra Appl.*, 55:1–9, 1983.

[85] H. H. Harman. *Modern Factor Analysis.* University of Chicago Press, Chicago, 1976.

[86] D. Hershkowitz. Existence of matrices satisfying prescribed conditions. *Masters thesis, Technion, Haifa*, 1978.

[87] D. Hershkowitz. Existence of matrices with prescribed eigenvalues and entries. *Linear and Multilinear Algebra*, 14(4):315–342, 1983.

[88] J. Hipp, U. Güntzer, and G. Nakhaeizadeh. Algorithms for association rule mining — a general survey and comparison. *SIGKDD Explor. Newsl.*, 2(1):58–64, 2000.

[89] J.-B. Hiriart-Urruty and D. Ye. Sensitivity analysis of all eigenvalues of a symmetric matrix. *Numerische Mathematik*, 70(1):45–72, 1995.

[90] P. K. Hopke. *Receptor Modeling in Environmental Chemistry.* Wiley-Interscience, New York, 1985.

[91] P. K. Hopke. *Receptor Modeling for Air Quality Management.* Elsevier, Amsterdam, 1991.

[92] A. Horn. Doubly stochastic matrices and the diagonal of a rotation matrix. *Amer. J. Math.*, 76:620–630, 1954.

[93] P. Horst. *Factor Analysis of Data Matrices*. Holt, Rinehart and Winston, New York, 1965.

[94] A. S. Householder. *The theory of matrices in numerical analysis*. Dover Publications Inc., New York, 1975. Reprint of 1964 edition.

[95] P. O. Hoyer. Nonnegative sparse coding. In *Proceedings of IEEE Workshop Neural Networks for Signal Processing*, Martigny, 2002.

[96] P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469, 2004.

[97] L. Hubert, J. Meulman, and W. Heiser. Two purposes for matrix factorization: a historical appraisal. *SIAM Rev.*, 42(1):68–82 (electronic), 2000.

[98] Z. Hung. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, Dept. of Computer Science, The University of British Columbia, Canada, pp. 18.*, 1997.

[99] K. D. Ikramov and V. N. Chugunov. Inverse matrix eigenvalue problems. *J. Math. Sci. (New York)*, 98(1):51–136, 2000. Algebra, 9.

[100] S. K. Jain and J. Tynan. Nonnegative rank factorization of a nonnegative matrix $A$ with $A^\dagger A \geq 0$. *Linear Multilinear Algebra*, 51(1):83–95, 2003.

[101] M. W. Jeter and W. C. Pye. A note on nonnegative rank factorizations. *Linear Algebra Appl.*, 38:171–173, 1981.

[102] M. W. Jeter and W. C. Pye. Some nonnegative matrices without nonnegative rank factorizations. *Indust. Math.*, 32(1):37–41, 1982.

[103] D. Johnson. *Advanced structural mechanics: an introduction to continuum mechanics and structural dynamics*. Thomas Telford Ltd., London, 2000.

[104] M. Kac. Can one hear the shape of a drum? *Amer. Math. Monthly*, 73(4, part II):1–23, 1966.

[105] T. Kawamoto, K. Hotta, T. Mishima, J. Fujiki, M. Tanaka, and T. Kurita. Estimation of single tones from chord sounds using non-negative matrix factorization. *Neural Network World*, 3:429–436, 2000.

[106] E. Kim, P. K. Hopke, and E. S. Edgerton. Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. *SIAM J. Matrix Anal. Appl.*, 30(2):713–730, 2008.

[107] H. Kim and H. Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics*, 23(12):1495–1502, 2007.

[108] H. Kim and H. Park. Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method. *SIAM J. Matrix Anal. Appl.*, 30(2):713–730, 2008.

[109] D. A. Klain and G.-C. Rota. *Introduction to geometric probability*. Lezioni Lincee. [Lincei Lectures]. Cambridge University Press, Cambridge, 1997.

[110] M. Koyutürk and A. Grama. Proximus: a framework for analyzing very high dimensional discrete-attributed datasets. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 147–156, New York, NY, USA, 2003. ACM.

[111] M. Koyutürk, A. Grama, and N. Ramakrishnan. Nonorthogonal decomposition of binary matrices for bounded-error data compression and analysis. *ACM Trans. Math. Software*, 32(1):33–69, 2006.

[112] Y.-C. Kuo, W.-W. Lin, and S.-F. Xu. New methods for finite element model updating problems. *AIAA J.*, 44:1310–1316, 2006.

[113] Y.-C. Kuo, W.-W. Lin, and S.-F. Xu. Solutions of the partially described inverse quadratic eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 29(1):33–53 (electronic), 2006/07.

[114] P. Lancaster. Inverse spectral problems for semisimple damped vibrating systems. *SIAM J. Matrix Anal. Appl.*, 29(1):279–301 (electronic), 2006/07.

[115] P. Lancaster. Model-updating for self-adjoint quadratic eigenvalue problems. *Linear Algebra Appl.*, 428(11-12):2778–2790, 2008.

[116] P. Lancaster. Model-updating for self-adjoint quadratic eigenvalue problems. *Linear Algebra Appl.*, 428(11-12):2778–2790, 2008.

[117] P. Lancaster and U. Prells. Inverse problems for damped vibrating systems. *J. Sound Vibration*, 283(3-5):891–914, 2005.

[118] C. L. Lawson and R. J. Hanson. *Solving least squares problems*, volume 15 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995. Revised reprint of the 1974 original.

[119] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[120] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, volume 13, pages 556–562, 2001.

[121] E. Lee, C. Chun, and P. Paatero. Application of positive matrix factorization in source apportionment of particulate pollutants. *Atmospheric Environment*, 33:3201–3212, 1999.

[122] B. Levin. On calculating maximum rank one underapproximations for positive arrays. *Columbia University Biostatistics Technical Report Series. Working Paper 10. http://biostats.bepress.com/columbiabiostat/papers/art10*, 1985.

[123] M. M. Lin, B. Dong, and M. T. Chu. Semi-definite programming techniques for structured quadratic inverse eigenvalue problems. preprint, North Carolina State University, Raleigh, North Carolina, 2009.

[124] W. Liu and J. Yi. Existing and new algorithms for nonnegative matrix factorization. Technical report, Department of Computer Sciences, University of Texas at Austin, 2003.

[125] J. Lofberg. YALMIP: a toolbox for modeling and optimization in MATLAB. In *2004 IEEE International Symposium on Computer Aided Control Systems Design*, pages 284–289, Taipei, Taiwan, 2004. See also `http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php`.

[126] Z.-Q. Luo, J. F. S. Sturm, and S. Zhang. Conic convex programming and self-dual embedding. *Optimization Methods and Software*, 14(3):169–218, 2000.

[127] J. Macqueen. Some methods for classification and analysis of multivariate observations. In *In Proceedings of the 5th Berkeley Symposium, vol. 1. 281297.*, 1967.

[128] O. Mangasarian and P. Bradley. k-plane clustering. *Journal of Global Optimization*, 16(1):23–32, 2000.

[129] R. Mathar. The best euclidian fit to a given distance matrix in prescribed dimensions. *Linear Algebra and its Applications*, 67:1 – 6, 1985.

[130] J. Meulman. *A distance approach to nonlinear multivariate analysis.* DSWO Press, Leiden, Netherlands, 1986.

[131] B. Mohar and P. Svatopluk. Eigenvalues in combinatorial optimization. *In Combinatorial and Graph Theoretic Problems in Linear Algebra, R. Brualdi, S. Friedland, and V. Klee, eds. Springer-Verlag, New York, Berlin*, 50:107–151, 1993.

[132] S. D. Morgan. Cluster analysis in electronic manufacturing. *Ph.D. dissertation, North Carolina State University, Raleigh, NC 27695.*, 2001.

[133] J. E. Mottershead and Y. M. Ram. Inverse eigenvalue problems in vibration absorption: Passive modification and active control. *Mechanical Systems and Signal Processing*, 20(1):5 – 44, 2006.

[134] Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13 of *SIAM Studies in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1994.

[135] N. K. Nichols and J. Kautsky. Robust eigenstructure assignment in quadratic matrix polynomials: nonsingular case. *SIAM J. Matrix Anal. Appl.*, 23(1):77–102 (electronic), 2001.

[136] M. L. Overton. Large-scale optimization of eigenvalues. *SIAM Journal on Optimization*, 2(1):88–120, 1992.

[137] P. Paatero. Least squares formulation of robust non-negative factor analysis. *Chemometrics and Intelligent Laboratory Systems*, 37:23–35, 1997.

[138] P. Paatero. The multilinear engine–a table-driven, least squares program for solving multilinear problems, including the n-way parallel factor analysis model. *J. of Computational and Graphical Statistics*, 8(4):854–888, 1999.

[139] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

[140] A. Pascual-Montano, J. M. Carzzo, K. Lochi, D. Lehmann, and R. D. Pascual-Marqui. Nonsmooth nonnegative matrix factorization (nsnmf). *IEEE Transactions on, Pattern Analysis and Machine Intelligence*, 28:403–415, 2006.

[141] R. E. Pfiefer. The historical development of j. j. sylvester's four point problem. *Mathematics Magazine*, 62(5):309–317, 1989.

[142] J. Piper, P. Pauca, R. Plemmons, and M. Giffin. Object characterization from spectral data using nonnegative factorization and information theory. In *Proceedings of AMOS Technical Conference*, 2004.

[143] J. Qian and W.-W. Lin. A numerical method for quadratic eigenvalue problems of gyroscopic systems. *J. Sound Vibration*, 306(1-2):284–296, 2007.

[144] Y. M. Ram and S. Elhay. An inverse eigenvalue problem for the symmetric tridiagonal quadratic pencil with application to damped oscillatory systems. *SIAM J. Appl. Math.*, 56(1):232–244, 1996.

[145] Y. M. Ram and I. Elishakoff. Reconstructing the cross-sectional area of an axially vibrating nonuniform rod from one of its mode shapes. *Proc. R. Soc. Lond.*, A460:1583–1596, 2004.

[146] Y. M. Ram and G. M. L. Gladwell. Constructing a finite element model of a vibratory rod from eigendata. *Journal of Sound and Vibration*, 169(2):229 – 237, 1994.

[147] D. J. Richman and H. Schneider. Primes in the semigroup of non-negative matrices. *Linear and Multilinear Algebra*, 2:135–140, 1974.

[148] K. Sekitani and Y. Yamamoto. A recursive algorithm for finding the minimum norm point in a polytope and a pair of closest points in two polytopes. *Math. Programming*, 61(2, Ser. A):233–249, 1993.

[149] N. Shaked-Monderer. Minimal cp-matrices. *ELA*, 8:140–157, 2001.

[150] N. Shaked-Monderer. A note on the cp-rank of matrices generated by a soules matrix. *ELA*, 12:2–5, 2004.

[151] J. Shen and G. Isral. A receptor model using a specific non-negative transformation technique for ambient aerosol. *Atmospheric Environment (1967)*, 23(10):2289 – 2298, 1989.

[152] S. Sra and I. S. Dhillon. Nonnegative matrix approximation: Algorithms and applications. Technical report, Deptartment of Computer Sciences, University of Texas at Austin, 2006.

[153] L. Starek and D. J. Inman. A symmetric inverse vibration problem for nonproportional underdamped systems. *Trans. ASME J. Appl. Mech.*, 64(3):601–605, 1997.

[154] J. Sturm. *SeDuMi*. Advanced Optimization Laboratory, McMaster University. Available at `http://sedume.mcmaster.ca`.

[155] J. J. Sylvester. On a special class of questions on the theory of probabilities. *Birmingham British Assoc. Rept.*, pages 8–9, 1865.

[156] L. B. Thomas. Solution to problem 73-14: Rank factorization of nonnegative matrices by a. berman and r. j. plemmons. *SIAM Review*, 16(3):393–394, 1974.

[157] F. Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43(2):235–286 (electronic), 2001.

[158] M. J. Todd. Semidefinite optimization. *Acta Numer.*, 10:515–560, 2001.

[159] R. H. Tütüncü, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Math. Program.*, 95(2, Ser. B):189–217, 2003. See also `http://www.math.nus.edu.sg/~mattohkc/sdpt3.html`.

[160] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[161] s. A. Vavasis. On the complexity of nonnegative matrix factorization, 2007.

[162] J. von Neumann and O. Morgenstern. *Theory of games and economic behavior*. Princeton University Press, Princeton, NJ, anniversary edition, 2007. With an introduction by Harold W. Kuhn and an afterword by Ariel Rubinstein.

[163] J. H. M. Wedderburn. *Lectures on matrices*. Dover Publications Inc., New York, 1964.

[164] F.-S. Wei. Mass and stiffness interaction effects in analytical model modification. *AIAA J.*, 28:1686–1688, 1990.

[165] H. K. Wimmer. Inertia theorems for matrices, controllability, and linear vibrations. *Linear Algebra and Appl.*, 8:337–343, 1974.

[166] P. Wolfe. Finding the nearest point in a polytope. *Math. Programming*, 11(2):128–149, 1976.

[167] H. Wolkowicz. *Bibliography on semidefinite programming*. Department of Combinatorics and Optimization, University of Waterloo. `http://liinwww.ira.uka.de/bibliography/Math/psd.html`.

[168] H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors. *Handbook of semidefinite programming*. International Series in Operations Research & Management Science, 27. Kluwer Academic Publishers, Boston, MA, 2000. Theory, algorithms, and applications.

[169] W. Xu, X. Liu, and Y. Gong. Document-clustering based on non-negative matrix factorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference*, pages 267–273, 2003.

[170] J. Ye. Generalized low rank approximations of matrices. *Mach. Learn.*, 61(1-3):167–191, 2005.

[171] Y. Yuan. A model updating method for undamped structural systems. *J. Comput. Appl. Math.*, 219(1):294–301, 2008.

[172] W.-X. Zhong. *Duality system in applied mechanics and optimal control*. Advances in Mechanics and Mathematics. Kluwer Academic Publishers, Boston, 2004.

# APPENDIX

# Appendix A

# QR Factorization with Column Pivoting

In this appendix we briefly review some basic concepts of QR factorization. Numerical details of computing QR factorization are referred to [72]. Consider a matrix $A \in \mathbb{R}^{\mathtt{m} \times \mathtt{n}}$, with $m \geq n$, factorized into

$$A = QR \tag{A.1}$$

where $Q \in \mathbb{R}^{m \times m}$ is an orthogonal matrix and $R \in \mathbb{R}^{m \times n}$ is an upper trapezoidal matrix with zero rows from the $n+1$-th row on. This factorization is called the *QR factorization* of the matrix $A$.

If $\text{rank}(A) = n$, the first $n$ columns of $Q$ form an orthonormal basis for the vector space $\text{range}(A)$. Then, the construction of the QR factorization could be thought of as finding out the orthonormal basis for $\text{range}(A)$. On the other hand, if $\text{rank}(A) = s < n$, the QR factorization does not guarantee to produce an orthonormal basis for $\text{range}(A)$. For example, consider the QR factorization of the matrix $A$ such that

$$A = [\mathbf{q}_1 \quad \mathbf{q}_2 \quad \mathbf{q}_3] \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.2}$$

Then $\text{rank}(A) = 2$ but $\text{range}(A)$ is not equal to $\text{span}\{\mathbf{q}_1, \mathbf{q}_2\}$, $\text{span}\{\mathbf{q}_1, \mathbf{q}_3\}$, or $\text{span}\{\mathbf{q}_2, \mathbf{q}_3\}$

**Algorithm 11**: QR factorization with column pivoting

**Input**: Matrix $A \in \mathbb{R}^{m\times n}$ and $\text{rank}(A) = s$

**Output**: Orthogonal matrix $Q \in \mathbb{R}^{m\times m}$, upper trapezoidal matrix $R \in \mathbf{R}^{m\times n}$, and permutation matrix $P \in \mathbf{R}^{n\times n}$

11.1 **begin**

11.2 Initialize $Q = I$, $R = A = R_{22}$, $P = I$

11.3 **for** $i \leftarrow 1$ **to** $s$ **do**

11.4 Let $p$ be the smallest index of the column of $R_{22}$ with the greatest 2-norm

11.5 Swap the columns $i$ and $i + p - 1$ in $R$ and in $P$

11.6 Employ a suitable transformation $H_i$, for example Householder matrices or Givens rotations, that triangularizes the first $i$ columns of $R$ such that

$$R = \begin{array}{c} \\ \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & R_{22} \end{array}\right] \\ \begin{array}{cc} i & m-i \end{array} \end{array} \begin{array}{c} i \\ n-i \end{array}$$

11.7 $Q \leftarrow QQ_i^\top$, $R \leftarrow Q_i^\top R$

11.8 **end**

11.9 **end**

[72]. However, we still can obtain a factorization of the form

$$Q^\top AP = \begin{array}{c} \\ \left[\begin{array}{cc} R_{11} & R_{12} \\ 0 & 0 \end{array}\right] \\ \begin{array}{cc} s & n-s \end{array} \end{array} \begin{array}{c} s \\ m-s \end{array} , \tag{A.3}$$

where $Q \in \mathbb{R}^{m\times m}$ is orthogonal and $R_{11} \in \mathbb{R}^{s\times s}$ is a nonsingular upper triangular matrix with diagonal elements arranged in terms of non-increasing absolute values by the permutation matrix $P$. This factorization is called the *QR factorization with column pivoting* of the matrix $A$. This factorization can be constructed by Algorithm 11.

With this in mind, we can generate a reduced version of the QR factorization (A.3) as follows:

**Remark A.0.1** *Assume $A \in \mathbb{R}^{m\times n}$, with $m \geq n$, and* $\text{rank}(A) = s < n$. *Then formula (A.3) of the QR factorization with column pivoting of $A$ could be rewritten as*

$$A = Q_1 \left[\begin{array}{cc} R_{11} & R_{22} \end{array}\right] \tag{A.4}$$

*where $Q_1 = Q(1 : m, 1 : s)$ has orthonormal vector columns. We call this reduced factorization of $A$* compact QR factorization *of $A$.*

It is easy to know that Algorithm 11 can also be used while $\text{rank}(A) = n$. In such case,

$$Q^\top AP = \begin{array}{c} \left[ \begin{array}{c} R_{11} \\ 0 \end{array} \right] \\ \begin{array}{c} n \\ m \end{array} \end{array} \begin{array}{c} n \\ m-n \end{array} , \tag{A.5}$$

where $Q \in \mathbb{R}^{m \times m}$ is orthogonal and $R_{11} \in \mathbb{R}^{s \times s}$ is a nonsingular upper triangular matrix with diagonal elements arranged in terms of non-increasing absolute values by the permutation matrix $P$.