Nonlinear Eigenvector Algorithms for Generalized Rayleigh Quotient Optimizations and Applications

By

DONG MIN ROH
DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Applied Mathematics

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Zhaojun Bai, Chair

Rishidev Chaudhuri

Shiqian Ma

Committee in Charge

2023

To my family.

Contents

Dong Min Roh, 915520552

Abstract

The dissertation focuses on nonlinear eigenvector algorithms for generalized Rayleigh quotient optimizations and their applications. A well-known result in linear algebra and optimization is the connection between the Rayleigh quotient optimization and a linear eigenvalue problem. Consequently, linear eigensolvers can solve the Rayleigh quotient optimization. Another known result is the characterization of the trace ratio optimization by an eigenvector-dependent nonlinear eigenvalue problem (NEPv). The NEPv formulation of the trace ratio optimization can be solved using the self-consistent field (SCF) iteration. However, in the case of addressing the generalized variants of the Rayleigh quotient optimization, it remains an active area of research. We explore the nonlinear generalization of the Rayleigh quotient optimization in the context of the robust common spatial pattern, an algorithm used for signal processing in brain-computer interface system. Within this framework, the nonlinear Rayleigh quotient optimization is associated with a NEPv. We propose to solve this NEPv using the SCF iteration. The numerical advantages of this approach over existing methods are demonstrated using real-world datasets. Afterwards, we discuss Wasserstein discriminant analysis, a bi-level optimization for dimensionality reduction that is formulated as a nonlinear trace ratio optimization. We present an eigenvector algorithm for Wasserstein discriminant analysis that leverages the NEPv formulations of the inner and outer optimizations. We demonstrate convergence and scalability of our proposed eigenvector algorithm for Wasserstein discriminant analysis.

Acknowledgments

First and foremost, I offer my sincere gratitude to my advisor, Professor Zhaojun Bai, for his unwavering support, guidance, and teachings throughout our countless number of meetings in the past several years. Without his invaluable role as an advisor, the completion of my doctorate degree would not have been possible. I am especially grateful for his patience and his belief in my capability, which have contributed significantly to my growth as a scientific researcher.

In addition to Professor Zhaojun Bai, I would like to express my gratitude to Professor Rishidev Chaudhuri and Professor Shiqian Ma for their invaluable service on my dissertation committee. I am truly thankful for their time and the valuable feedback they provided.

I extend my heartfelt gratitude to numerous individuals who have made a positive impact on my journey. I am immensely thankful to the Delta Modeling group at the California Department of Water Resources for their generous financial support over the past year and a half, and for providing me with the opportunity to participate in exciting projects. A special thanks goes to Dr. Minxue (Kevin) He for his guidance and suggestions that have shaped the direction of these projects.

I am also deeply grateful for the exceptional staff members in our department. I extend my sincere thanks to Tina Denena, who welcomed me with open arms from day one and has always been there to offer support whenever I needed it.

My PhD colleagues and friends have provided me with countless number of wonderful memories, and I consider myself blessed to have known each and every one of them. To Wencin Poh, Haolin Chen, and Austin Tran, thank you for being my gym buddies during the challenging times of the pandemic. I am also extremely grateful for the friendships I've cultivated with Joseph Pappe and Xiaotie (Jessie) Chen from the beginning to the end of my PhD career. Thank you for embracing my silliness.

Lastly, I want to express my deepest appreciation to my family from the bottom of my heart. Mom, dad, and brother, you have been my unwavering pillars of support. I would not have achieved this milestone without your unwavering care and love.

CHAPTER 1

Introduction

1.1. Motivation and Background

Central to many machine learning problems is the concept of optimizing the contrast between two quantities, which often leads to the optimization problem of the Rayleigh quotient or its generalized variants. Classical examples of this concept include principal component analysis and Fisher's linear discriminant analysis [**48**]. Solving the Rayleigh quotient optimization is often relied on the fundamental result that its solution corresponds to an eigenvector of a linear eigenvalue problem [**36**,**49**,**84**,**101**]. Consequently, the solution of the Rayleigh quotient optimization is found by linear eigensolvers. Meanwhile, the trace ratio optimization can be addressed by its equivalent eigenvector-dependent nonlinear eigenvalue problem formulation [**124**, **126**]. This formulation is efficiently solved using the self-consistent field iteration [**25**, **125**].

Addressing the nonlinear generalizations of the Rayleigh quotient optimization and the trace ratio optimization, however, remains an active area of research. The application of the nonlinear Rayleigh quotient optimization includes the robust Rayleigh quotient optimization for modeling data uncertainty, such as robust Fisher LDA [**61**], robust generalized eigenvalue classifier [**119**], and robust common spatial pattern [**59**]. The nonlinear trace ratio optimization arises in Wasserstein discriminant analysis [**45**, **72**, **90**], a bi-level optimization for dimensionality reduction.

From a linear algebra perspective, the discussion of these nonlinear generalizations revolves around the eigenvector-dependent nonlinear eigenvalue problem (NEPv) in which the matrices depend nonlinearly on the eigenvector(s). NEPv arises in various engineering domains, such as the Kohn-Sham equation of density functional theory in electronic structure calculations [**29**, **77**, **92**, **105**] and the Gross-Pitaevskii equation in modeling particles in Bose-Einstein condensates [**12**,**54**]. NEPv also emerges in many machine learning problems, including dimensionality reduction

[**30**,**48**,**115**,**120**,**124**], spectral clustering [**5**,**109**,**110**], and applications of robust Rayleigh quotient optimization [**11**].

The self-consistent field (SCF) iteration is the most commonly used algorithm to solve NEPv [**9**, **10**, **25**, **26**, **27**, **28**, **73**, **74**, **111**, **121**, **122**]. Some of the applications for which it is prominently applied includes the Kohn-Sham equation [**77**, **92**], the trace ratio optimization [**115**, **124**], and robust Rayleigh quotient optimization [**11**]. In essence, the SCF iteration iteratively solves the linear eigenvalue problems that arise from fixing the eigenvector dependence at the current iterate. The SCF iteration can be efficiently implemented by the use of fast Krylov subspace methods such as the implicitly restarted Arnoldi algorithm [**100**] for solving the linear eigenvalue problems.

1.2. Contributions and Organization

The dissertation is organized as follows. In Chapter 2, we review the Rayleigh quotient optimization and the trace ratio optimization, along with their respective linear eigenvalue problem and NEPv formulations. Moreover, we discuss the Dinkelbach's algorithm for tackling the trace ratio optimization. In Chapter 3, we extend the Rayleigh quotient optimization to a nonlinear setting and demonstrate its occurrence in a specific application called the robust common spatial pattern, a signal processing algorithm in brain-computer interface system. Additionally, we provide the NEPv formulation of the robust common spatial pattern. In Chapter 4, we present an algorithm based on the SCF iteration for solving this NEPv formulation. Extensive numerical results supplement the proposed algorithm. Chapters 5 and 6 are based on the preprint [**11**], co-authored with Prof. Zhaojun Bai and Prof. Ren-Cang Li. In Chapter 5, we introduce Wasserstein discriminant analysis, a bi-level optimization for dimensionality reduction which can consider both global and local interconnections between data classes. We show that Wasserstein discriminant analysis is formulated as a nonlinear trace ratio optimization. In Chapter 6, we propose an eigenvector algorithm to solve Wasserstein discriminant analysis. The proposed algorithm leverages the NEPv formulations of the inner and outer optimizations of Wasserstein discriminant analysis. In addition, the convergence analysis as well as the scalability of the proposed algorithm are presented. In Chapter 7, we conclude the dissertation and address remaining open questions. The dissertation also includes two appendices: Appendix A discusses vector and matrix calculus that are primarily utilized in Chapter 3 and Appendix B introduces the Dinkelbach's algorithm, which is applied to the trace ratio optimization in Chapter 2.

My own contributions to this dissertation include:

- A detailed derivation and explicit formulations of the eigenvector-dependent nonlinear eigenvalue problems for the robust common spatial pattern (Section 3.4). Extensive numerical results on convergence behavior and classification rate of the SCF-based algorithm using real-world data (Section 4.2).
- Presenting a novel eigenvector algorithm for Wasserstein discriminant analysis (Section 6.2), along with numerical results showcasing its convergence, classification performance, and scalability (Section 6.3).

1.3. Notation

$\mathbb{R}^n$ and $\mathbb{R}^{n\times m}$ denote the sets of all real vectors of size n and all $n\times m$ real matrices, respectively. With the addition of the '+' in the subscript, $\mathbb{R}^n_+$ and $\mathbb{R}^{n\times m}_+$ represent the sets of vectors and matrices, respectively, whose components are non-negative. Capital letters (e.g., A, B) are reserved for matrices while lower-case letters (e.g., x) are reserved for vectors.

$1_n \in \mathbb{R}^n$ denotes the vector whose components are all ones, and $I_p \in \mathbb{R}^{p\times p}$ denotes the identity matrix. δ_{ij} denotes the Kronecker delta, i.e., $\delta_{ij} = 1$ only if $i = j$ and $\delta_{ij} = 0$ otherwise. The notation $\cdot^T$ denotes the transpose of a vector or a matrix, while $\cdot^{-1}$ denotes the inverse of a square matrix. $A \succ 0$ and $A \succeq 0$ indicate that the symmetric matrix A (with $A^T = A$) is positive definite and positive semi-definite, respectively. Similarly, $A \prec 0$ and $A \preceq 0$ denote that the symmetric A is negative definite and negative semi-definite. Throughout, λ is reserved for denoting an eigenvalue, except in Chapter 5 and Chapter 6 where it represents a regularization parameter.

Given a vector x of size n, $\|x\|_2 := \sqrt{x^Tx}$ denotes the Euclidean vector norm, and $\|x\|_W = \sqrt{x^TWx}$ denotes the weighted vector norm under the matrix $W \in \mathbb{R}^{n\times n}$. The notation $\langle A, B\rangle := \mathrm{Tr}(A^TB)$ represents the Frobenius inner product between matrices $A, B \in \mathbb{R}^{n\times m}$, where the symbol Tr denotes the trace operator on a square matrix. $D(x) \in \mathbb{R}^{n\times n}$ represents a diagonal matrix constructed from the vector $x \in \mathbb{R}^n$, where the elements of x are placed on the diagonal. The symbol ./ denotes element-wise division between vectors of the same dimension. ∇_x and ∇_X denote gradient operators with respect to vector x and matrix X, respectively. If there's no ambiguity, the subscript notation of the gradient operators is dropped.

CHAPTER 2

Preliminaries

2.1. Rayleigh Quotient Optimization (RQopt)

Many scientific problems involve optimization of the ratio of two quadratic forms. In mechanics, for example, the ratio between potential energy x^TAx and kinetic energy x^TBx is often maximized. Consequently, a key function in these type of problems is the Rayleigh quotient:

$$q(x) := \frac{x^TAx}{x^TBx} \tag{2.1}$$

defined for all $x \neq 0$ where $A, B \in \mathbb{R}^{n\times n}$ are symmetric matrices with B positive definite ($B \succ 0$).

2.1.1. Rayleigh Quotient and Eigenvalues. There exists a fundamental relation between the Rayleigh quotient $q(x)$ (2.1) and the eigenvalues of the following linear eigenvalue problem

$$Ax = \lambda Bx, \tag{2.2}$$

where the matrices A and B correspond to matrices in (2.1). Namely, the eigenvalues of (2.2) are characterized as the minmax/maxmin optimizations of the Rayleigh quotient $q(x)$ (2.1). This result is known as the Courant-Fischer minimax theorem [**84**, Theorem 10.2.1].

Theorem 2.1.1 (Courant-Fischer minimax theorem). *Let $\mathcal{W}$ denote a subspace of $\mathbb{R}^n$. If $A, B \in \mathbb{R}^{n\times n}$ are symmetric matrices with B positive definite such that the eigenvalues of the matrix pair (A, B) are ordered as $\lambda_1(A,B) \geqslant \lambda_2(A,B) \geqslant \cdots \geqslant \lambda_n(A,B)$, then*

$$\lambda_i(A,B) = \min_{dim(\mathcal{W})=n-i+1} \max_{\substack{w\in\mathcal{W}\\ w^TBw=1}} w^TAw \tag{2.3}$$

and

$$\lambda_i(A,B) = \max_{dim(\mathcal{W})=i} \min_{\substack{w\in\mathcal{W}\\ w^TBw=1}} w^TAw \tag{2.4}$$

for $i = 1, 2, \ldots, n$.

The generalization of the Courant-Fischer minimiax theorem to the sum of eigenvalues is the result presented in [**43**, Theorem 1], known commonly as Ky Fan's maximum principle or Ky Fan trace theorem.

THEOREM 2.1.2 (Ky Fan trace theorem). *If* $A, B \in \mathbb{R}^{n\times n}$ *are symmetric matrices with* B *positive definite such that the eigenvalues of the matrix pair* (A, B) *are ordered as* $\lambda_1(A, B) \geqslant \lambda_2(A, B) \geqslant \cdots \geqslant \lambda_n(A, B)$*, then*

$$\sum_{i=1}^{p} \lambda_i(A, B) = \max_{X^TBX=I_p} \operatorname{Tr}(X^TAX) \tag{2.5}$$

and

$$\sum_{i=d-p+1}^{d} \lambda_i(A, B) = \min_{X^TBX=I_p} \operatorname{Tr}(X^TAX) \tag{2.6}$$

for $p = 1, 2, \ldots, n$. X *is a solution to the maximum problem in* (2.5) *if and only if* X *is an orthonormal eigenbasis corresponding to the* p *largest eigenvalues of* (A, B)*. Similarly,* X *is a solution to the minimum problem in* (2.6) *if and only if* X *is an orthonormal eigenbasis corresponding to the* p *smallest eigenvalues of* (A, B).

2.1.2. RQopt and Linear Eigenvalue Problem. We consider the problem of maximizing the Rayleigh quotient, which we refer to as RQopt:

$$\max_{x\neq 0} \left\{ q(x) := \frac{x^TAx}{x^TBx} \right\}. \tag{RQopt}$$

(RQopt) arises in various applications, such as the study of vibrations analysis, buckling, elasticity, etc. In machine learning, it arises in problems such as generalized eigenvalue classifiers [**76**], common spatial pattern [**17**], and Fisher's linear discriminant analysis [**48**].

As a direct consequence of Theorem 2.1.1 (the Courant-Fischer minimax theorem), we have the following result between (RQopt) and the linear eigenvalue problem (2.2).

THEOREM 2.1.3. x *is a global maximizer of* (RQopt) *if and only if* x *is an eigenvector of* (2.2) *with respect to the largest eigenvalue.*

2.1.3. RQopt Algorithms. To solve (RQopt), we can compute the eigenvector of (2.2) corresponding to the largest eigenvalue.

There are several strategies and algorithms available for computing the largest eigenpair of (2.2). One approach involves transforming (2.2) into a standard linear eigenvalue problem by applying Cholesky decomposition, where B is decomposed as LL^T. After this transformation, classical algorithms such as the Power method, Shift-and-invert method, Lanczos method, and Jacobi-Davidson method can be applied to the matrix $L^{-1}AL^{-T}$ [**8**,**49**,**84**]. Additionally, other algorithms, including the implicitly restarted Arnoldi algorithm [**100**], the Krylov-Schur algorithm [**102**], and LOBPCG [**64**], can be employed to compute the largest eigenpair of (2.2).

2.1.4. Nonlinear Generalization of RQopt. The nonlinear Rayleigh quotient optimization (NRQopt) is a nonlinear generalization of (RQopt) in which the matrices depend nonlinearly on the vector. It is defined as:

$$\text{(NRQopt)} \qquad \max_{x\neq 0}\left\{q(x) := \frac{x^TA(x)x}{x^TB(x)x}\right\},$$

where, similar to (RQopt), $A(x), B(x) \in \mathbb{R}^{n\times n}$ are symmetric matrices with $B(x)$ positive definite.

(NRQopt) can arise from the robust Rayleigh quotient optimization, where the data matrices of the of the Rayleigh quotient are influenced by uncertainties [**11**]. Some of the applications for the robust Rayleigh quotient optimization include the robust Fisher LDA [**61**], robust generalized eigenvalue classifier [**119**], and robust common spatial pattern [**59**]. In Chapter 3 and Chapter 4, we discuss (NRQopt) in the context of the robust common spatial pattern, a signal processing technique in brain-computer interface system. The robust common spatial pattern improves the robustness of the standard common spatial pattern by taking into account the nonstationarity of the data. In this context, (NRQopt) is formulated as an eigenvector-dependent nonlinear eigenvalue problem (NEPv). Subsequently, we employ the self-consistent field (SCF) iteration to solve the resulting NEPv.

2.2. Trace Ratio Optimization (TRopt)

2.2.1. Trace Ratio. Many dimensionality reduction and feature extraction problems in pattern recognition, computer vision, and machine learning concern the trace ratio optimization

(TRopt). In TRopt, the goal is to maximize the trace ratio $q(X)$, subject to the orthonormality constraint:

$$\text{(TRopt)} \qquad \max_{X^TX=I_p} \left\{ q(X) := \frac{\operatorname{Tr}(X^TAX)}{\operatorname{Tr}(X^TBX)} \right\}.$$

(TRopt) is defined for symmetric matrices $A, B \in \mathbb{R}^{n\times n}$, where B is positive definite. [1] Here, p is typically much smaller than n.

(TRopt) predominantly arises in problems such as Fisher's linear discriminant analysis (LDA) [**48**], Marginal Fisher analysis (MFA) [**120**], and Local discriminant embedding [**30**], where the goal is to maximize one quantity while minimizing another. In LDA, for example, the objective is to find an orthonormal projection X that maximizes the dispersion between classes while minimizing the dispersion within classes. This objective leads to (TRopt), where A corresponds to the between-class scatter matrix S_b, and B corresponds to the within-class scatter matrix S_w.

2.2.1.1. *TRopt is not RTopt.* While sometimes overlooked, it is crucial to differentiate between (TRopt) and the ratio trace optimization (RTopt), defined as:

$$\text{(RTopt)} \qquad \max_{X\in\mathbb{R}^{n\times p}} \operatorname{Tr}\left((X^TBX)^{-1}X^TAX\right).$$

The solution of (RTopt) is an eigenbasis of (2.2) corresponding to the p largest eigenvalues [**48**]. On the other hand, the solution of (TRopt) corresponds to an eigenbasis of an eigenvector-dependent nonlinear eigenvalue problem (NEPv).

The use of the surrogate model (RTopt) often leads to a suboptimal solution of (TRopt). Moreover, in dimensionality reduction and feature extraction problems, an additional challenge of discriminability arises, especially when posterior classification or clustering is important. The discriminant capability of (RTopt) is generally inferior to that of (TRopt), resulting in (RTopt) achieving inferior classification and clustering performance compared to (TRopt) [**51**, **79**, **115**].

2.2.2. Relation between TRopt and NEPv. The following lemma states that a local maximizer of (TRopt) is an orthonormal eigenbasis of a NEPv corresponding to some p eigenvalues.

[1] While we defined (TRopt) for a positive definite B, it is possible to relax the positive definiteness of B to positive semi-definiteness. As long as the rank of B is at least $n-p+1$, it can be positive semi-definite to guarantee that $\operatorname{Tr}(X^TBX) > 0$ [**79**].

Lemma 2.2.1 ([**124**, **126**]). *If X is a local maximizer of* (TRopt)*, then it is an orthonormal eigenbasis of the following NEPv:*

$$H(X)X = XM_X, \tag{2.7}$$

where

$$H(X) := A - q(X)B \quad \textit{with} \quad q(X) = \frac{\mathrm{Tr}(X^T AX)}{\mathrm{Tr}(X^T BX)}, \tag{2.8}$$

and $M_X := X^T H(X)X$ is an eigenblock. Moreover, the sum of the eigenvalues of $H(X)$ corresponding to the orthonormal eigenbasis X is zero.

Proof. The Lagrangian function of (TRopt) is defined as

$$L(X, \Gamma) = \frac{Tr(X^T AX)}{Tr(X^T BX)} - Tr(\Gamma(X^T X - I_p)), \tag{2.9}$$

where $\Gamma \in \mathbb{R}^{p\times p}$ is symmetric and contains the Lagrange multipliers. By the result $\nabla\mathrm{Tr}(X^T AX) = 2AX$ for symmetric A (Lemma A.0.1(c)), the gradient of $L(X, \Gamma)$ with respect to X yields

$$\begin{aligned}\nabla_X L(X, \Gamma) &= \frac{(2AX)(Tr(X^T BX)) - (Tr(X^T AX))(2BX)}{(Tr(X^T BX))^2} - 2X\Gamma \\ &= \frac{2}{Tr(X^T BX)} H(X)X - 2X\Gamma. \end{aligned} \tag{2.10}$$

The first-order necessary conditions [**82**, Theorem 12.1, p.321] of (TRopt),

$$\nabla_X L(X, \Lambda) = 0 \quad \text{and} \quad X^T X = I_p,$$

imply

$$\Gamma = \frac{1}{\mathrm{Tr}(X^T BX)} X^T H(X)X. \tag{2.11}$$

Substituting Γ (2.11) into (2.10) and setting it to 0 we obtain

$$H(X)X = X(X^T H(X)X).$$

Moreover, the sum of the eigenvalues of $H(X)$ corresponding to the orthonormal eigenbasis X is

$$\mathrm{Tr}(M_X) = \mathrm{Tr}(X^T H(X) X) = \mathrm{Tr}(X^T A X) - q(X)\mathrm{Tr}(X^T B X) = 0.$$

□

A local maximizer X of (TRopt) is an orthonormal eigenbasis of $H(X)$ such that the corresponding p eigenvalues sum to zero. However, the *eigenvalue ordering* is unknown, meaning that we do not know which of the eigenvalues of $H(X)$ these p eigenvalues correspond to.

Fortunately, when X is a global maximizer, the eigenvalue ordering is known.

Theorem 2.2.1 ([**124, 126**]). *X is a global maximizer of* (TRopt) *if and only if X is an orthonormal eigenbasis of the NEPv* (2.7) *corresponding to the p largest eigenvalues of $H(X)$.*

Proof. Let X be a global maximizer of (TRopt). By Lemma 2.2.1, X is an orthonormal eigenbasis of the NEPv (2.7). Suppose, on the contrary, that the eigenvalues corresponding to X are not the p largest eigenvalues of $H(X)$. Instead, let V be an orthonormal eigenbasis of $H(X)$ corresponding to the p largest eigenvalues. Then, according to Theorem 2.1.2 (the Ky Fan trace theorem), V is a solution to

$$\max_{Y^T Y = I_p} \mathrm{Tr}(Y^T H(X) Y)$$

while X is not. Thus, we have

$$\mathrm{Tr}(V^T H(X) V) > \mathrm{Tr}(X^T H(X) X) = 0.$$

The inequality indicates that $q(V) > q(X)$, which is a contradiction since X is a global optimizer of (TRopt). Therefore, X must be an orthonormal eigenbasis of $H(X)$ corresponding to the p largest eigenvalues.

Conversely, suppose that X is an orthonormal eigenbasis of $H(X)$ corresponding to the p largest eigenvalues. Then, for any orthonormal V, we have

$$0 = \mathrm{Tr}(X^T H(X) X) \geqslant \mathrm{Tr}(V^T H(X) V) = \mathrm{Tr}(V^T A V) - q(X)\mathrm{Tr}(V^T B V) = \mathrm{Tr}(V^T B V)(q(V) - q(X)).$$

Since $\mathrm{Tr}(V^TBV) > 0$ due to the positive definiteness of B, this implies $q(X) \geqslant q(V)$, indicating that X is a global maximizer of (TRopt). $\square$

2.2.3. TRopt Algorithms. A widely used approach for solving (TRopt) is through the application of parametrization techniques. Research efforts [**51**, **55**, **79**, **80**, **115**, **124**, **128**] in this approach utilize the fact that the maximal value of (TRopt) corresponds to the *unique root* of the parametric function $f(q)$, defined as

$$f(q) := \max_{X^TX=I_p} \mathrm{Tr}(X^T(A - qB)X), \qquad q \in \mathbb{R}. \tag{2.12}$$

Notably, an efficient iterative method proposed in [**115**] computes the root of $f(q)$ by solving the nonlinear parametrization programming (2.12) at each iteration. This iterative method can be interpreted as the Newton–Raphson method for solving $f(q) = 0$ [**55**, **79**], as well as the self-consistent field (SCF) iteration for solving the NEPv (2.7) [**25**, **125**].

Interestingly enough, applying the Dinkelbach's algorithm [**37**], designed for nonlinear fractional programming, to (TRopt) also yields an equivalence with this iterative method. This connection has not been previously explored in the literature. In light of this, we provide an overview of the Dinkelbach's algorithm for solving (TRopt). Afterwards, we discuss the Newton–Raphson method and the SCF iteration, and establish the equivalence of the Dinkelbach's algorithm to these two methods.

2.2.3.1. *Dinkelbach's Algorithm.* We provide a summary of the Dinkelbach's algorithm in Appendix B for a general nonlinear fractional programming.

Note that $\mathrm{Tr}(X^TAX)$ and $\mathrm{Tr}(X^TBX)$ are continuous and real-valued functions defined over $\{X \in \mathbb{R}^{n\times p} : X^TX = I_p\}$, a compact and connected subset. In addition, we have $\mathrm{Tr}(X^TBX) > 0$. Therefore, according to Lemma B.1.1:

- The parametric function $f(q)$ (2.12) is a continuous and strictly monotonic decreasing function.
- $f(q)$ has a unique root.

Furthermore, as stated in Theorem B.1.1:

- q_* is the maximal value of (TRopt) if and only if q_* is the root of $f(q)$.

- X_* is a global maximizer of (TRopt) if and only if X_* is a global maximizer of the parametrization programming (2.12) at $q = q_*$.

In other words, we have

$$q_* = \frac{\mathrm{Tr}(X_*^T A X_*)}{\mathrm{Tr}(X_*^T B X_*)} = \max_{X^T X = I_p} \frac{\mathrm{Tr}(X^T A X)}{\mathrm{Tr}(X^T B X)}$$

and

$$0 = f(q_*) = \mathrm{Tr}(X_*^T (A - q_* B) X_*) = \max_{X^T X = I_p} \mathrm{Tr}(X^T (A - q_* B) X).$$

At each iteration, the Dinkelbach's algorithm computes the following steps:

(1) $q_k = \frac{\mathrm{Tr}(X_k^T A X_k)}{\mathrm{Tr}(X_k^T B X_k)}$.
(2) $X_{k+1} = \mathrm{argmax}_{X^T X = I_p} \{\mathrm{Tr}(X^T (A - q_k B) X)\}$.

According to Theorem 2.1.2 (the Ky Fan trace theorem), X_{k+1} in the second step corresponds to the orthonormal eigenbasis of the matrix $A - q_k B$ corresponding to the p largest eigenvalues.

The convergence proof of Dinkelbach's algorithm in Appendix B.2.1 guarantees that $\{q_k\}$ is a strictly monotonic increasing sequence converging to the root q_* of $f(q)$, or equivalently, the maximal value of (TRopt).

2.2.3.2. *Newton-Raphson Method.* Applied to computing the root of the parametric function $f(q)$ (2.12), the Newton-Raphson method makes an update according to

$$q_{k+1} = q_k - \frac{f(q_k)}{f'(q_k)}, \tag{2.13}$$

where f' denotes the derivative of f. In the following, we derive $f'(q_k)$.

To each $q \in \mathbb{R}$, we can express the following eigenvalue problem

$$(A - qB)X(q) = X(q)D(q),$$

where $X(q)$ is an orthonormal eigenbasis of $A - qB$ corresponding to the p largest eigenvalues, and $D(q)$ is a diagonal matrix containing these eigenvalues. Thus, we can denote $f(q)$ as

$$f(q) = \mathrm{Tr}(X(q)^T (A - qB) X(q)). \tag{2.14}$$

Additionally, since the matrix $A - qB$ is symmetric and differentiable with respect to $q \in \mathbb{R}$, the orthonormal eigenbasis $X(q)$ is differentiable with respect to $q \in \mathbb{R}$ as well [**58**, Chapter II, section 6]. We outline the steps to compute the derivative of (2.14).

First, we have

$$\begin{aligned}
\frac{d}{dq}(X(q)^T(A - qB)X(q)) &= \frac{d}{dq}(X(q)^T AX(q)) - \frac{d}{dq}(qX(q)^T BX(q)) \\
&= \frac{dX(q)}{dq}^T AX(q) + X(q)^T A\frac{dX(q)}{dq} \\
&\quad - X(q)^T BX(q) - q\bigg(\frac{dX(q)}{dq}^T BX(q) + X(q)^T B\frac{dX(q)}{dq}\bigg) \\
&= \frac{dX(q)}{dq}^T (A - qB)X(q) + X(q)^T(A - qB)\frac{dX(q)}{dq} - X(q)^T BX(q) \\
&= \frac{dX(q)}{dq}^T X(q)D(q) + D(q)X(q)^T \frac{dX(q)}{dq} - X(q)^T BX(q). \qquad (2.15)
\end{aligned}$$

Furthermore, the orthonormality of $X(q)$ implies

$$0 = \frac{d(X(q)^T X(q))}{dq} = \frac{dX(q)}{dq}^T X(q) + X(q)^T \frac{dX(q)}{dq},$$

which indicates that $\frac{dX(q)}{dq}^T X(q)$ has zero diagonals. Consequently, by taking the trace of (2.15), we obtain

$$\begin{aligned}
\frac{df(q)}{dq} &= 2\mathrm{Tr}\bigg(\frac{dX(q)}{dq}^T X(q)D(q)\bigg) - \mathrm{Tr}(X(q)^T BX(q)) \\
&= -\mathrm{Tr}(X(q)^T BX(q)).
\end{aligned}$$

Therefore, the Newton-Raphson method (2.13) is:

$$\begin{aligned}
q_{k+1} &= q_k - \frac{\mathrm{Tr}(X(q_k)^T(A - q_k B)X(q_k))}{-\mathrm{Tr}(X(q_k)^T BX(q_k))} \\
&= \frac{\mathrm{Tr}(X(q_k)^T AX(q_k))}{\mathrm{Tr}(X(q_k)^T BX(q_k))}.
\end{aligned}$$

In summary, at each iteration, the Newton-Raphson method computes the following steps:

(1) Compute $X(q_k)$ as the orthonormal eigenbasis of $A - q_k B$ corresponding to the p largest eigenvalues.

(2) $q_{k+1} = \frac{\mathrm{Tr}(X(q_k)^T AX(q_k))}{\mathrm{Tr}(X(q_k)^T BX(q_k))}$.

2.2.3.3. *SCF Iteration.* In Theorem 2.2.1, we established that a global maximizer of (TRopt) corresponds to an orthonormal eigenbasis of the NEPv (2.7) associated with the p largest eigenvalues. A commonly employed approach for solving a NEPv problem is the self-consistent field (SCF) iteration.

Applied to solving the NEPv (2.7), the SCF iteration can be summarized as follows:

$$X_{k+1} \longleftarrow \text{ eigenvectors of the } p \text{ largest eigenvalues of } A - q(X_k)B. \tag{2.16}$$

2.2.3.4. *Equivalence of Algorithms.* As evident from the discussion, when applied to solving (TRopt), the Dinkelbach's algorithm shares the same updating scheme as the Newton-Raphson method (2.13) and the SCF iteration (2.16) despite not relying on differentiability or addressing eigenvector-dependent nonlinear eigenvalue problems.

We provide the summary of the algorithm in Algorithm 1.

Algorithm 1 (TRopt) Solver

Input: Initial $X_0 \in S$, tolerance *tol*.

Output: A global maximizer X_* of (TRopt).

```
1: for $k = 0, 1, \ldots$ do
2:    Set $q_k = \frac{\mathrm{Tr}(X_k^T AX_k)}{\mathrm{Tr}(X_k^T BX_k)}$.
3:    Compute an orthonormal eigenpair $(X_{k+1}, \Lambda_{k+1})$ of $A - q_k B$ corresponding to the $p$ largest
      eigenvalues.
4:    if $\mathrm{Tr}(\Lambda_{k+1}) \leqslant \mathrm{tol}$ then
5:       return $X_* = X_{k+1}$.
6:    end if
7: end for
```

The stopping criteria in line 4 of Algorithm 1 can be interpreted in two ways: either as checking whether the root of the parametric function $f(q)$ (2.12) has been found or as verifying that the sum of the p largest eigenvalues of the NEPv (2.7) is zero.

Algorithm 1 is globally convergent to a global maximizer of (TRopt) [**37**, **79**, **115**, **124**] and has a local quadratic convergence [**25**, **124**].

2.2.4. Nonlinear Trace Ratio Optimization (NTRopt). Similar to the generalization of (RQopt) to (NRQopt), the nonlinear trace ratio optimization (NTRopt) is a nonlinear generalization of (TRopt). It is defined as the maximization of the nonlinear trace ratio $q(X)$, subject to the orthonormality constraint:

$$\text{(NTRopt)} \qquad \max_{X^TX=I_p} \left\{ q(X) := \frac{\operatorname{Tr}(X^TA(X)X)}{\operatorname{Tr}(X^TB(X)X)} \right\},$$

where $A(X), B(X) \in \mathbb{R}^{n\times n}$ are symmetric with $B(X)$ positive definite. As in (TRopt), p is typically much smaller than n.

One instance where (NTRopt) can arise is in the effort to enhance the robustness of the data matrices of (TRopt) which are subject to uncertainties. Similar to how (NRQopt) is derived from (RQopt), (NTRopt) can also be derived from the robust trace ratio optimization using the same methodology.

However, more generally, (NTRopt) can be encountered in a bi-level optimization of the trace ratio involving inner and outer optimizations. In Chapter 5 and Chapter 6, we discuss dimensionality reduction application called Wasserstein discriminant analysis (WDA), which follows the structure of bi-level optimization and is formulated as (NTRopt). WDA can account for both global and local interconnections between data classes by using the underlying principles of optimal transport. We introduce an eigenvector algorithm that solves WDA by leveraging the bi-level structure within the framework of eigenvector-dependent nonlinear eigenvalue problems.

CHAPTER 3

Robust Common Spatial Pattern Analysis: NRQopt and NEPv

3.1. Introduction

3.1.1. BCI, EEG, and ERD. Brain-computer interface (BCI) is a system that translates the participating subject's intent into control actions, such as control of computer applications or prosthetic devices [**41**, **69**, **118**]. BCI studies have utilized a range of approaches for capturing the brain signals pertaining to the subject's intent, from invasive (such as brain implants) and partially invasive (inside the skull but outside the brain) to non-invasive (such as head-worn electrodes). Among non-invasive approaches, electroencephalogram (EEG) based BCIs are the most widely studied due to their ease of setup and non-invasive nature.

In EEG based BCIs, the brain signals are often captured through the electrodes (small metal discs) attached to the subject's scalp while they perform motor imagery tasks. An EEG-BCI session typically involves two phases: the calibration phase and the feedback phase [**17**]. During the calibration phase, the preprocessed and spatially filtered EEG signals are used to train a classifier. In the feedback phase, the trained classifier interprets the subject's EEG signals and translates them into control actions, such as cursor control or button selection, often occurring in real-time.

The translation of the subject's motor imagery into commands is based on the phenomenon known as the event-related desynchronization (ERD) [**17**, **88**]. Processing of motor imagery commands causes an ERD, a decrease in the rhythmic activity over the sensorimotor cortex. Specific locations in the sensorimotor cortex are related to corresponding parts of the body. For example, the left and right hands correspond to the right and left motor cortex, respectively. Consequently, the cortical area where an ERD occurs indicates which motor task is taking place.

3.1.2. CSP. Spatial filtering is a crucial step to uncover the discriminatory modulation of an ERD [**17**]. One popular spatial filtering technique in EEG-based BCIs is common spatial pattern

(CSP) analysis [**17**, **65**, **75**, **89**]. The CSP is a data-driven approach that computes subject-specific spatial filters to increase the signal-to-noise ratio in EEG signals and uncover the discrimination between two motor imagery conditions [**95**]. Mathematically, the principal spatial filters that optimally discriminate the motor imagery conditions can be expressed as solutions to Rayleigh quotient optimizations.

3.1.3. Robust CSP and Existing Work. However, the CSP is often subject to poor performance due to the nonstationary nature of EEG signals [**17**, **95**]. The nonstationarity of a signal is influenced by various factors such as artifacts resulting from non-task-related muscle movements and eye movements, as well as between-session errors such as small differences in electrode positions and a gradual increase in the subject's tiredness. Consequently, some of the EEG signals can be noisy and act as outliers. This leads to poor average covariance matrices that do not accurately capture the underlying variances of the signals. As a result, the CSP, which computes its principal spatial filters using the average covariance matrices in the Rayleigh quotient, can be highly sensitive to noise and prone to overfitting [**89**, **123**].

There are many variants of the CSP that seek to improve its robustness. One approach involves regularizing the CSP to ensure its spatial filters remain invariant to nonstationarities in the signals [**14**, **95**], or by first removing the nonstationary contributions before applying the CSP [**94**]. Another approach includes the use of divergence functions from information geometry, such as the Kullback–Leibler divergence [**6**] and the beta divergence [**93**], to mitigate the influence of outliers. Reformulations of the CSP based on norms other than the L2-norm also seek to suppress the effect of outliers. These norms include the L1-norm [**71**, **114**] and the L21-norm [**50**]. Another approach aims to enhance the robustness of the covariance matrices. This includes the dynamic time warping approach [**7**], the application of the minimum covariance determinant estimator [**123**], and the minmax CSP [**59**, **60**].

3.2. Common Spatial Pattern Analysis

3.2.1. Data Preprocessing and Data Covariance Matrices. An EEG-BCI session involves collecting trials, which are time series of EEG signals recorded from electrode channels placed on the subject's scalp during motor imagery tasks. Data preprocessing steps of the trials include channel selection, bandpass filtering, and time interval selection. Channel selection involves choosing the electrode channels around the cortical area corresponding to the motor imagery actions to increase the signal-to-noise ratio. Bandpass filtering is used to filter out low and high frequencies that may originate from non-EEG related artifacts. Time interval selection is used to choose the time interval in which the motor imagery takes place. By applying these preprocessing steps, the quality of the EEG signals is improved, leading to better performance of CSP [**17**,**40**,**78**].

We use $Y_c^{(i)}$ to denote the preprocessed i^{th} trial for the motor imagery condition c, where $c \in \{-,+\}$ represents the condition of interest, such as left hand and right hand motor imagery. Each trial $Y_c^{(i)}$ is a $n \times t$ matrix, where a row corresponds to an EEG signal sampled at an electrode channel with t many sampled time points. We assume that the trials are scaled and centered, i.e.,

$$Y_c^{(i)} \longleftarrow \frac{1}{\sqrt{t-1}} Y_c^{(i)} (I_t - \frac{1}{t} 1_t 1_t^T).$$

The data covariance matrix for the trial $Y_c^{(i)}$ is defined as

$$\Sigma_c^{(i)} = Y_c^{(i)} Y_c^{(i)T} \in \mathbb{R}^{n \times n}. \tag{3.1}$$

Denoting N_c as the number of trials, the average of the data covariance matrices is given by

$$\overline{\Sigma}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \Sigma_c^{(i)} \in \mathbb{R}^{n \times n}. \tag{3.2}$$

Assumption 3.2.1. *We assume that the average covariance matrix* $\overline{\Sigma}_c$ (3.2) *is positive definite, i.e.,*

$$\overline{\Sigma}_c \succ 0. \tag{3.3}$$

Assumption 3.2.1 is reasonable given that $\overline{\Sigma}_c$ is the average of the data covariance matrices $\Sigma_c^{(i)}$ which are positive semi-definite.

3.2.2. Spatial Filters and Linear Eigenvalue Problems.

3.2.2. Spatial Filters and Linear Eigenvalue Problems. The principle of CSP is to find spatial filters such that the variance of the spatially filtered signals under one condition is minimized, while that of the other condition is maximized. Let the columns of $X \in \mathbb{R}^{n \times n}$ denote the n spatial filters. X is defined such that the following couple of properties hold [**17**, **65**]:

(i) The average covariance matrices $\overline{\Sigma}_-$ and $\overline{\Sigma}_+$ are simultaneously diagonalized such that

$$\begin{aligned} X^T\overline{\Sigma}_-X &= \Lambda_-, \\ X^T\overline{\Sigma}_+X &= \Lambda_+, \end{aligned} \tag{3.4}$$

where Λ_- and Λ_+ are diagonal matrices.

(ii) The spatial filters X are scaled such that

$$X^T(\overline{\Sigma}_- + \overline{\Sigma}_+)X = \Lambda_- + \Lambda_+ = I_n. \tag{3.5}$$

Let $Z_c^{(i)}$ denote the spatially filtered trial of $Y_c^{(i)}$, i.e.,

$$Z_c^{(i)} := X^T Y_c^{(i)}. \tag{3.6}$$

Then,

(i) By the property (3.4), the average covariance matrix of $Z_c^{(i)}$ is diagonalized:

$$\frac{1}{N_c}\sum_{i=1}^{N_c} Z_c^{(i)} {Z_c^{(i)}}^T = \frac{1}{N_c}\sum_{i=1}^{N_c} X^T Y_c^{(i)} {Y_c^{(i)}}^T X = X^T\Big(\frac{1}{N_c}\sum_{i=1}^{N_c}\Sigma_c^{(i)}\Big)X = X^T\overline{\Sigma}_c X = \Lambda_c. \tag{3.7}$$

This indicates that the spatially filtered signals between electrode channels are uncorrelated.

(ii) The diagonals of Λ_c (3.7) correspond to the average variances of the filtered signals in condition c. Since both $\overline{\Sigma}_-$ and $\overline{\Sigma}_+$ are positive definite by Assumption 3.2.1, the property (3.5) indicates that the variance of either condition must lie between 0 and 1, and that the sum of the variances of the two conditions must be 1. This indicates that a small variance in one condition implies a large variance in other condition, and vice versa.

The properties (3.4) and (3.5) together imply that the spatial filters X correspond to the eigenvectors of the eigenvalue problem

$$\begin{cases} \overline{\Sigma}_-X = (\overline{\Sigma}_- + \overline{\Sigma}_+)X\Lambda_- \\ X^T(\overline{\Sigma}_- + \overline{\Sigma}_+)X = I_n, \end{cases} \tag{3.8}$$

or, equivalently, the eigenvectors of the eigenvalue problem

$$\begin{cases} \overline{\Sigma}_{+}X = (\overline{\Sigma}_{-} + \overline{\Sigma}_{+})X\Lambda_{+} \\ X^{T}(\overline{\Sigma}_{-} + \overline{\Sigma}_{+})X = I_{n}. \end{cases} \tag{3.9}$$

3.2.3. Principal Spatial Filters and Rayleigh Quotient Optimizations. In practice, only a small subset of the spatial filters, rather than all n of them, are necessary for CSP [**17**, **59**]. In this paper, we consider the computations of the **principal spatial filters** x_{-} and x_{+}, where x_{-} minimizes the variance of condition '$-$' and x_{+} minimizes the variance of condition '$+$'.

By the equivalence of the spatial filters X to the eigenvectors of the eigenvalue problems (3.8) and (3.9), x_{-} is the eigenvector of of the matrix pair $(\overline{\Sigma}_{-}, \overline{\Sigma}_{-} + \overline{\Sigma}_{+})$ with respect to the smallest eigenvalue, and x_{+} is the eigenvector of of the matrix pair $(\overline{\Sigma}_{+}, \overline{\Sigma}_{-} + \overline{\Sigma}_{+})$ with respect to the smallest eigenvalue.

By Theorem 2.1.1 (the Courant-Fischer minimax theorem), x_{-} is the solution of the Rayleigh quotient optimization (RQopt)[1]

$$\min_{x\neq 0} \frac{x^{T}\overline{\Sigma}_{-}x}{x^{T}(\overline{\Sigma}_{-} + \overline{\Sigma}_{+})x} \tag{3.10}$$

and x_{+} is the solution of the Rayleigh quotient optimization

$$\min_{x\neq 0} \frac{x^{T}\overline{\Sigma}_{+}x}{x^{T}(\overline{\Sigma}_{-} + \overline{\Sigma}_{+})x}. \tag{3.11}$$

3.3. Minmax CSP

The minmax CSP proposed in [**59**, **60**] improves CSP's robustness to the nonstationarity and artifacts present in the EEG signals. Instead of being limited to fixed covariance matrices as in (3.10) and (3.11) for CSP, the minmax CSP considers the sets $\mathcal{S}_{c}$, for $c \in \{-, +\}$ containing candidate covariance matrices. The so-called robust principal spatial filters x_{-} and x_{+} are computed by

[1]Strictly speaking, the solution of the Rayleigh quotient optimization (3.10) may not satisfy the normalization constraint $x^{T}(\overline{\Sigma}_{-} + \overline{\Sigma}_{+})x = 1$. But, in this case, we can always normalize the obtained solution by applying the transformation $x \longleftarrow \frac{x}{\sqrt{x^{T}(\overline{\Sigma}_{-}+\overline{\Sigma}_{+})x}}$ to obtain x_{-}. Hence, we can regard the Rayleigh quotient optimization (3.10) as the optimization of interest for computing the principal spatial filter x_{-}.

solving the minmax optimizations[2]

$$\min_{x\neq 0} \max_{\substack{\Sigma_-\in\mathcal{S}_-\\ \Sigma_+\in\mathcal{S}_+}} \frac{x^T\Sigma_- x}{x^T(\Sigma_- + \Sigma_+)x} \tag{3.12}$$

and

$$\min_{x\neq 0} \max_{\substack{\Sigma_-\in\mathcal{S}_-\\ \Sigma_+\in\mathcal{S}_+}} \frac{x^T\Sigma_+ x}{x^T(\Sigma_- + \Sigma_+)x}, \tag{3.13}$$

respectively. The sets $\mathcal{S}_c$ are called the **tolerance sets** and they are constructed to reflect the regions of variability of the covariance matrices. The inner maximization in (3.12) and (3.13) corresponds to seeking the worst-case Rayleigh quotient within all possible covariance matrices in the tolerance sets. Optimizing for the worst-case behavior is a popular paradigm in optimization to obtain solutions that are more robust to noise and overfitting [**13**].

3.3.1. Tolerance Sets of the Minmax CSP. We consider the data-driven approach to construct $\mathcal{S}_c$ as the set of covariance matrices described by the interpolation matrices derived from the data covariance matrices $\{\Sigma_c^{(i)}\}_{i=1}^{N_c}$ (3.1). The tolerance sets in this approach can effectively capture the variability present in the data covariance matrices and eliminate the need for additional assumptions or prior information [**59**, **60**].

Given the average covariance matrix $\overline{\Sigma}_c$ (3.2), the symmetric interpolation matrices $V_c^{(i)} \in \mathbb{R}^{n\times n}$ for $i = 1, \ldots, m$, and the weight matrix $W_c \in \mathbb{R}^{m\times m}$ with positive weights $w_c^{(i)}$,

$$W_c := \operatorname{diag}\big(w_c^{(1)}, w_c^{(2)}, \ldots, w_c^{(m)}\big), \tag{3.14}$$

the tolerance sets of the data-driven approach are constructed as

$$\mathcal{S}_c := \left\{ \Sigma_c(\alpha_c) = \overline{\Sigma}_c + \sum_{i=1}^{m} \alpha_c^{(i)} V_c^{(i)} \;\middle|\; \|\alpha_c\|_{W_c^{-1}} \leqslant \delta_c \right\}, \tag{3.15}$$

[2]We point out that in [**59**, **60**], the optimizations are stated as equivalent maxmin optimizations. For instance, $\max_{x\neq 0} \min_{\substack{\Sigma_-\in\mathcal{S}_-\\ \Sigma_+\in\mathcal{S}_+}} (x^T\Sigma_+ x)/(x^T(\Sigma_- + \Sigma_+)x)$ is equivalent to our minmax optimization (3.12) using $(x^T\Sigma_+ x)/(x^T(\Sigma_- + \Sigma_+)x) = 1 - (x^T\Sigma_- x)/(x^T(\Sigma_- + \Sigma_+)x)$.

where $\delta_c > 0$ denotes the tolerance set radius. $\|\alpha_c\|_{W_c^{-1}}$ is a weighted vector norm defined as[3]

$$\|\alpha_c\|_{W_c^{-1}} := \sqrt{\alpha_c^T W_c^{-1} \alpha_c}. \tag{3.16}$$

The symmetric interpolation matrices $\{V_c^{(i)}\}_{i=1}^m$ and the weight matrix W_c are parameters that are obtained from the data covariance matrices $\{\Sigma_c^{(i)}\}_{i=1}^{N_c}$ (3.1) to reflect their variability.

3.3.1.1. *Why $\Sigma_c(\alpha_c)$ of $\mathcal{S}_c$ (3.15) increases robustness.* The average covariance matrix $\overline{\Sigma}_c$ tends to overestimate the largest eigenvalue and underestimate the smallest eigenvalue [**68**]. For CSP, which relies on the average covariance matrices, this effect can significantly influence the variance of two conditions. On the other hand, the covariance matrices $\Sigma_c(\alpha_c)$ of the tolerance set $\mathcal{S}_c$ (3.15) reduce the relative impact of under- and over-estimations by the addition of the term $\Sigma_{i=1}^m \alpha_c^{(i)} V_c^{(i)}$ to $\overline{\Sigma}_c$. As a result, $\Sigma_c(\alpha_c)$ can robustly represent the variance of its corresponding condition. Another factor contributing to the robustness of $\Sigma_c(\alpha_c)$ is the adequate size of the tolerance set $\mathcal{S}_c$. If the radius δ_c of $\mathcal{S}_c$ is zero, then $\Sigma_c(\alpha_c)$ is equal to $\overline{\Sigma}_c$, the covariance matrix used in the standard CSP. If δ_c is set to be large, then one can capture the influence of the outliers. By selecting an appropriate δ_c, $\Sigma_c(\alpha_c)$ can capture the intrinsic variability of the data while ignoring outliers that may dominate the variability.

3.3.1.2. *Computing $V_c^{(i)}$ and weight matrix W_c.* One way to obtain the interpolation matrices $V_c^{(i)}$ and the weight matrix W_c is by performing the principal component analysis (PCA) on the data covariance matrices $\{\Sigma_c^{(i)}\}_{i=1}^{N_c}$ (3.1). The following steps outline this process:

(1) Vectorize each data covariance matrix $\Sigma_c^{(i)}$ by stacking its columns into a n^2-dimensional vector, i.e., obtain $\mathrm{vec}(\Sigma_c^{(i)}) \in \mathbb{R}^{n^2}$.
(2) Compute the covariance matrix $\Gamma_c \in \mathbb{R}^{n^2 \times n^2}$ of the vectorized covariance matrices $\{\mathrm{vec}(\Sigma_c^{(i)})\}_{i=1}^m$.
(3) Compute the m largest eigenvalues and corresponding eigenvectors (principal components) of Γ_c as $\{w_c^{(i)}\}_{i=1}^m$ and $\{\nu_c^{(i)}\}_{i=1}^m$, respectively.
(4) Transform the eigenvectors $\{\nu_c^{(i)}\}_{i=1}^m$ into $n \times n$ matrices, i.e., obtain $\mathrm{mat}(\nu_c^{(i)})$, then symmetrizing afterwards to obtain the interpolation matrices $\{V_c^{(i)}\}_{i=1}^m$.
(5) Form the weight matrix $W_c := \mathrm{diag}\big(w_c^{(1)}, w_c^{(2)}, \ldots, w_c^{(m)}\big)$.

[3]In [**59**,**60**], the norm (3.16) was referred to as the *PCA-based norm*, reflecting the derivation of the weights $\{w_c^{(i)}\}_{i=1}^m$ from the PCA applied to the data covariance matrices. However, we recognize the norm (3.16) as a weighted vector norm defined for general values of $w_c^{(i)}$.

The interpolation matrices $V_c^{(i)}$ and the weights $w_c^{(i)}$ obtained in this process correspond to the principal components and variances of the data covariance matrices $\{\Sigma_c^{(i)}\}_{i=1}^{N_c}$, respectively. Consequently, the tolerance set $\mathcal{S}_c$ (3.15) has a nice geometric interpretation as an ellipsoid of covariance matrices that is centered at the average covariance matrix $\overline{\Sigma}_c$ and spanned by the principal components $V_c^{(i)}$. Moreover, note that a larger variance $w_c^{(i)}$ allows a larger value of $\alpha_c^{(i)}$ according to the norm (3.16). This implies that more significant variations are permitted in the directions indicated by the leading principal components. Therefore, the tolerance set $\mathcal{S}_c$ obtained in this process effectively captures the variability of the data covariance matrices.

We point out that the covariance matrix Γ_c may only be positive semi-definite if the number of data covariance matrices is insufficient. In such cases, some of the eigenvalues $w_c^{(i)}$ may be zero, causing the norm definition (3.16) to be ill-defined. To address this issue, we make sure to choose m such that all $\{w_c^{(i)}\}_{i=1}^{m}$ are positive.

3.3.2. Minmax CSP and Nonlinear Rayleigh Quotient Optimization. We first note that a covariance matrix $\Sigma_c(\alpha_c)$ of the tolerance set $\mathcal{S}_c$ (3.15) can be characterized as a function on the subspace $\Omega_c \subseteq \mathbb{R}^m$:

$$\Omega_c := \left\{\alpha_c \in \mathbb{R}^m \;\middle|\; \|\alpha_c\|_{W_c^{-1}} \leqslant \delta_c\right\}. \tag{3.17}$$

Consequently, we can express the optimizations (3.12) and (3.13) of the minmax CSP as

$$\min_{x\neq 0} \max_{\substack{\alpha_- \in \Omega_- \\ \alpha_+ \in \Omega_+}} \frac{x^T\Sigma_-(\alpha_-)x}{x^T(\Sigma_-(\alpha_-)+\Sigma_+(\alpha_+))x} \tag{3.18}$$

and

$$\min_{x\neq 0} \max_{\substack{\alpha_- \in \Omega_- \\ \alpha_+ \in \Omega_+}} \frac{x^T\Sigma_+(\alpha_+)x}{x^T(\Sigma_-(\alpha_-)+\Sigma_+(\alpha_+))x}, \tag{3.19}$$

respectively.

Assumption 3.3.1. *We assume that each covariance matrix in the tolerance set* (3.15) *is positive definite, i.e.,*

$$\Sigma_c(\alpha_c) \succ 0 \quad \textit{for all } \alpha_c \in \Omega_c. \tag{3.20}$$

Assumption 3.3.1 holds in practice for a small tolerance set radius δ_c. If $\Sigma_c(\alpha_c)$ fails the positive definiteness property, we may set its negative eigenvalues to zero, add a small perturbation term ϵI, or perform the modified Cholesky algorithm (as described in [**31**]) to obtain a positive definite matrix.

In the following, we show that the minmax CSP (3.18) and (3.19) can be formulated as nonlinear Rayleigh quotient optimizations [**11**, **59**, **60**].

THEOREM 3.3.1. *Under Assumption 3.3.1,*

(i) The minmax CSP (3.18) *is equivalent to the following* ***OptNRQ***$_-$*:*

$$\min_{x\neq 0}\left\{q_-(x) := \frac{x^T\Sigma_-(x)x}{x^T(\Sigma_-(x)+\Sigma_+(x))x}\right\} \tag{3.21}$$

where, for $c \in \{+,-\}$,

$$\Sigma_c(x) = \overline{\Sigma}_c + \sum_{i=1}^{m}\alpha_c^{(i)}(x)V_c^{(i)} \succ 0. \tag{3.22}$$

$\alpha_c^{(i)}(x)$ *is a component of* $\alpha_c(x) : \mathbb{R}^n \to \mathbb{R}^m$*, a vector-valued function of* x*, defined as*

$$\alpha_c(x) = -\frac{c\delta_c}{\|v_c(x)\|_{W_c}}W_c v_c(x), \tag{3.23}$$

where W_c *is the weight matrix defined in* (3.14) *and* $v_c(x) : \mathbb{R}^n \to \mathbb{R}^m$ *is a vector-valued function defined as*

$$v_c(x) := \begin{bmatrix} x^T V_c^{(1)} x \\ x^T V_c^{(2)} x \\ \vdots \\ x^T V_c^{(m)} x \end{bmatrix}. \tag{3.24}$$

$\{V_c^{(i)}\}_{i=1}^m$ *are the interpolation matrices.*

(ii) The minmax CSP (3.19) *is equivalent to the following* ***OptNRQ***$_+$*:*

$$\min_{x\neq 0}\left\{q_+(x) := \frac{x^T\Sigma_+(x)x}{x^T(\Sigma_-(x)+\Sigma_+(x))x}\right\} \tag{3.25}$$

where, for $c \in \{+, -\}$,

$$\Sigma_c(x) = \overline{\Sigma}_c + \sum_{i=1}^{m} \alpha_c^{(i)}(x) V_c^{(i)} \succ 0. \tag{3.26}$$

$\alpha_c^{(i)}(x)$ is a component of $\alpha_c(x) : \mathbb{R}^n \to \mathbb{R}^m$, a vector-valued function of x, defined as

$$\alpha_c(x) = \frac{c\delta_c}{\|v_c(x)\|_{W_c}} W_c v_c(x), \tag{3.27}$$

where W_c is the weight matrix defined in (3.14) and $v_c(x)$ is a vector defined in (3.24). $\{V_c^{(i)}\}_{i=1}^m$ are the interpolation matrices.

Proof. We show the proof for result (i). Result (ii) follows the proof of result (i) by swapping the roles of '$-$' and '$+$'.

By Assumption 3.3.1, it follows that

$$\max_{\substack{\alpha_- \in \Omega_- \\ \alpha_+ \in \Omega_+}} \frac{x^T \Sigma_-(\alpha_-) x}{x^T(\Sigma_-(\alpha_-) + \Sigma_+(\alpha_+))x} = \frac{\max_{\alpha_- \in \Omega_-} x^T \Sigma_-(\alpha_-) x}{\max_{\alpha_- \in \Omega_-} x^T \Sigma_-(\alpha_-) x + \min_{\alpha_+ \in \Omega_+} x^T \Sigma_+(\alpha_+) x}. \tag{3.28}$$

For both $c \in \{-, +\}$, we have

$$x^T \Sigma_c(\alpha_c) x = x^T \overline{\Sigma}_c x + \sum_{i=1}^{m} \alpha_c^{(i)} (x^T V_c^{(i)} x) = x^T \overline{\Sigma}_c x + \alpha_c^T v_c(x)$$

with $v_c(x)$ as defined in (3.24). Thus, the solutions to the two optimizations in (3.28) are found by optimizing

$$\operatorname*{argmax}_{\alpha_- \in \Omega_-} \alpha_-^T v_-(x) \qquad \text{and} \qquad \operatorname*{argmin}_{\alpha_+ \in \Omega_+} \alpha_+^T v_+(x). \tag{3.29}$$

We have

$$|\alpha_c^T v_c(x)| = |\alpha_c^T W_c^{-1/2} W_c^{1/2} v_c(x)| = |(W_c^{-1/2}\alpha_c)^T (W_c^{1/2} v_c(x))|$$

$$\leqslant \|W_c^{-1/2}\alpha_c\|_2 \cdot \|W_c^{1/2} v_c(x)\|_2 = \sqrt{\alpha_c^T W_c^{-1} \alpha_c} \cdot \sqrt{v_c(x)^T W_c v_c(x)} \tag{3.30}$$

$$\leqslant \delta_c \cdot \|v_c(x)\|_{W_c} \tag{3.31}$$

where the inequality (3.30) follows from the Cauchy-Schwarz inequality and the inequality (3.31) follows from the constraint $\|\alpha_c\|_{W_c^{-1}} \leqslant \delta_c$ of the subspace Ω_c (3.17).

With the choice of $\alpha_c(x) = \pm \frac{c\delta_c}{\|v_c(x)\|_{W_c}} W_c v_c(x)$ in Ω_c, we have $|\alpha_c^T v_c(x)| = \delta_c \cdot \|v_c(x)\|_{W_c}$. Therefore, the solutions to the two optimizations in (3.29) correspond to the $\alpha_c(x)$ as defined in (3.23). $\square$

In the rest of this chapter, we focus our discussion and analysis on OptNRQ$_-$ (3.21) associated with the principal spatial filter x_- for simplicity. The results for OptNRQ$_+$ (3.25) associated with the principal spatial filter x_+ can be obtained by swapping the sign '$-$' with '$+$'.

3.3.3. Equivalent Constrained Optimizations. Due to the homogeneity of $\alpha_c(x)$ (3.23), for both $c \in \{-, +\}$, the covariance matrix $\Sigma_c(x)$ (3.22) is a homogeneous function, i.e.,

$$\Sigma_c(\gamma x) = \Sigma_c(x) \tag{3.32}$$

for any nonzero $\gamma \in \mathbb{R}$. The homogeneous property (3.32) enables OptNRQ$_-$ (3.21) to be described by the following equivalent constrained optimization

$$\begin{cases} \min & x^T \Sigma_-(x) x, \\ \text{s.t.} & x^T(\Sigma_-(x) + \Sigma_+(x))x = 1. \end{cases} \tag{3.33}$$

Alternatively, OptNRQ$_-$ (3.21) can be described by the following equivalent optimization on the unit sphere $S^{n-1} := \{x \in \mathbb{R}^n : x^T x = 1\}$:

$$\min_{x \in S^{n-1}} \frac{x^T \Sigma_-(x) x}{x^T(\Sigma_-(x) + \Sigma_+(x))x}. \tag{3.34}$$

While the constrained optimizations (3.33) and (3.34) are equivalent in the sense that an appropriate scaling of the solution of one optimization leads to the solution of the other, the algorithms for solving the two optimizations differ. In next section, we start with the optimization (3.33), and show that it can be tackled by solving the associated eigenvector-dependent nonlinear eigenvalue problem (NEPv) using the self-consistent field (SCF) iteration.

The optimization (3.34) can be addressed using matrix manifold optimization methods (refer to [**1**] for a variety of algorithms). In Section 4.2.3, we provide examples of convergence for the

Riemannian conjugate gradient algorithm and the trust region algorithm, applied to solving the optimization (3.34). We demonstrate that the SCF iteration proves to be more efficient than the two manifold optimization algorithms.

3.4. NEPv Formulations

In this section, we begin with a NEPv formulation of OptNRQ$_-$ (3.21) by utilizing the first order optimality conditions of the constrained optimization (3.33). Unfortunately, this first NEPv formulation encounters a so-called "eigenvalue ordering issue".[4] To resolve this issue, we derive an alternative NEPv by considering the second order optimality conditions of (3.33).

3.4.1. First Order Optimality Conditions.

3.4.1.1. *Preliminaries.* To facilitate the conveyance of the results of this section, we introduce a new vector $\eta_c(x)$ as

$$\eta_c(x) := \frac{1}{\|v_c(x)\|_{W_c}} W_c v_c(x). \tag{3.35}$$

Consequently, $\alpha_c(x)$ in (3.23) is

$$\alpha_c(x) = -c\delta_c \eta_c(x)$$

and $\Sigma_c(x)$ in (3.22) is

$$\Sigma_c(x) = \overline{\Sigma}_c - c\delta_c \sum_{i=1}^{m} \eta_c^{(i)}(x) V_c^{(i)}. \tag{3.36}$$

In the following Lemma, we present the gradients of functions $v_c(x)$, $\|v_c(x)\|_{W_c}$ and $\eta_c(x)$.

Lemma 3.4.1. *Let $v_c(x) \in \mathbb{R}^m$ and $\eta_c(x) \in \mathbb{R}^m$ be as defined in* (3.24) *and* (3.35)*, respectively. Then,*

(a) $\nabla v_c(x) = \begin{bmatrix} 2V_c^{(1)}x & 2V_c^{(2)}x & \cdots & 2V_c^{(m)}x \end{bmatrix}$.

(b) $\nabla \|v_c(x)\|_{W_c} = \nabla v_c(x) \eta_c(x)$.

(c) $\nabla \eta_c(x) = \frac{1}{\|v_c(x)\|_{W_c}} \Big(-\nabla v_c(x) \eta_c(x) \eta_c(x)^T + \nabla v_c(x) W_c \Big)$.

[4]Given an eigenvalue λ, we use the terminology *eigenvalue ordering* to define the position of λ among the eigenvalues $\lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$.

PROOF. (a) For $i = 1, 2, \ldots, m$, the i^{th} column of $\nabla v_c(x) \in \mathbb{R}^{n\times m}$ is $\nabla v_c^{(i)}(x) \in \mathbb{R}^n$, the gradient of the i^{th} component of $v_c(x)$. By Lemma A.0.1(b), $\nabla v_c^{(i)}(x) = \nabla(x^T V_c^{(i)} x) = 2V_c^{(i)} x$, and the result follows.

(b) By definitions,

$$\nabla \|v_c(x)\|_{W_c} = \nabla\sqrt{v_c(x)^T W_c v_c(x)} = \frac{\nabla(v_c(x)^T W_c v_c(x))}{2\sqrt{v_c(x)^T W_c v_c(x)}} = \frac{2\nabla v_c(x) W_c v_c(x)}{2\sqrt{v_c(x)^T W_c v_c(x)}} \tag{3.37}$$

$$= \nabla v_c(x)\eta_c(x) \tag{3.38}$$

by using the chain rule and Lemma A.0.2(a) in (3.37), and the definition of $\eta_c(x)$ (3.35) in (3.38).

(c) We apply Lemma A.0.2(c) with $a(x) = \frac{1}{\|v_c(x)\|_{W_c}}$ and $y(x) = W_c v_c(x)$ to obtain

$$\nabla \eta_c(x) = \nabla\left(\frac{1}{\|v_c(x)\|_{W_c}}\right)\left(v_c(x)^T W_c\right) + \left(\frac{1}{\|v_c(x)\|_{W_c}}\right)\nabla\left(W_c v_c(x)\right). \tag{3.39}$$

By the chain rule and applying the result $\nabla\|v_c(x)\|_{W_c}$ in Lemma 3.4.1(b), we have

$$\nabla\left(\frac{1}{\|v_c(x)\|_{W_c}}\right) = -\frac{1}{\|v_c(x)\|_{W_c}^2}\nabla v_c(x)\eta_c(x). \tag{3.40}$$

For $\nabla(W_c v_c(x))$, note that

$$W_c v_c(x) = \begin{bmatrix} w_c^{(1)}(x^T V_c^{(1)} x) \\ w_c^{(2)}(x^T V_c^{(2)} x) \\ \vdots \\ w_c^{(m)}(x^T V_c^{(m)} x) \end{bmatrix}$$

so that its gradient is

$$\nabla(W_c v_c(x)) = \begin{bmatrix} 2w_c^{(1)} V_c^{(1)} x & 2w_c^{(2)} V_c^{(2)} x & \cdots & 2w_c^{(m)} V_c^{(m)} x \end{bmatrix} = \nabla v_c(x) W_c \tag{3.41}$$

by using $\nabla v_c(x)$ in Lemma 3.4.1(a).

With results (3.40) and (3.41), $\nabla \eta_c(x)$ (3.39) is

$$\begin{aligned}\nabla \eta_c(x) &= -\frac{1}{\|v_c(x)\|_{W_c}^2}\nabla v_c(x)\eta_c(x)v_c(x)^T W_c + \frac{1}{\|v_c(x)\|_{W_c}}\nabla v_c(x) W_c \\ &= -\frac{1}{\|v_c(x)\|_{W_c}}\nabla v_c(x)\eta_c(x)\eta_c(x)^T + \frac{1}{\|v_c(x)\|_{W_c}}\nabla v_c(x) W_c. \qquad (3.42)\end{aligned}$$

where the equality (3.42) follows from the definition of $\eta_c(x)$ (3.35).

□

We introduce the following quantity $s_c(x)$ for representing the quadratic form of the matrix $\Sigma_c(x)$:

$$s_c(x) := x^T \Sigma_c(x) x, \qquad \text{for both } c \in \{-, +\}. \qquad (3.43)$$

Lemma 3.4.2. *Let $s_c(x)$ be the quadratic form as defined in* (3.43). *Then, its gradient is*

$$\nabla s_c(x) = 2\Sigma_c(x)x. \qquad (3.44)$$

Proof. By the definition of $\Sigma_c(x)$ (3.36) and the definition of $v_c(x)$ (3.24), we have

$$\begin{aligned}s_c(x) &= x^T \overline{\Sigma}_c x - c\delta_c \sum_{i=1}^{m} \eta_c^{(i)}(x)(x^T V_c^{(i)} x) = x^T \overline{\Sigma}_c x - c\delta_c \eta_c(x)^T v_c(x) \\ &= x^T \overline{\Sigma}_c x - c\delta_c \|v_c(x)\|_{W_c} \qquad (3.45)\end{aligned}$$

where the second term $\eta_c(x)^T v_c(x)$ was simplified to $\|v_c(x)\|_{W_c}$.

By Lemma A.0.1(b), the gradient of the first term $x^T \overline{\Sigma}_c x$ of (3.45) is $2\overline{\Sigma}_c x$. For the second term, we use the result $\nabla \|v_c(x)\|_{W_c} = \nabla v_c(x)\eta_c(x)$ in Lemma 3.4.1(b). Therefore, by combining the gradients of the two terms in (3.45), the gradient of $s_c(x)$ is

$$\begin{aligned}\nabla s_c(x) &= 2\overline{\Sigma}_c x - c\delta_c \nabla v_c(x)\eta_c(x) \qquad (3.46)\\ &= 2\overline{\Sigma}_c x - 2c\delta_c \sum_{i=1}^{m} \eta_c(x) V_c^{(i)} x \qquad (3.47)\\ &= 2\Sigma_c(x)x. \qquad (3.48)\end{aligned}$$

where the equality (3.47) follows from the result $\nabla v_c(x)$ in Lemma 3.4.1(a) and the equality (3.48) follows from the definition of $\Sigma_c(x)$ (3.36). $\square$

3.4.1.2. *NEPv with Unknown Eigenvalue Ordering.* We proceed to state the first order necessary conditions of OptNRQ$_-$ (3.21) in Theorem 3.4.1 and derive a NEPv.

THEOREM 3.4.1. *Under Assumption 3.3.1, if x is a local minimizer of OptNRQ$_-$* (3.21)*, then x is an eigenvector of the NEPv:*

$$\begin{cases} \Sigma_-(x)x = \lambda(\Sigma_-(x) + \Sigma_+(x))x \\ x^T(\Sigma_-(x) + \Sigma_+(x))x = 1, \end{cases} \tag{3.49}$$

for some eigenvalue $\lambda > 0$.

PROOF. Let us consider the equivalent problem, the constrained optimization (3.33) of OptNRQ$_-$ (3.21). The Lagrangian function of the constrained optimization (3.33) is defined as

$$L(x, \lambda) = x^T\Sigma_-(x)x - \lambda(x^T(\Sigma_-(x) + \Sigma_+(x))x - 1). \tag{3.50}$$

By Lemma 3.4.2, the gradient of (3.50) with respect to x is

$$\nabla_x L(x, \lambda) = 2\Sigma_-(x)x - 2\lambda(\Sigma_-(x) + \Sigma_+(x))x. \tag{3.51}$$

Then, the first-order necessary conditions [**82**, Theorem 12.1, p.321] of the constrained optimization (3.33),

$$\nabla_x L(x, \lambda) = 0 \quad \text{and} \quad x^T(\Sigma_-(x) + \Sigma_+(x))x = 1,$$

imply that a local minimizer x is an eigenvector of the NEPv (3.49).

The eigenvalue λ is positive due to the positive definiteness of $\Sigma_c(x)$ under Assumption 3.3.1. $\square$

REMARK 3.4.1. *Note that while a local minimizer of OptNRQ$_-$* (3.21) *is an eigenvector of the NEPv* (3.49) *corresponding to some eigenvalue $\lambda > 0$, the eigenvalue ordering is unknown. That is, we do not know in advance which of the n eigenvalues of the matrix pair $(\Sigma_-(x), \Sigma_-(x) + \Sigma_+(x))$ corresponds to the eigenvalue λ. To address this, we introduce an alternative NEPv for which the eigenvalue ordering of λ is known in advance.*

3.4.2. Second Order Optimality Conditions.

3.4.2.1. *Preliminaries.*

LEMMA 3.4.3. *Let $s_c(x)$ be the quadratic form as defined in* (3.43). *Then, its Hessian matrix is*

$$\nabla^2 s_c(x) = 2(\Sigma_c(x) + \widetilde{\Sigma}_c(x)), \tag{3.52}$$

where

$$\widetilde{\Sigma}_c(x) := -\frac{1}{2} c\delta_c \nabla\eta_c(x)\nabla v_c(x)^T \tag{3.53}$$

with $\nabla\eta_c(x)$ and $\nabla v_c(x)$ as defined in Lemma 3.4.1(c) and Lemma 3.4.1(a), respectively.

PROOF. We compute the gradients of the two terms of $\nabla s_c(x)$ in (3.46) to compute the Hessian matrix $\nabla^2 s_c(x)$. The gradient of the first term $2\overline{\Sigma}_c x$ of (3.46) is $2\overline{\Sigma}_c$ by Lemma A.0.1(a). For the second term, we apply Lemma A.0.2(b) with $B(x) = \nabla v_c(x)$ and $y(x) = \eta_c(x)$:

$$\nabla(\nabla v_c(x)\eta_c(x)) = \sum_{i=1}^{m} (\eta_c^{(i)}(x))(2V_c^{(i)}) + \nabla\eta_c(x)\nabla v_c(x)^T.$$

Therefore, by combining the gradients of the two terms in (3.46) together, we have

$$\begin{aligned}
\nabla^2 s_c(x) &= 2\overline{\Sigma}_c - 2c\delta_c \sum_{i=1}^{m} \eta_c^{(i)}(x) V_c^{(i)} - c\delta_c \nabla\eta_c(x)\nabla v_c(x)^T \\
&= 2\left(\overline{\Sigma}_c - c\delta_c \sum_{i=1}^{m} \eta_c^{(i)}(x) V_c^{(i)} - \frac{1}{2} c\delta_c \nabla\eta_c(x)\nabla v_c(x)^T\right) \\
&= 2(\Sigma_c(x) + \widetilde{\Sigma}_c(x)) \qquad (3.54)
\end{aligned}$$

where the equality (3.54) follows from the definition of $\Sigma_c(x)$ (3.36) and $\widetilde{\Sigma}_c(x)$ (3.53). □

LEMMA 3.4.4. *Let $G_c(x) \in \mathbb{R}^{n\times n}$ be defined as*

$$G_c(x) := \Sigma_c(x) + \widetilde{\Sigma}_c(x). \tag{3.55}$$

Then, the following properties for $G_c(x)$ hold:

(a) *$G_c(x)$ is symmetric.*

(b) *$G_c(x)x = \Sigma_c(x)x$.*

(c) Under Assumption 3.3.1, $G_-(x) \succ 0$.

PROOF. (a) Since $\Sigma_c(x)$ is symmetric, we only need to show that $\widetilde{\Sigma}_c(x)$ is symmetric. From the result of $\nabla\eta_c(x)$ in Lemma 3.4.1(c) and $\nabla v_c(x)$ in Lemma 3.4.1(a), we obtain

$$\nabla\eta_c(x)\nabla v_c(x)^T = -\frac{1}{\|v_c(x)\|_{W_c}}\Big(\nabla v_c(x)\eta_c(x)\eta_c(x)^T\nabla v_c(x)^T - \nabla v_c(x)W_c\nabla v_c(x)^T\Big), \tag{3.56}$$

which is a symmetric matrix in $\mathbb{R}^{n\times n}$. Therefore, by the definition of $\widetilde{\Sigma}_c(x)$ in (3.53), $\widetilde{\Sigma}_c(x)$ is symmetric.

(b) We prove that $\widetilde{\Sigma}_c(x)x = 0$. From the equation $\nabla\eta_c(x)\nabla v_c(x)^T$ in (3.56), we have the following:

$$\nabla\eta_c(x)\nabla v_c(x)^T x = -\frac{1}{\|v_c(x)\|_{W_c}}\Big(2\nabla v_c(x)\eta_c(x)\eta_c(x)^T v_c(x) - 2\nabla v_c(x)W_c v_c(x)\Big) \tag{3.57}$$

$$= -2\nabla v_c(x)\eta_c(x) + 2\nabla v_c(x)\eta_c(x) \tag{3.58}$$

$$= 0$$

by applying $\nabla v_c(x)^T x = 2v_c(x)$ for the equality (3.57), and applying $\eta_c(x)^T v_c(x) = \|v_c(x)\|_{W_c}$ and the definition of $\eta_c(x)$ (3.35) for the equality (3.58). Therefore, by the definition of $\widetilde{\Sigma}_c(x)$ in (3.53), $\widetilde{\Sigma}_c(x)x = 0$.

(c) We first show that $\widetilde{\Sigma}_-(x)$ is positive semi-definite while $\widetilde{\Sigma}_+(x)$ is negative semi-definite. Then, since $G_-(x)$ is defined as the sum of $\Sigma_-(x)$, a positive definite matrix, and $\widetilde{\Sigma}_-(x)$, a positive semi-definite matrix, it is positive definite.

By the definition of $\widetilde{\Sigma}_c(x)$ in (3.53) and the equation of $\nabla\eta_c(x)\nabla v_c(x)^T$ in (3.56), we have

$$\widetilde{\Sigma}_c(x) = \frac{1}{2}c\delta_c \cdot \frac{1}{\|v_c(x)\|_{W_c}}\Big(\nabla v_c(x)\eta_c(x)\eta_c(x)^T\nabla v_c(x)^T - \nabla v_c(x)W_c\nabla v_c(x)^T\Big) \tag{3.59}$$

for both $c \in \{-,+\}$. We consider the definiteness on the matrix inside the parenthesis of (3.59). That is, for a nonzero $y \in \mathbb{R}^n$ we check the sign of the quantity

$$y^T\Big(\nabla v_c(x)\eta_c(x)\eta_c(x)^T\nabla v_c(x)^T - \nabla v_c(x)W_c\nabla v_c(x)^T\Big)y. \tag{3.60}$$

By the definition of $\eta_c(x)$ (3.35) and the equation $\nabla v_c(x)$ in Lemma 3.4.1(a), we can show that the quantity (3.60) is equal to

$$\frac{4}{\|v_c(x)\|_{W_c}^2}\left(\sum_{i=1}^m w_c^{(i)}(x^TV_c^{(i)}x)(y^TV_c^{(i)}x)\right)^2 - 4\sum_{i=1}^m w_c^{(i)}(y^TV_c^{(i)}x)^2$$
$$= 4\left[\frac{\left(\sum_{i=1}^m w_c^{(i)}(x^TV_c^{(i)}x)(y^TV_c^{(i)}x)\right)^2}{\sum_{i=1}^m w_c^{(i)}(x^TV_c^{(i)}x)^2} - \sum_{i=1}^m w_c^{(i)}(y^TV_c^{(i)}x)^2\right]. \tag{3.61}$$

With vectors $a_c, b_c \in \mathbb{R}^m$ defined as

$$a_c := \begin{bmatrix} \sqrt{w_c^{(1)}}(x^TV_c^{(1)}x) \\ \sqrt{w_c^{(2)}}(x^TV_c^{(2)}x) \\ \vdots \\ \sqrt{w_c^{(m)}}(x^TV_c^{(m)}x) \end{bmatrix}, \qquad b_c := \begin{bmatrix} \sqrt{w_c^{(1)}}(y^TV_c^{(1)}x) \\ \sqrt{w_c^{(2)}}(y^TV_c^{(2)}x) \\ \vdots \\ \sqrt{w_c^{(m)}}(y^TV_c^{(m)}x) \end{bmatrix}, \tag{3.62}$$

the quantity (3.61) is further simplified as:

$$4\left[\frac{(a_c^Tb_c)^2}{\|a_c\|_2^2} - \|b_c\|_2^2\right]. \tag{3.63}$$

By the Cauchy-Schwarz inequality we have $(a_c^Tb_c)^2 \leqslant \|a_c\|_2^2 \cdot \|b_c\|_2^2$ so that

$$4\left[\frac{(a_c^Tb_c)^2}{\|a_c\|_2^2} - \|b_c\|_2^2\right] \leqslant 4\left[\frac{\|a_c\|_2^2 \cdot \|b_c\|_2^2}{\|a_c\|_2^2} - \|b_c\|_2^2\right] = 0. \tag{3.64}$$

Therefore, according to the equation of $\widetilde{\Sigma}_c(x)$ in (3.59), for any nonzero $y \in \mathbb{R}^m$ we have

$$y^T\widetilde{\Sigma}_-(x)y \geqslant 0, \qquad y^T\widetilde{\Sigma}_+(x)y \leqslant 0.$$

Therefore, $\widetilde{\Sigma}_-(x)$ is positive semi-definite and $\widetilde{\Sigma}_+(x)$ is negative semi-definite.

□

3.4.2.2. *NEPv with Known Eigenvalue Ordering.* Utilizing Lemma 3.4.4, we derive another NEPv. The eigenvalue ordering of this alternative NEPv can be established using the second order optimality conditions of OptNRQ$_-$ (3.21).

THEOREM 3.4.2. *Under Assumption 3.3.1,* *(i) If x is a local minimizer of OptNRQ$_-$* (3.21)*, then x is an eigenvector of the NEPv, referred to as **OptNRQ-nepv**:*

$$\begin{cases} G_-(x)x = \lambda\big(G_-(x) + G_+(x)\big)x \\ x^T\big(G_-(x) + G_+(x)\big)x = 1, \end{cases} \tag{3.65}$$

corresponding to the smallest positive eigenvalue λ.

(ii) If λ is the smallest positive eigenvalue of OptNRQ-nepv (3.65) *and is simple, then the corresponding eigenvector x of OptNRQ-nepv* (3.65) *is a strict local minimizer of OptNRQ$_-$* (3.21).

Proof. For result (i): Let x be a local minimizer of the equivalent problem the constrained optimization (3.33) of OptNRQ$_-$ (3.21). By Theorem 3.4.1, x is an eigenvector of the NEPv (3.49) for some eigenvalue $\lambda > 0$. By utilizing the property $G_c(x)x = \Sigma_c(x)x$ in Lemma 3.4.4(b), x is also an eigenvector of OptNRQ-nepv (3.65) corresponding to the same eigenvalue $\lambda > 0$.

Next, we show that λ must be the smallest positive eigenvalue of OptNRQ-nepv (3.65) using the second order necessary conditions of a constrained optimization [**82**, Theorem 12.5, p.332].

Both $G_-(x)$ and $G_+(x)$ are symmetric by Lemma 3.4.4(a), and $G_-(x) \succ 0$ by Lemma 3.4.4(c). The positive definiteness of $G_+(x)$ and $G_-(x) + G_+(x)$, however, are not guaranteed. This implies that the eigenvalues of the matrix pair $(G_-(x), G_-(x)+G_+(x))$ are real (possibly including infinity) but not necessarily positive. We denote the eigenvalues as $\lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ and the corresponding eigenvectors as $v_1, v_2, \ldots, v_n$ so that

$$G_-(x)v_i = \lambda_i(G_-(x) + G_+(x))v_i, \quad \text{for} \quad i = 1, \ldots, n. \tag{3.66}$$

We take the eigenvectors to be $G_-(x)$-orthogonal, i.e.,

$$v_i^T G_-(x) v_j = \begin{cases} 1 & \text{if} \quad i = j \\ 0 & \text{otherwise.} \end{cases} \tag{3.67}$$

As (λ, x) is an eigenpair of OptNRQ-nepv (3.65), we have

$$\lambda = \lambda_{j_0}, \quad x = v_{j_0} \tag{3.68}$$

for some $1 \leqslant j_0 \leqslant n$. We will deduce the order of λ from the second order necessary conditions of the constrained optimization (3.33) for a local minimizer x:

$$z^T \nabla_{xx} L(x,\lambda) z \geqslant 0, \quad \forall z \in \mathbb{R}^n \quad \text{such that} \quad z^T(\Sigma_-(x) + \Sigma_+(x))x = 0 \tag{3.69}$$

where $L(x,\lambda)$ is the Lagrangian function as defined in (3.50). Let $z \in \mathbb{R}^n$ be a vector satisfying the condition (3.69), i.e., $z^T(\Sigma_-(x) + \Sigma_+(x))x = 0$. We can write the expansion of z in terms of the eigenbasis $\{v_i\}_{i=1}^n$:

$$z = \sum_{i=1}^n a_i v_i$$

where $a_i = z^T G_-(x) v_i$ by using (3.67). In particular, we have $a_{j_0} = z^T G_-(x) v_{j_0} = \lambda_{j_0} z^T (G_-(x) + G_+(x)) v_{j_0} = \lambda z^T(\Sigma_-(x) + \Sigma_+(x))x = 0$ by (3.66) and (3.68). So, z can be written as

$$z = \sum_{i=1, i\neq j_0}^n a_i v_i.$$

We use Lemma 3.4.3 to compute the gradient of $\nabla_x L(x,\lambda)$ in (3.51):

$$\nabla_{xx} L(x,\lambda) = 2G_-(x) - 2\lambda(G_-(x) + G_+(x)).$$

So, the inequality of (3.69) becomes

$$2 \cdot z^T \Big(G_-(x) - \lambda(G_-(x) + G_+(x)) \Big) z = 2 \sum_{i=1, i\neq j_0}^n a_i^2 \left(1 - \frac{\lambda_{j_0}}{\lambda_i} \right) \geqslant 0 \tag{3.70}$$

by using (3.67) and substituting $\lambda = \lambda_{j_0}$ from (3.68). Since (3.70) holds for arbitrary a_i, the inequality (3.70) implies that $1 - \frac{\lambda_{j_0}}{\lambda_i} \geqslant 0, \quad \forall i = 1, \ldots, n$ with $i \neq j_0$. As $\lambda = \lambda_{j_0} > 0$, this establishes $\lambda_i \geqslant \lambda$ for all positive eigenvalues λ_i.

For result (ii): If $\lambda = \lambda_{j_0}$ is the smallest positive eigenvalue of OptNRQ-nepv (3.65), and λ is simple, then for the corresponding eigenvector x of OptNRQ-nepv (3.65) we have

$$z^T \nabla_{xx} L(x,\lambda) z > 0, \quad \forall z \in \mathbb{R}^n \quad \text{such that} \quad z^T(\Sigma_-(x) + \Sigma_+(x))x = 0 \quad \text{and} \quad z \neq 0$$

which is precisely the second order sufficient conditions [**82**, Theorem 12.6, p.333] for x to be a strict local minimizer of the constrained optimization (3.33). Consequently, x is a strict local minimizer of OptNRQ$_-$ (3.21). $\square$

In the following example, we highlight the contrast in certainty concerning the order of the eigenvalue λ with respect to the NEPv (3.49) and OptNRQ-nepv (3.65). Specifically, we demonstrate that while the order of λ for the NEPv (3.49) cannot be predetermined, it consistently corresponds to the smallest positive eigenvalue for OptNRQ-nepv (3.65).

Example 3.4.1. (Eigenvalue ordering issue.)

We consider solving OptNRQ$_c$ ((3.21) and (3.25)), for conditions $c \in \{-,+\}$, using the synthetic dataset (see Section 4.2.1 for its description). We generate trials $Y_c^{(i)} \in \mathbb{R}^{10\times 200}$ with 50 trials for each condition. For both conditions, we choose two out of the 10 sources to be discriminative. That is, we have $s^{n_1} \in \mathbb{R}^8$ for the nondiscriminative sources and $s^{n_2} \in \mathbb{R}^2$ for the discriminative sources. The first discriminative source is sampled from $\mathcal{N}(0, 0.2)$[5] for condition '$-$' and $\mathcal{N}(0, 1.8)$ for condition '$+$'. The second discriminative source is sampled from $\mathcal{N}(0, 1.4)$ for condition '$-$' and $\mathcal{N}(0, 0.6)$ for condition '$+$'. For both conditions, all 8 nondiscriminative sources are sampled from $\mathcal{N}(0, 1)$ and the 10 nonstationary noise sources of ϵ are sampled from $\mathcal{N}(0, 2)$. Finally, the tolerance set radius is set as $\delta_c = 6$ for both conditions.

Let $\widehat{x}_-$ and $\widehat{x}_+$ denote the solutions of OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25), respectively. These solutions are obtained using the algorithm OptNRQ-nepv (discussed in Section 4.1.2) and verified by the Riemannian trust region algorithm.

Consider the NEPv (3.49). For $\widehat{x}_-$, the first two eigenvalues of the matrix pair $(\Sigma_-(\widehat{x}_-), \Sigma_-(\widehat{x}_-) + \Sigma_+(\widehat{x}_-))$ are

$$\lambda_1 = 0.3486, \quad \underline{\lambda_2 = 0.4286},$$

where the eigenvector corresponding to the underlined eigenvalue is $\widehat{x}_-$. Meanwhile, for $\widehat{x}_+$, the first few eigenvalues of the matrix pair $(\Sigma_+(\widehat{x}_+), \Sigma_-(\widehat{x}_+) + \Sigma_+(\widehat{x}_+))$ are

$$\lambda_1 = 0.3035, \quad \ldots, \quad \underline{\lambda_6 = 0.5120},$$

[5]$\mathcal{N}(\mu, \sigma^2)$ denotes the Gaussian distribution where μ is the mean and σ^2 is the variance.

where the eigenvector corresponding to the underlined eigenvalue is $\widehat{x}_+$. As we can see, the eigenvalue ordering of the NEPv (3.49) for $\widehat{x}_-$ is different from that of $\widehat{x}_+$. Based on this observation, we conclude that we could not have known in advance the order of the eigenvalue of the NEPv (3.49).

Next, let us consider the OptNRQ-nepv (3.65). For $\widehat{x}_-$, the first two eigenvalues of the matrix pair $(G_-(\widehat{x}_-), G_-(\widehat{x}_-) + G_+(\widehat{x}_-))$ are

$$\underline{\lambda_1 = 0.4286}, \quad \lambda_2 = 0.5155,$$

where the eigenvector corresponding to the underlined eigenvalue is $\widehat{x}_-$. Meanwhile, for $\widehat{x}_+$, the first three eigenvalues of the matrix pair $(G_+(\widehat{x}_+), G_-(\widehat{x}_+) + G_+(\widehat{x}_+))$ are

$$\lambda_1 = -4.7463, \quad \underline{\lambda_2 = 0.5120}, \quad \lambda_3 = 0.5532,$$

where the eigenvector corresponding to the underlined eigenvalue is $\widehat{x}_+$. As we can see, for both $\widehat{x}_c$, the corresponding eigenvalue is the smallest positive eigenvalue of the OptNRQ-nepv (3.65). $\square$

Remark 3.4.2. We do not pursue solving the NEPv (3.49) due to the challenge posed by its unknown eigenvalue ordering. This uncertainty makes it difficult to determine which eigenvector of the NEPv (3.49) to compute. Instead, we opt to solve OptNRQ-nepv (3.65), for which we can always compute the eigenvector associated with the smallest positive eigenvalue.

CHAPTER 4

Robust Common Spatial Pattern Analysis: Algorithms and Numerical Results

4.1. Algorithms

In this section, we discuss the algorithms for OptNRQ$_-$ (3.21) to compute the principal spatial filter x_-. The algorithms for OptNRQ$_+$ (3.25) associated with the principal spatial filter x_+ is obtained by swapping the sign '$-$' and '$+$'. We consider two algorithms. The first algorithm is a fixed-point type iteration scheme on OptNRQ$_-$ (3.21). The second algorithm employs the self-consistent field (SCF) iteration on OptNRQ-nepv (3.65) formulation of OptNRQ$_-$ (3.21).

4.1.1. Fixed-point Iteration. As presented in [**59**], one natural idea for solving OptNRQ$_-$ (3.21) is using the following fixed-point type iteration:

$$x_{k+1} \longleftarrow \operatorname*{argmin}_{x\neq 0} \frac{x^T\Sigma_-(x_k)x}{x^T(\Sigma_-(x_k)+\Sigma_+(x_k))x}. \tag{4.1}$$

The minimization in (4.1) is a Rayleigh quotient optimization. We refer to this scheme as **OptNRQ-fp**. By Theorem 2.1.1 (the Courant-Fischer minimax theorem), x_{k+1} of OptNRQ-fp (4.1) is the eigenvector of the eigenvalue problem

$$\Sigma_-(x_k)\,x = \lambda\big(\Sigma_-(x_k)+\Sigma_+(x_k)\big)\,x, \tag{4.2}$$

corresponding to the smallest eigenvalue. In other words, OptNRQ-fp (4.1) is equivalent to

$$x_{k+1} \longleftarrow \text{an eigenvector w.r.t. the smallest eigenvalue of } \big(\Sigma_-(x_k), \Sigma_-(x_k)+\Sigma_+(x_k)\big). \tag{4.3}$$

This is the self-consistent field (SCF) iteration for solving the NEPv (3.49) where the corresponding eigenvalue is the smallest.

There are two issues with OptNRQ-fp (4.3):

(i) OptNRQ-fp (4.3) often fails to converge and arrives at suboptimal solutions (illustrated in Section 4.2.3).

(ii) As illustrated in Example 3.4.1, the solution of OptNRQ$_-$ (3.21) may not be an eigenvector with respect to the smallest eigenvalue of the NEPv (3.49). Consequently, even when OptNRQ-fp (4.3) converges, its converged vector may not be the solution of OptNRQ$_-$ (3.21).

4.1.2. Self-consistent Field Iteration.

4.1.2.1. *Plain SCF.* Applied to OptNRQ-nepv (3.65), the SCF iteration is given by

(4.4)

$$x_{k+1} \longleftarrow \text{an eigenvector w.r.t. the smallest positive eigenvalue of } (G_-(x_k), G_-(x_k) + G_+(x_k)).$$

The converged solution of the SCF iteration (4.4) is an eigenvector with respect to the smallest positive eigenvalue of OptNRQ-nepv (3.65). According to Theorem 3.4.2, if the corresponding eigenvalue is simple, then the solution is guaranteed to be a strict local minimizer of OptNRQ$_-$ (3.21).

4.1.2.2. *Line search.* An issue with the plain SCF iteration (4.4) is that the value of the objective function $q_-(x)$ of OptNRQ$_-$ (3.21) may not decrease monotonically. That is, denoting $\widetilde{x}_{k+1}$ as the next iterate of x_k in the plain SCF iteration (4.4), it may occur that $q_-(\widetilde{x}_{k+1}) \nless q_-(x_k)$.

To address this issue, we employ the line search method. In the line search method, the search direction d_k is computed, and the step size β is determined to specify the distance to move along the search direction from the current iterate x_k. The updated iterate is then given by

$$x_{k+1} = x_k + \beta d_k$$

to satisfy the condition

$$q_-(x_{k+1}) < q_-(x_k). \tag{4.5}$$

The effectiveness of the line search method relies on making appropriate choices of the search direction d_k and the step size β. A desirable search direction is a descent direction, in which case we can find a step size β such that $x_{k+1} = x_k + \beta d_k$ satisfies (4.5) [**82**, p.21,22]. d_k is descent direction of $q_-(x)$ at x_k if

$$d_k^T \nabla q_-(x_k) < 0, \tag{4.6}$$

where, by applying Lemma 3.4.2, the gradient of $q_-(x)$ is

$$\nabla q_-(x) = \frac{2}{x^T(\Sigma_-(x) + \Sigma_+(x))x}\Big(\Sigma_-(x) - q_-(x)(\Sigma_-(x) + \Sigma_+(x))\Big)x. \tag{4.7}$$

In the following, we discuss a strategy for choosing a descent direction d_k. In this strategy, we first consider the search direction to be

$$d_k = \widetilde{x}_{k+1} - x_k. \tag{4.8}$$

d_k (4.8) is always a descent direction (with a possible sign change of $\widetilde{x}_{k+1}$) except for a rare case. To see this, note that for the direction (4.8) we have

$$\begin{aligned} \frac{1}{2} d_k^T \nabla q_-(x_k) &= \big(\lambda_{k+1} - q_-(x_k)\big) \frac{\widetilde{x}_{k+1}^T(\Sigma_-(x_k) + \Sigma_+(x_k))x_k}{x_k^T(\Sigma_-(x_k) + \Sigma_+(x_k))x_k} \\ &= \frac{t_k}{x_k^T(\Sigma_-(x_k) + \Sigma_+(x_k))x_k}, \end{aligned}$$

where

$$t_k := (\lambda_{k+1} - q_-(x_k)) \cdot \widetilde{x}_{k+1}^T(\Sigma_-(x_k) + \Sigma_+(x_k))x_k. \tag{4.9}$$

λ_{k+1} is the smallest positive eigenvalue of the matrix pair $(G_-(x_k), G_-(x_k) + G_+(x_k))$. Since $x_k^T(\Sigma_-(x_k) + \Sigma_+(x_k))x_k$ is always positive, the sign of t_k determines whether d_k (4.8) is a descent direction.

- t_k is negative. In this case, $d_k = \widetilde{x}_{k+1} - x_k$ is a descent direction.
- t_k is positive. In this case, we can simply scale $\widetilde{x}_{k+1}$ by -1, i.e.,

$$\widetilde{x}_{k+1} \longleftarrow -\widetilde{x}_{k+1}, \tag{4.10}$$

  which will ensure that $d_k = \widetilde{x}_{k+1} - x_k$ is a descent direction.
- t_k is zero. We can assume that $\lambda_{k+1} - q_-(x_k) \neq 0$, as otherwise, it would indicate that the iteration has already converged. Thus, only in the rare case where $\widetilde{x}_{k+1}^T(\Sigma_-(x_k) +$

$\Sigma_+(x_k))x_k = 0$ we have that $t_k = 0$. In this rare case, or more generally when $\widetilde{x}_{k+1}^T(\Sigma_-(x_k)+\Sigma_+(x_k))x_k$ is close to zero, we reset the search direction as the steepest-descent direction

$$d_k = -\frac{\nabla q_-(x_k)}{\|\nabla q_-(x_k)\|^2}. \tag{4.11}$$

By adopting this strategy, we ensure that we can always choose a descent direction. Moreover, we scarcely choose the steepest-descent direction (4.11) and avoid replicating the steepest descent method, which can suffer from slow convergence [**82**].

We follow the standard line search practice where an inexact step size β is computed to satisfy the following sufficient decrease condition (also known as Armijo condition):

$$q_-(x_k + \beta d_k) \leqslant q_-(x_k) + \mu\beta \cdot d_k^T \nabla q_-(x_k), \tag{4.12}$$

for some $\mu \in (0, 1)$. As the objective function $q_-(x)$ is uniformly bounded below by zero, there exists a β satisfying the sufficient decrease condition (4.12) [**82**, Lemma 3.1].

To obtain the inexact step size β, we apply the Armijo backtracking [**82**, Algorithm 3.1, p.37], which ensures that the step size β is short enough to have a sufficient decrease but not too short. We start with an initial $\beta = 1$ and update the step length as $\beta \longleftarrow \tau\beta$, for some $\tau \in (0, 1)$, until the sufficient decrease condition (4.12) is satisfied.

4.1.2.3. *Algorithm pseudocode.* The algorithm for the SCF iteration (4.4) with line search is shown in Algorithm 2. We refer to Algorithm 2 as **OptNRQ-nepv**.

Algorithm 2 OptNRQ-nepv

Input: Initial $x_0 \in \mathbb{R}^n$, line search factors $\mu, \tau \in (0,1)$ (e.g. $\mu = \tau = 0.01$), tolerance *tol*.

Output: Approximate solution $\widehat{x}_-$ to OptNRQ$_-$ (3.21).

```
1:  for $k = 0, 1, \ldots$ do
2:    Set $A_k = G_-(x_k)$, $B_k = G_-(x_k) + G_+(x_k)$, and $q_k = \frac{x_k^T A_k x_k}{x_k^T B_k x_k}$.
3:    if $\frac{||A_k x_k - q_k B_k x_k||_2}{||A_k x_k||_2 + q_k ||B_k x_k||_2} <$ tol then
4:      return $\widehat{x}_- = x_k$.
5:    end if
6:    Compute the smallest positive eigenvalue and eigenvector $(\lambda_{k+1}, x_{k+1})$ of $(A_k, B_k)$.
7:    if $q_-(x_{k+1}) \geqslant q_-(x_k)$ then
8:      Set $t_k = (\lambda_{k+1} - q_-(x_k)) \cdot x_{k+1}^T B_k x_k$.
9:      if $|t_k| <$ tol then
10:       Set $d_k = -\frac{\nabla q_-(x_k)}{\|\nabla q_-(x_k)\|^2}$.
11:     else if $t_k > 0$ then
12:       Set $d_k = -x_{k+1} - x_k$.
13:     else
14:       Set $d_k = x_{k+1} - x_k$.
15:     end if
16:     Set $\beta = 1$.
17:     while $q_-(x_k + \beta d_k) > q_-(x_k) + \mu\beta \cdot d_k^T \nabla q_-(x_k)$ do
18:       $\beta := \tau\beta$.
19:     end while
20:     $x_{k+1} := x_k + \beta d_k$.
21:   end if
22: end for
```

4.1.2.4. *Implementation details.*

- In line 3, the stopping criterion checks whether OptNRQ-nepv (3.65) is satisfied.
- The smallest positive eigenpair in line 6 can be efficiently computed using sparse eigensolvers such as the implicitly restarted Arnoldi method [**100**].

- If the value of the objective function $q_-(x)$ does not decrease at the next iterate (line 7), the algorithm takes appropriate steps (lines 8 to 15) to find a descent direction, and the Armijo backtracking (lines 16 to 19) is used to obtain an appropriate step length β, and consequently, the next iterate is updated in line 20.

REMARK 4.1.1. While a formal convergence proof of OptNRQ-nepv (Algorithm 2) is still subject to further study, our numerical experiments demonstrate both convergence and quadratic local convergence. For the proof of quadratic local convergence for the SCF iteration pertaining to a general robust Rayleigh quotient optimization, we refer the reader to [**11**].

4.2. Numerical Experiments

The numerical experiments section is organized into four subsections. In the first subsection, we provide descriptions of the synthetic datasets and real-world BCI datasets used in the experiments. The specific experimental settings are outlined in the second subsection. In the third subsection, we present the convergence behavior and running time of OptNRQ-nepv (Algorithm 2), OptNRQ-fp (4.1), and manifold optimization algorithms. Lastly, the fourth subsection presents the classification results obtained using OptNRQ-nepv, OptNRQ-fp, and CSP for both the synthetic and real-world datasets.

4.2.1. Datasets.

4.2.1.1. *Synthetic Dataset.* The synthetic dataset, utilized in [**11**, **59**], comprises of trials

$$Y_c^{(i)} = \begin{bmatrix} x_1 & x_2 & \cdots x_t \end{bmatrix} \in \mathbb{R}^{n \times t},$$

where, for $i = 1, 2, \ldots, t$, $x_i \in \mathbb{R}^n$ is a synthetic signal of n channels. The synthetic signals are randomly generated using the following linear mixing model:

$$x_i = A \begin{bmatrix} s^{n_1} \\ s^{n_2} \end{bmatrix} + \epsilon. \tag{4.13}$$

The matrix $A \in \mathbb{R}^{n \times n}$ is a random rotation matrix that simulates the smearing effect that occurs when the source signals are transmitted through the skull and the skin of the head and recorded by multiple electrodes on the scalp during EEG recordings. The discriminative sources $s^{n_1} \in \mathbb{R}^{n_1}$

represent the brain sources that exhibit changes in rhythmic activity during motor imagery, whereas the nondiscriminative sources $s^{n_2} \in \mathbb{R}^{n_2}$ represent the brain sources that maintain their rhythmic activity throughout. The nonstationary noise $\epsilon \in \mathbb{R}^n$ corresponds to artifacts that occur during the experiment, such as blinking and muscle movements.

4.2.1.2. *Dataset IIIa.* The dataset is from the BCI competition III [**15**] and includes EEG recordings from three subjects performing left hand, right hand, foot, and tongue motor imageries. EEG signals were recorded from 60 electrode channels, and each subject had either six or nine runs. Within each run, there were ten trials of motor imagery performed for each motor condition. For more detailed information about the dataset, please refer to the competition's website.[1]

We use the recordings corresponding to left hand and right hand motor imageries. Following the preprocessing steps suggested in [**75**, **95**], we apply a 5th order Butterworth filter to bandpass filter the EEG signals in the 8-30Hz frequency band. We then extract the features of the EEG signals from the time segment from 0s to 3.0s after the cue instructing the subject to perform motor imagery. In summary, for each subject, there are $N_c = 60$ or 90 trials $Y_c^{(i)} \in \mathbb{R}^{60\times 750}$ for both conditions $c \in \{-, +\}$.

4.2.1.3. *Distraction Dataset.* The distraction dataset is obtained from the BCI experiment [**19**, **20**, **21**] that aimed to simulate real-world scenarios by introducing six types of distractions on top of the primary motor imagery tasks (left hand and right hand). The six secondary tasks included no distraction (Clean), closing eyes (Eyes), listening to news (News), searching the room for a particular number (Numbers), watching a flickering video (Flicker), and dealing with vibro-tactile stimulation (Stimulation). The experiment involved 16 participants and EEG recordings were collected from 63 channels. The participants performed 7 runs of EEG recordings, with each run consisting of 36 trials of left hand motor imagery and 36 trials of right hand motor imagery. In the first run, motor imageries were performed without any distractions, and in the subsequent six runs, each distraction was added, one at a time. The dataset is publicly available.[2]

For preprocessing the EEG signals, we use the provided subject-specific frequency bands and time intervals to bandpass filter and select the time segment of the EEG signals. In summary, there

[1]http://www.bbci.de/competition/iii/desc_IIIa.pdf
[2]https://depositonce.tu-berlin.de/handle/11303/10934.2

are $N_c = 252$ trials $Y_c^{(i)} \in \mathbb{R}^{63 \times t}$ for both conditions $c \in \{-, +\}$, where the number of time samples t differs for each subject and ranges between $1,550$ to $3,532$.

4.2.1.4. *Berlin Dataset.* This dataset is from the Berlin BCI experiment described in [**16**, **96**], which involved 80 subjects performing left hand, right hand, and right foot motor imagery while EEG signals were recorded from 119 channels. The dataset consists of three runs of motor imagery calibration and three runs of motor imagery feedback, with each calibration run comprising 75 trials per condition and each feedback run comprising 150 trials per condition. Although the dataset is publicly available[3], it includes signals from only 40 of the subjects.

We focus on the recordings corresponding to the left hand and right hand. We preprocess the EEG signals by bandpass filtering them with a 2nd order Butterworth filter in the 8-30Hz frequency band and selecting 86 out of the 119 electrode channels that densely cover the motor cortex. For each subject, there are $N_c = 675$ trials $Y_c^{(i)} \in \mathbb{R}^{86 \times 301}$ for both conditions $c \in \{-, +\}$.

4.2.1.5. *Gwangju Dataset.* The EEG signals in this dataset[4] were recorded during the motor-imagery based Gwangju BCI experiment [**32**]. EEG signals were collected from 64 channels while 52 subjects performed left and right hand motor imagery. Each subject participated in either 5 or 6 runs, with each run consisting of 20 trials per motor imagery condition. Some of the trials were labeled as 'bad trials' either if their voltage magnitude exceeded a particular threshold or if they exhibited a high correlation with electromyography (EMG) activity.

Following the approach in [**32**], we discard the bad trials for each subject and exclude the subjects 's29' and 's34', who had more than 90% of their trials declared as bad trials. To preprocess the data, we apply a 4th order Butterworth bandpass filter to the signals in the 8-30Hz frequency range, and select the time interval from 0.5s to 2.5s after the cue instructing the subject to perform motor imagery. For each subject, there are N_c trials $Y_c^{(i)} \in \mathbb{R}^{64 \times 1025}$ for both conditions $c \in \{-, +\}$, where the number of trials N_c differs for each subject and ranges between 83 to 240.

4.2.1.6. *Summary.* In Table 4.1, we summarize the datasets, which include the number of channels and time samples of each trial, as well as the number of trials and participating subjects. Among the available motor imagery types, we italicize the ones we utilize.

[3]https://depositonce.tu-berlin.de/handle/11303/8979?mode=simple
[4]http://gigadb.org/dataset/view/id/100295/

TABLE 4.1. Summary of datasets.

Name	Channels(n)	Time Samples (t)	Trials (N_c)	Subjects	Motor Imagery Type
Synthetic	n	t	N_c	N/A	N/A
Dataset IIIa	60	750	60 or 90	3	*left hand, right hand,* foot, tongue
Distraction	63	range of 1550 to 3532	252	16	*left hand, right hand*
Berlin	119 (86 chosen)	301	675	80 (40 available)	*left hand, right hand,* right foot
Gwangju	64	1025	range of 83 to 240	52 (50 used)	*left hand, right hand*

4.2.2. Experiment Setting. We solve OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25) to compute the principal spatial filters x_- and x_+, respectively, of the minmax CSP. Three different approaches are considered to do so:

- OptNRQ-fp (4.3) solves OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25) directly.
- OptNRQ-nepv (Algorithm 2) solves the NEPv formulation, OptNRQ-nepv (3.65), of OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25).
- Manifold optimization algorithms in Manopt [**18**], a popular optimization toolbox on manifolds, solve the equivalent the constrained optimization (3.34) of OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25).

We initialize all algorithms with the solutions of CSP, i.e., the Rayleigh quotient optimizations (3.10) and (3.11). The stopping criteria tolerance *tol* is set to 10^{-8} for all algorithms.

We utilize the BBCI toolbox[5] to preprocess the raw EEG signals. For both conditions $c \in \{-,+\}$, we choose $m = 10$ as the number of interpolation matrices $\{V_c^{(i)}\}_{i=1}^m$ and weights $\{w_c^{(i)}\}_{i=1}^m$ in the formulations of OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25).

The experiments were conducted using MATLAB on a PC with an Apple M1 Pro processor with 16GB of RAM. To advocate for reproducible research, we share the preprocessed datasets as

[5]`https://github.com/bbci/bbci_public`

well as our MATLAB codes for the algorithms and the numerical experiments presented in this paper. [6]

4.2.3. Convergence Behavior and Timing.

In this section, we examine the convergence behavior of OptNRQ-nepv (Algorithm 2) in comparison to OptNRQ-fp (4.3) and manifold optimization algorithms. Our findings reveal that OptNRQ-fp fails to converge and exhibits oscillatory behavior. The manifold optimization algorithms do converge, but they are relatively slow when compared to OptNRQ-nepv. On the other hand, OptNRQ-nepv converges within a small number of iterations and displays a local quadratic convergence.

Example 4.2.1 (Synthetic dataset). We generate trials $Y_c^{(i)} \in \mathbb{R}^{10\times 200}$ with 50 trials for each condition. For both conditions, we choose two out of the 10 sources to be discriminative. That is, we have $s^{n_1} \in \mathbb{R}^8$ for the nondiscriminative sources and $s^{n_2} \in \mathbb{R}^2$ for the discriminative sources. The first discriminative source is sampled from $\mathcal{N}(0, 0.2)$ for condition '$-$' and $\mathcal{N}(0, 1.8)$ for condition '$+$'. The second discriminative source is sampled from $\mathcal{N}(0, 1.4)$ for condition '$-$' and $\mathcal{N}(0, 0.6)$ for condition '$+$'. For both conditions, all 8 nondiscriminative sources are sampled from $\mathcal{N}(0, 1)$ and the 10 nonstationary noise sources of ϵ are sampled from $\mathcal{N}(0, 2)$. The tolerance set radius is set as $\delta_c = 6$ for both conditions.

Four algorithms are used to solve OptNRQ$_-$ (3.21): OptNRQ-fp (4.3), OptNRQ-nepv (Algorithm 2), the Riemannian conjugate gradient algorithm in Manopt, and the Riemannian trust region algorithm in Manopt.

The convergence behaviors of the algorithms are depicted in Figure 4.1. Panel (a) displays the value of the objective function $q_-(x)$ of the minmax CSP$_-$ at each iterate x_k, while panel (b) shows the errors, measured as the difference between $q_-(x_k)$ and $q_-(x_*)$, where x_* is the solution computed by OptNRQ-nepv.

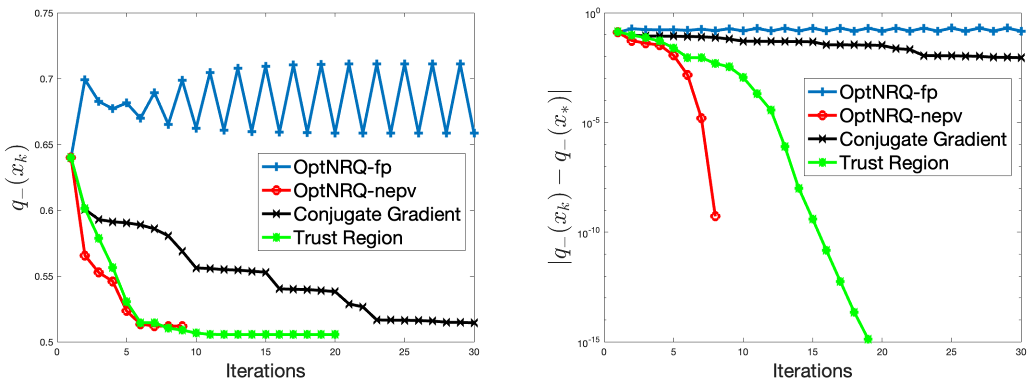
These results demonstrate that OptNRQ-fp (4.3) oscillates between two points and fails to converge. Thus, it computes a suboptimal solution. Meanwhile, the Riemannian conjugate gradient algorithm takes 497 iterations to converge. The computation time is 1.416 seconds. The Riemannian

[6]Github page: `https://github.com/gnodking7/Robust-CSP.git`

trust region algorithm takes 19 iterations. The computation time is 0.328 seconds. The OptNRQ-nepv achieves convergence in just 9 iterations with only 3 line searches. The computation time is 0.028 seconds.

	OptNRQ-fp	OptNRQ-nepv	Conjugate Gradient	Trust Region
Iteration	Does not converge	9 (3)	497	19
Time (seconds)	0.05	0.028	1.416	0.328

Although the trust region algorithm demonstrates faster convergence compared to the conjugate gradient algorithm due to its utilization of second-order gradient information, both algorithms fall short in terms of convergence speed when compared to OptNRQ-nepv. We attribute the swift convergence of OptNRQ-nepv to its utilization of the explicit equation associated with the second order derivative information. Namely, OptNRQ-nepv takes advantage of solving OptNRQ-nepv (3.65) derived from the second order optimality conditions of OptNRQ$_-$ (3.21).



(a) (b)

FIGURE 4.1. Synthetic dataset: convergence behaviors of OptNRQ-fp (4.3), OptNRQ-nepv (Algorithm 2), the Riemannian conjugate gradient algorithm in Manopt, and the Riemannian trust region algorithm in Manopt. (a) displays the objective value $q_-(x)$ of the minmax CSP$_-$ at each iteration, and (b) shows the errors between the objective values of the iterate x_k and the solution x_* obtained by OptNRQ-nepv.

EXAMPLE 4.2.2 (Berlin dataset). We continue the convergence behavior experiments on the Berlin dataset. We focus on the subject #40 and utilize the trials from the calibration runs. For both conditions, we set $\delta_c = 0.4$ as the tolerance set radius.

Four algorithms are considered for solving OptNRQ$_-$ (3.21): OptNRQ-fp (4.3), OptNRQ-nepv (Algorithm 2), the Riemannian conjugate gradient algorithm in Manopt, and the Riemannian trust region algorithm in Manopt.

The convergence behaviors of algorithms for this subject are shown in Figure 4.2. In panel (a), we display the value of the objective function $q_-(x)$ of the minmax CSP$_-$ at each iterate x_k, while in panel (b), we present the errors measured as the difference between $q_-(x_k)$ and $q_-(x_*)$, where x_* is the solution computed by OptNRQ-nepv.

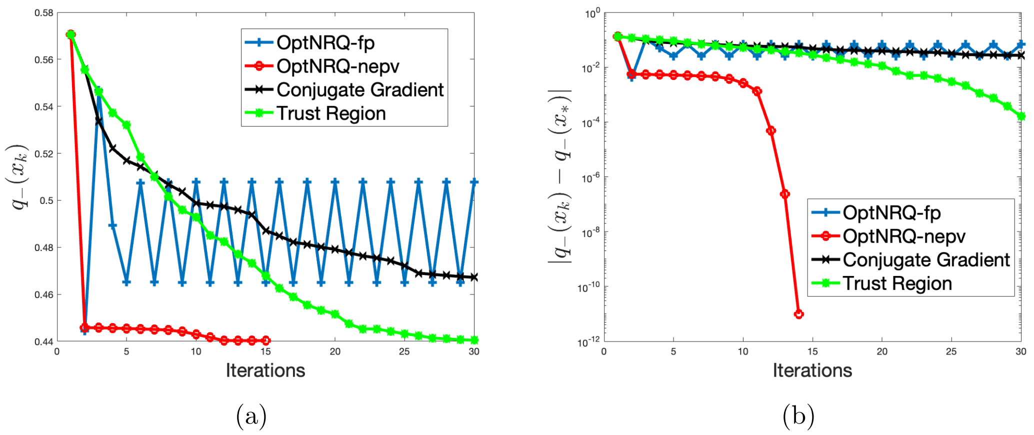


FIGURE 4.2. Berlin dataset: convergence behaviors of OptNRQ-fp (4.3), OptNRQ-nepv (Algorithm 2), the conjugate gradient algorithm using Manopt, and the trust region algorithm using Manopt. (a) displays the objective values $q_-(x)$ of the minmax CSP$_-$ at each iteration, and (b) shows the errors between the objective values of the iterate x_k and the solution x_* obtained by OptNRQ-nepv.

Similar to the results observed in Example 4.2.1, we find that OptNRQ-fp (4.3) exhibits oscillatory behavior and fails to converge. Consequently, it finds a suboptimal solution. The Riemannian conjugate gradient algorithm fails to converge within the preset 1000 iterations. The computation time is 38.2 seconds. The Riemmanian trust region algorithm converges in 73 iterations. The computation time is 73.792 seconds. Once again, OptNRQ-nepv achieves the fastest convergence. It converges in just 14 iterations and requires only 4 line searches. The computation time is only 0.389 seconds.

	OptNRQ-fp	OptNRQ-nepv	Conjugate Gradient	Trust Region
Iteration	Does not converge	14 (4)	1000+	73
Time (seconds)	0.91	0.389	38.2	73.792

4.2.4. Classification Results. We follow the common signal classification process in BCI-EEG experiments:

(i) The principal spatial filters x_- and x_+ are computed using the trials corresponding to the training dataset.
(ii) A linear classifier is trained using the principal spatial filters x_- and x_+.
(iii) The conditions of the testing trials are determined using the trained linear classifier.

4.2.4.1. *Linear classifier and classification rate.* After computing the principal spatial filters x_- and x_+, a common classification practice is to apply a linear classifier to the log-variance features of the filtered signals [**16**, **17**, **59**]. While other classifiers that do not involve the application of the logarithm are discussed in [**106**, **107**], we adhere to the prevailing convention of using a linear classifier on the log-variance features.

For a given trial Y, the log-variance features $f(Y) \in \mathbb{R}^2$ are computed as

$$f(Y) = \begin{bmatrix} \log(x_-^T Y Y^T x_-) \\ \log(x_+^T Y Y^T x_+) \end{bmatrix} \tag{4.14}$$

and a linear classifier is defined as

$$\varphi(Y) := a^T f(Y) - b \tag{4.15}$$

where the sign of the classifier $\varphi(Y)$ determines the condition of Y.

The linear weights $a \in \mathbb{R}^2$ and $b \in \mathbb{R}$ are determined from the training trials $\{Y_c^{(i)}\}_{i=1}^{N_c}$ using Fisher's linear discriminant analysis (LDA). Specifically, denoting $f_c^{(i)} := f(Y_c^{(i)})$ as the log-variance features, and

$$C_c = \sum_{i=1}^{N_c} (f_c^{(i)} - m_c)(f_c^{(i)} - m_c)^T \quad \text{with} \quad m_c = \frac{1}{N_c} \sum_{i=1}^{N_c} f_c^{(i)} \tag{4.16}$$

as the scatter matrices for $c \in \{-, +\}$, the weights are determined by

$$a = \frac{\widetilde{a}}{\|\widetilde{a}\|_2} \quad \text{with} \quad \widetilde{a} = (C_- + C_+)^{-1}(m_- - m_+), \tag{4.17}$$

and

$$b = \frac{1}{2} a^T (m_- + m_+). \tag{4.18}$$

Note that the linear classifier $\varphi(Y)$ (4.15) is positive if and only if $a^T f(Y)$ is closer to $a^T m_-$ than it is to $a^T m_+$. Therefore, the classification of a testing trial $Y_-^{(i)}$ in condition '$-$' is counted to be correct when $\varphi(Y_-^{(i)}) > 0$, and similarly, a testing trial $Y_+^{(i)}$ in condition '$+$' is counted to be correctly classified when $\varphi(Y_+^{(i)}) < 0$. The **classification rate** of a subject is defined as:

$$\text{Classification Rate } = \frac{|\{\varphi(Y_-^{(i)}) > 0\}_{i=1}^{N_-}|}{N_-} + \frac{|\{\varphi(Y_+^{(i)}) < 0\}_{i=1}^{N_+}|}{N_+},$$

where $|\{\varphi(Y_-^{(i)}) > 0\}_{i=1}^{N_-}|$ represents the number of times that $\varphi(Y_-^{(i)}) > 0$, and $|\{\varphi(Y_+^{(i)}) < 0\}_{i=1}^{N_+}|$ is defined similarly.

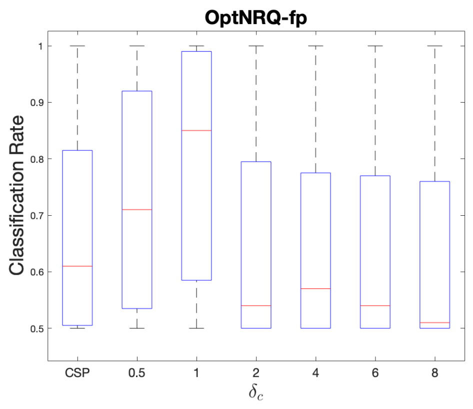
EXAMPLE 4.2.3 (Synthetic dataset). For both conditions $c \in \{-, +\}$, we generate trials $Y_c^{(i)} \in \mathbb{R}^{10\times 200}$ from the synthetic dataset. We have $s^{n_1} \in \mathbb{R}^8$ for the nondiscriminative sources and $s^{n_2} \in \mathbb{R}^2$ for the discriminative sources. The first discriminative source is sampled from $\mathcal{N}(0, 0.2)$ for condition '$-$' and $\mathcal{N}(0, 1.8)$ for condition '$+$'. The second discriminative source is sampled from $\mathcal{N}(0, 1.4)$ for condition '$-$' and $\mathcal{N}(0, 0.6)$ for condition '$+$'. For both conditions, all 8 nondiscriminative sources are sampled from $\mathcal{N}(0, 1)$. The 10 nonstationary noise sources of ϵ are sampled from $\mathcal{N}(0, 2)$ for the training trials, while they are sampled from $\mathcal{N}(0, 30)$ for the testing trials. We create 50 trials for training and 50 trials for testing for each condition.

We compute three sets of principal spatial filters x_- and x_+:

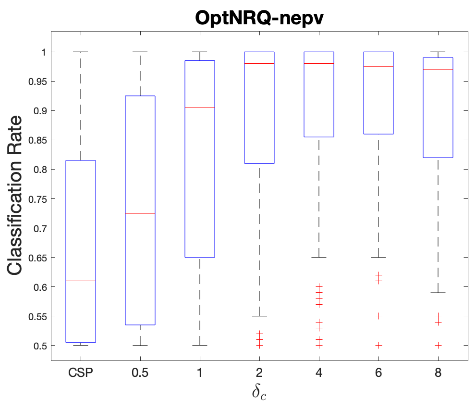
(i) One set obtained from CSP by solving the Rayleigh quotient optimizations (3.10) and (3.11),
(ii) Another set obtained from OptNRQ-fp (4.3) for OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25),
(iii) The last set obtained from OptNRQ-nepv (Algorithm 2) solving the NEPv formulation, OptNRQ-nepv (3.65), of OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25).

We consider various tolerance set radii $\delta_c \in \{0.5, 1, 2, 4, 6, 8\}$ for OptNRQ$_-$ (3.21) and OptNRQ$_+$ (3.25). When OptNRQ-fp (4.3) fails to converge, we select the solutions with the smallest objective value among its iterations. The experiment is repeated for 100 random generations of the synthetic dataset.

The resulting classification rates are summarized as boxplots in Figure 4.3. In both panels (a) and (b), the first boxplot corresponds to the classification rate of CSP. The rest of the boxplots correspond to the classification rate results by OptNRQ-fp (4.3) in panel (a), and by OptNRQ-nepv in panel (b) for different δ_c values.



(a)



(b)

FIGURE 4.3. The boxplots of the classification rate for the synthetic dataset. The first boxplot of both panel (a) and panel (b) correspond to the classification rate of CSP. The rest of the boxplots correspond to the classification rates of OptNRQ-fp (4.3) in panel (a), and by OptNRQ-nepv (Algorithm 2) in panel (b) for the tolerance set radius $\delta_c \in \{0.5, 1, 2, 4, 6, 8\}$.

For small δ_c values of 0.5 and 1, OptNRQ-fp (4.3) converges, but it fails to converge for higher δ_c values. This leads to suboptimal solutions and thus inaccurate principal spatial filters, and an inability to effectively utilize the tolerance set with an appropriate radius. Consequently, the classification rate of OptNRQ-fp (4.3) deteriorates significantly for δ_c values larger than 1, performing even worse than CSP.

In contrast, OptNRQ-nepv demonstrates a steady increase in performance as δ_c increases beyond 1. It reaches an appropriate radius δ_c that allows the tolerance set to capture the intrinsic variability of the data while disregarding the outliers. Our observations suggest that this optimal δ_c value lies between 4 and 6. At these values, OptNRQ-nepv achieves significant improvements compared to CSP.

	$\delta_c = 0.5$		$\delta_c = 1$		$\delta_c = 2$		$\delta_c = 4$		$\delta_c = 6$		$\delta_c = 8$	
For x_-	4	0	4	0	5	0	6	0	10	4	12	5
For x_+	4	0	4	0	5	0	6	0	9	1	17	7

TABLE 4.2. Number of iterations and line searches of OptNRQ-nepv (Algorithm 2) on the synthetic dataset. To each tolerance set radius δ_c, two numbers are displayed for the principal spatial filters x_- and x_+, first for the number of iterations of OptNRQ-nepv and the second for the number of line searches invoked in OptNRQ-nepv. Both numbers tend to increase as the radius δ_c increases.

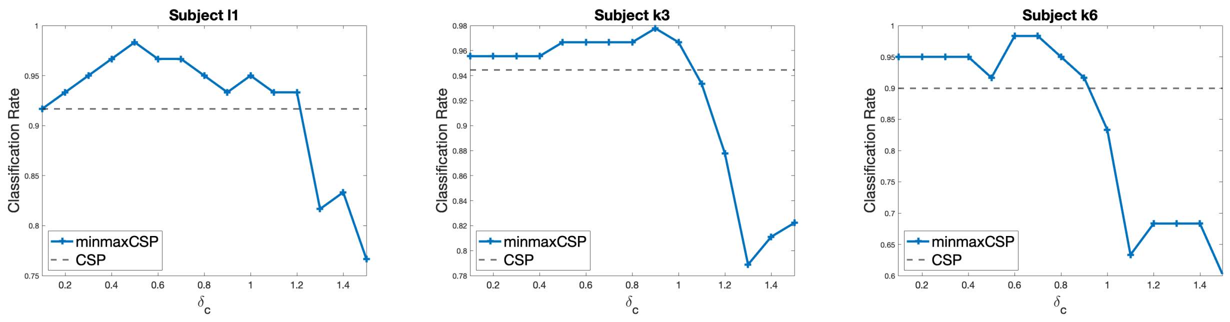
Additionally, we provide the iteration profile of OptNRQ-nepv for various values of δ_c. Table 4.2 presents the number of iterations and line searches. The first number represents the number of iterations, while the second number corresponds to the number of line searches. For small values of δ_c, the number of iterations is typically low, and no line searches are required. However, as δ_c increases, both the number of iterations and line searches also increase.

EXAMPLE 4.2.4 (Dataset IIIa). In this example, we investigate a suitable range of the tolerance set radius δ_c values for which the minmax CSP can achieve an improved classification rate compared to CSP when considering real-world BCI datasets. We use Dataset IIIa, where the trials for each subject are evenly divided between training and testing datasets, as indicated by the provided labels.

Figure 4.4 presents the classification rate results with δ_c values ranging from 0.1 to 1.5 with a step size of 0.1. The blue line represents the classification rates of the minmax CSP (solved by OptNRQ-nepv) for different δ_c values, while the gray dashed line represents the classification rate of CSP, serving as a baseline for comparison.

Our results indicate that the minmax CSP outperforms CSP for δ_c values less than 1. However, when the tolerance set radius becomes too large, the performance of the minmax CSP degrades significantly and falls below that of CSP. This decline in performance is attributed to the inclusion of outlying trials within the tolerance set, which results in a poor covariance matrix estimation. Based on the results in Figure 4.4, we restrict the tolerance set radii δ_c to values less than 1 for further classification experiments on real-world BCI datasets.

EXAMPLE 4.2.5 (Berlin dataset). For each subject, we use the calibration trials as the training dataset and the feedback trials as the testing dataset. We conduct two types of experiments:



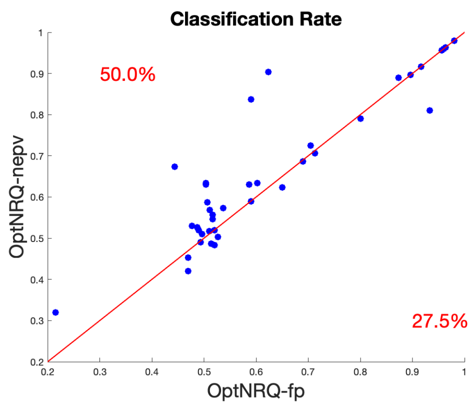
(a) (b) (c)

FIGURE 4.4. Classification rate plot for each subject. The blue line corresponds to the classification rates of the minmax CSP (solved by OptNRQ-nepv) for different values of the tolerance set radius δ_c, while the gray dashed line corresponds to the baseline classification rate obtained from CSP.

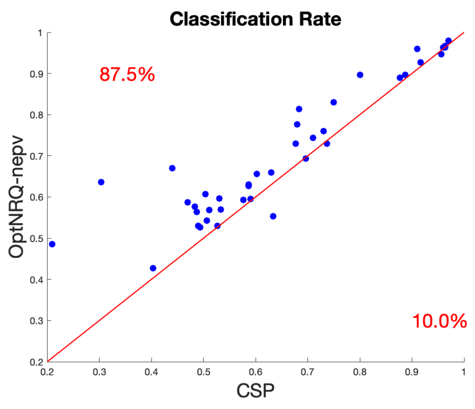
(i) comparing the classification performance between OptNRQ-fp (4.3) and OptNRQ-nepv (Algorithm 2), and (ii) comparing the classification performance between CSP and OptNRQ-nepv.

(i) We fix the tolerance set radius δ_c to be the same for all subjects in both OptNRQ-fp (4.3) and OptNRQ-nepv to ensure a fair comparison. Specifically, we choose $\delta_c = 0.4$, a value that yielded the best performance for the majority of subjects in both algorithms. When OptNRQ-fp (4.3) fails to converge, we obtain the solutions with the smallest objective value among its iterations. The classification rate results are presented in Figure 4.5(a), where each dot represents a subject. The red diagonal line represents equal performance between OptNRQ-fp (4.3) and OptNRQ-nepv, and a dot lying above the red diagonal line indicates an improved classification rate for OptNRQ-nepv. We observe that OptNRQ-nepv outperforms OptNRQ-fp (4.3) for half of the subjects. The average classification rate of the subjects improved from 62.1% with OptNRQ-fp to 65.3% with OptNRQ-nepv.

(ii) For OptNRQ-nepv, we select the optimal tolerance set radius δ_c for each subject from a range of 0.1 to 1.0 with a step size of 0.1. The red diagonal line represents equal performance between the CSP and OptNRQ-nepv, and a dot lying above the red diagonal line indicates an improved classification rate for OptNRQ-nepv. We observe that 87.5% of the dots lie above the diagonal line, indicating a significant improvement in classification rates for OptNRQ-nepv compared to the CSP. The average classification rate of the subjects improved from 64.9% with CSP to 70.5% with OptNRQ-nepv.



(a)



(b)

FIGURE 4.5. Scatter plots comparing the classification rates for Berlin dataset: (a) OptNRQ-fp (4.3) vs. OptNRQ-nepv (b) CSP vs. OptNRQ-nepv. Each dot represents a subject, and if the dot is above the red diagonal line, then OptNRQ-nepv has a higher classification rate for that subject. The percentages on the upper left and on the lower right indicate the percentage of subjects that performs better for the corresponding algorithm.

EXAMPLE 4.2.6 (Distraction dataset). In this example, we illustrate the robustness of OptNRQ-nepv to noises by utilizing the Distraction dataset. In particular, we measure the robustness by the classification rate. For each subject, the trials without any distractions are set as the training dataset and the trials from the six distraction tasks are set as the testing dataset.

We compute two sets of principal spatial filters from the training dataset: (i) one set obtained from CSP and (ii) another set obtained from OptNRQ-nepv. For each testing dataset of distraction task, the tolerance set radius δ_c is optimally selected from a range of 0.1 to 1.0 with a step size of 0.1.

The scatter plot in Figure 4.6 compares the classification rates of CSP and OptNRQ-nepv for each subject. The red diagonal line represents the equal performance of the two algorithms, and each dot represents a subject. A dot located above the diagonal line indicates an improvement in the classification rate for OptNRQ-nepv compared to CSP.

We observe clear improvements in classification rates for OptNRQ-nepv compared to CSP, particularly for the subjects with low BCI control, across all six distraction tasks. The average classification rate of the subjects increased from 62.35% with CSP to 66.68% with OptNRQ-nepv.

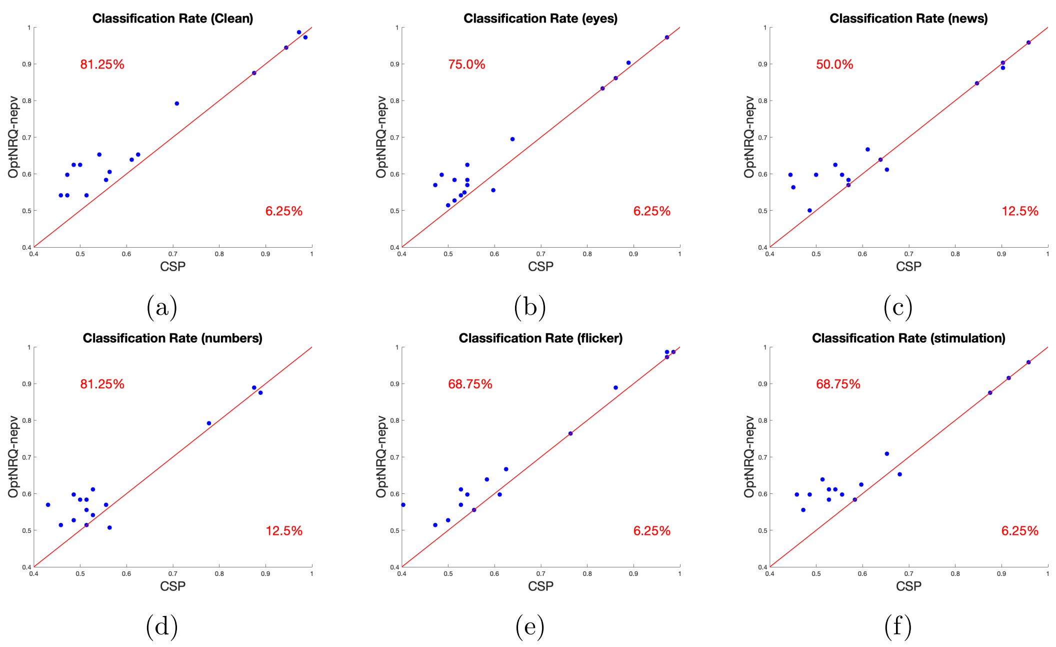


FIGURE 4.6. Classification rate scatter plots for 6 distraction tasks. For each subject, optimal δ_c are found for each distraction task. Each dot represents a subject - a dot above the red diagonal line indicates the performance improvement of OptNRQ-nepv (Algorithm 2) over CSP. The percentages in the upper left and lower right corners indicate the proportion of subjects for whom OptNRQ-nepv outperforms the CSP and vice versa, respectively.

These results demonstrate the increased robustness of OptNRQ-nepv to CSP in the presence of noise in EEG signals.

EXAMPLE 4.2.7 (Gwangju dataset). To perform a classification rate experiment on the Gwangju dataset, we follow the cross-validation steps outlined in [**32**]. Firstly, the trials for each motor condition are randomly divided into 10 subsets. Next, seven of these subsets are chosen as the training set, and the remaining three subsets are used for testing. For each subject, this procedure is conducted for all 120 possible combinations of training and testing datasets creation.

For each combination, we compute the principal spatial filters of CSP and OptNRQ-nepv (Algorithm 2). For OptNRQ-nepv, we find the optimal tolerance set radius δ_c for each subject from a range of 0.1 to 1.0 with a step size of 0.1.

The average classification rates of the 120 combinations for each of the 50 subjects are computed. These results are shown in Figure 4.7, where the red diagonal line represents equal performance between the CSP and OptNRQ-nepv, and each dot represents the average classification rates of a subject. A dot lying above the red diagonal line indicates an improved classification rate for OptNRQ-nepv compared to CSP.

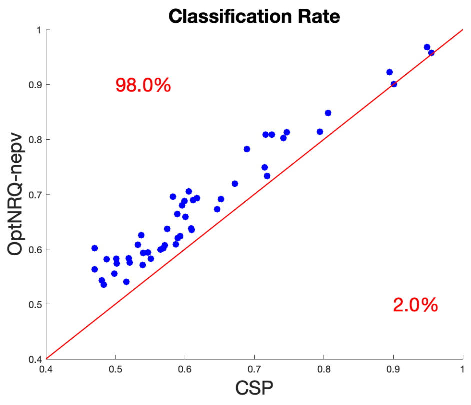


FIGURE 4.7. Scatter plots comparing the classification rates of CSP and OptNRQ-nepv (Algorithm 2) for the Gwangju dataset. Each dot represents a subject, and if the dot is above the red diagonal line, then OptNRQ-nepv has a higher classification rate than CSP. The percentages on the upper left and on the lower right indicate the percentage of subjects that OptNRQ-nepv performs better than CSP and vice versa, respectively.

With the exception of one subject whose classification rates for the CSP and OptNRQ-nepv are equal, all other subjects benefit from OptNRQ-nepv. The average classification rate of the subjects improved from 61.2% for CSP to 67.0% for OptNRQ-nepv.

CHAPTER 5

Wasserstein Discriminant Analysis: NTRopt

5.1. Introduction

As widely used feature extraction approaches in machine learning, dimensionality reduction (DR) methods [**23**, **33**, **46**, **112**] learn projections such that the projected lower dimensional subspaces maintain the coherent structure of datasets and reduce computational costs of classification or clustering. The linear projection obtained from linear DR methods takes the form of a matrix such that the embedding to the lower dimensional subspace only involves matrix multiplications. Due to such ease in interpretation and implementation, linear DR methods are often the favored choice among numerous DR methods. For example, principal component analysis (PCA) [**48**] seeks to find a linear projection that preserves the dataset's variation and is one of the most common and well-known DR methods. Other well-known DR methods include Fisher linear discriminant analysis (LDA) [**48**] to take into account the information of classes and compute a linear projection that best separates different classes, and Mahalanobis metric learning [**67**] to seek a distance metric that better models the relationship among dataset from a linear projection.

Wasserstein discriminant analysis (WDA) [**45**] is a supervised linear DR that is based on the use of regularized Wasserstein distances [**34**] as a distance metric. Similar to Fisher linear discriminant analysis (LDA), WDA seeks a projection matrix to maximize the dispersion of projected points between different classes and minimize the dispersion of projected points within same classes. An important distinction between LDA and WDA is that while LDA only considers the global relations between data points [**24**, **44**, **81**, **117**], WDA can dynamically take into account both global and local information through the choice of a regularization parameter. A recent study has shown that WDA outperforms other linear DR methods in various learning tasks, such as sequence pattern analysis [**103**, **104**], graph classification [**127**], and multi-view classification [**57**].

WDA is formulated as a bi-level nonlinear trace ratio optimization [**45**]. The inner optimization computes the optimal transport (OT) matrices of the regularized Wasserstein distance, an essential factor responsible for WDA's ability to account for both global and local relations. The outer optimization involves a nonlinear trace ratio for the dispersion of data classes.

5.2. Wasserstein Distances

We first present the background on Wasserstein distance [**34**, **113**], also known as Earth mover's distance [**91**], and its regularized variant known as regularized Wasserstein distance [**45**]. These distances define a geometry over the space of probability distributions.

5.2.1. Wasserstein Distance.

Given two probability vectors $r \in \mathbb{R}_+^n$ and $c \in \mathbb{R}_+^m$, i.e., $r^T 1_n = 1$ and $c^T 1_m = 1$, the following set of positive matrices

$$U(r,c) := \left\{ T \mid T \in \mathbb{R}_+^{n\times m},\, T1_m = r,\, T^T 1_n = c \right\} \tag{5.1}$$

is called the transport polytope of r and c. In analogy, if we consider each component of r as a pile of sand and each component of c as a hole to be filled in, the component T_{ij} of T represents the amount of sand from the pile r_i that gets moved to the hole c_j. For this reason, a matrix $T \in U(r,c)$ is referred to as a transport matrix of r and c.

Let $M \in \mathbb{R}^{n\times m}$ be a non-negative matrix whose elements $M_{ij} \geqslant 0$ represents the unit cost of transporting r_i to c_j. Then, the cost of transporting r to c using a transport matrix T and a cost matrix M is quantified as

$$\langle T, M\rangle = \sum_{ij} T_{ij} M_{ij}. \tag{5.2}$$

Given the matrix M, the optimization problem, referred to as the optimal transport problem [**34**, **113**],

$$W(r,c) := \min_{T \in U(r,c)} \langle T, M\rangle \tag{5.3}$$

seeks a transport matrix T that minimizes the transport cost between r and c. When M is a metric matrix, i.e., M belongs in the cone

$$\mathcal{M} = \Big\{ M \mid M \in \mathbb{R}_+^{n\times m},\ M_{ij} = 0 \Leftrightarrow i = j,\ \text{and } M_{ij} \leqslant M_{ik} + M_{kj},\ \forall i,j,k \Big\},$$

then $W(r,c)$ (5.3) forms a distance between r and c [**113**], and is called the **optimal transport distance** or the **Wasserstein distance**[1] between r and c.

5.2.2. Regularized Wasserstein Distance. Computing the Wasserstein distance, however, is subject to heavy costs that scale in super-cubic with respect to the size of the probability vector [**86**]. To reduce such heavy costs, Cuturi [**34**] proposed an entropic constraint on the transport matrix that not only lowers the computation cost of the problem but also smooths the search space. Namely, a convex subset $U_\alpha(r,c)$ of $U(r,c)$ was introduced:

$$U_\alpha(r,c) := \Big\{ T \mid T \in U(r,c) \text{ and } h(T) \geqslant h(r) + h(c) - \alpha \Big\}, \tag{5.4}$$

where $\alpha \geqslant 0$ and $h(r)$ and $h(T)$ are the entropy of a probability vector and the entropy of a transport matrix, respectively, defined as

$$h(r) = -\sum_{i=1}^{n} r_i \log r_i \quad \text{and} \quad h(T) = -\sum_{ij} T_{ij} \log T_{ij}. \tag{5.5}$$

Then, Cuturi [**34**] defined the following distance between r and c

$$W_\alpha(r,c) := \min_{T \in U_\alpha(r,c)} \langle T, M \rangle \tag{5.6}$$

as the Sinkhorn distance, such naming due to its solvability by the Sinkhorn-Knopp algorithm.

By considering the Lagrange multiplier for the entropy constraint of the Sinkhorn distance, the optimal solution to (5.6), denoted as T^λ, can be computed as the solution to the problem

$$T^\lambda = \operatorname*{argmin}_{T \in U(r,c)} \Big\{ \langle T, M \rangle - \frac{1}{\lambda} h(T) \Big\}. \tag{5.7}$$

[1]Strictly speaking, the correct terminology of $W(r,c)$ (5.3) is the 1st Wasserstein distance with discrete measure [**87**, **113**]. To be consistent with the terminology in Wasserstein Discriminant Analysis [**45**], we will refer to it as the Wasserstein distance and its regularized counterpart as the regularized Wasserstein distance.

Consistent with the terminology in [**45**], we refer to T^λ as the **optimal transport (OT) matrix** with λ as the **regularization parameter**, and

$$W_\lambda(r,c) := \langle T^\lambda, M\rangle \tag{5.8}$$

as the **regularized Wasserstein distance** between the probability vectors r and c. By duality theory, to each α in Sinkhorn distance (5.6) corresponds a $\lambda \in [0,\infty]$ in regularized Wasserstein distance (5.8) such that

$$W_\alpha(r,c) = W_\lambda(r,c).$$

Moreover, as the regularized parameter $\lambda \to \infty$, the regularized Wasserstein distance $W_\lambda(r,c)$ approaches the Wasserstein distance $W(r,c)$ [**34**].

5.2.3. Structure of the Optimal Transport Matrix. The entropy smoothed optimal transport problem (5.7) looks to minimize the total transport cost while maximizing the entropy of the transport matrix. Both of these terms are convex which makes the problem (5.7) a convex problem. Thus, the OT matrix T^λ exists and is unique [**34**]. Furthermore, the OT matrix T^λ admits a simple structure based on the first order analysis and Sinkhorn's Theorem [**98**].

THEOREM 5.2.1. *For $\lambda > 0$, the solution T^λ to the entropy smoothed optimal transport problem* (5.7) *is unique and has the form*

$$T^\lambda = D(u)KD(v), \tag{5.9}$$

where $u \in \mathbb{R}^n_+$, $v \in \mathbb{R}^m_+$, and $K := e^{-\lambda M}$ is the element-wise exponential of $-\lambda M$. u and v are uniquely defined up to a multiplication factor[2] *by the relation*

$$\begin{cases} u = r./(Kv) \\ v = c./(K^T u). \end{cases} \tag{5.10}$$

PROOF. See [**34**, Lemma 2]. □

Theorem 5.2.1 indicates that computing the OT matrix T^λ is equivalent to the problem of computing the vectors $u \in \mathbb{R}^n_+$ and $v \in \mathbb{R}^m_+$ from the given matrix $K \in \mathbb{R}^{n\times m}_+$ to satisfy (5.10). Such

[2]Given solutions u and v to (5.9), for any positive nonzero scalar α, αu and $\frac{1}{\alpha}v$ are also solutions to (5.9).

problem belongs to the class of well-known problems called the matrix balancing problems. We leave the discussion of the matrix balancing problem and its algorithms to Section 6.2.2.

5.2.4. Costs of Wasserstein Distance and Regularized Wasserstein Distance. Computing the Wasserstein distance (5.3) comes with a high computational cost. Assuming $d = n = m$ for simplicity, for any convex optimization algorithms, such as network simplex method or interior point method, the cost of computing the Wasserstein distance (5.3) scales in super-cubic $O(d^3 \log(d))$ [**86**]. On the other hand, as a matrix balancing problem, the regularized Wasserstein distance (5.8) can be solved more efficiently with existing algorithms. For instance, it can be computed with the Sinkhorn-Knopp (SK) iteration [**97, 98, 99**] with a linear convergence rate [**47, 62**]. The SK algorithm only involves matrix-vector multiplications and scales as $O(d^2)$. We will discuss the SK iteration and its accelerated variants in Section 6.2.2.

Remark 5.2.1. *As pointed out in [**45**], the entropy smoothed optimal transport problem* (5.7) *of the OT matrix T^λ can equivalently be written as*

$$T^\lambda = \operatorname*{argmin}_{T \in U(r,c)} \Big\{ \lambda \langle T, M \rangle - h(T) \Big\}. \tag{5.11}$$

We adopt the formulation (5.11) *for the rest of the WDA discussion.*

5.3. Wasserstein Discriminant Analysis

Wasserstein discriminant analysis (WDA) seeks to find a projection matrix that maximizes the dispersion between different classes and minimizes the dispersion within same classes. Meanwhile, WDA can also control global and local inter-relations between data classes using a regularized Wasserstein distance. In this section, we present the formulation of WDA as a nonlinear trace ratio optimization. Then, we conclude the section with discussions on the advantages of WDA.

5.3.1. Data Matrices and Their Distances. Consider the datasets $\{x_i^c\}_{i=1}^{N_c}$ of data points $x_i^c \in \mathbb{R}^d$ for different classes $c = 1, 2, \ldots, C$, where N_c denotes the number of data points in class c. For each class c, a data matrix $X^c \in \mathbb{R}^{d \times N_c}$ is formed by constructing its columns to be the vectors x_i^c in class c:

$$X^c = [\, x_1^c, \, \ldots, \, x_{N_c}^c \,].$$

Without loss of generality, we assume that the data matrices X^c are standardized, i.e., each feature is scaled such that it has mean 0 and standard deviation 1. Otherwise, we can always preprocess X^c as

$$X^c \leftarrow X^c - \frac{1}{N_c}(X^c 1_{N_c}) 1_{N_c}^T$$

to have mean 0 then divide each feature by its standard deviation to set its standard deviation to be 1. Data standardization is a common practice in machine learning algorithms to transform the features to a common scale while maintaining the structure of the data. This often leads to improved performance in algorithms [**52**].

5.3.2. Definition of Wasserstein Discriminant Analysis. WDA introduced in [**45**] presumes empirical measure as the underlying probability measure of the data matrices and uses the regularized Wasserstein distance as the distance metric. Specifically, the regularized Wasserstein distance between two projected data matrices by an orthonormal projection $P \in \mathbb{R}^{d \times p}$ ($p \ll d$) is defined as

$$W_\lambda(P^T X^c, P^T X^{c'}) := \langle T^{c,c'}(P), M_{P^T X^c, P^T X^{c'}} \rangle, \tag{5.12}$$

where $M_{P^T X^c, P^T X^{c'}}$ is the cost matrix defined as the Euclidean distances between projected points:

$$M_{P^T X^c, P^T X^{c'}} := \Big([\| P^T x_i^c - P^T x_j^{c'} \|_2^2]_{ij} \Big) \in \mathbb{R}^{N_c \times N_{c'}} \tag{5.13}$$

and $T^{c,c'}(P)$ is the OT matrix, defined as the solution of the entropy-smoothed OT problem:

$$T^{c,c'}(P) := \underset{T \in U_{N_c, N_{c'}}}{\operatorname{argmin}} \Big\{ \lambda \langle T, M_{P^T X^c, P^T X^{c'}} \rangle - h(T) \Big\}, \tag{5.14}$$

where $U_{N_c, N_{c'}}$ is the transport polytope between $P^T X^c$ and $P^T X^{c'}$ defined as

$$U_{N_c, N_{c'}} := \Big\{ T \mid T \in \mathbb{R}_+^{N_c \times N_{c'}},\, T 1_{N_{c'}} = \frac{1}{N_c} 1_{N_c},\, T^T 1_{N_c} = \frac{1}{N_{c'}} 1_{N_{c'}} \Big\}. \tag{5.15}$$

The regularization parameter $\lambda \geqslant 0$ plays a critical role in dynamically controlling the global and local relations between data points. We highlight the role of λ in the discussion of the advantages of WDA.

WDA adopts the formulation of Linear discriminant analysis (LDA) to seek a projection matrix $P \in \mathbb{R}^{d\times p}$ by solving

$$\max_{P^TP=I_p} \frac{\sum_{c,c'>c} W_\lambda(P^TX^c, P^TX^{c'})}{\sum_c W_\lambda(P^TX^c, P^TX^c)}. \tag{5.16}$$

The numerator and the denominator of (5.16) are the sums of the regularized Wasserstein distances (5.12) between the inter-classes and the intra-classes, respectively. As a maximization problem, WDA (5.16) seeks a projection that maximizes the numerator (the dispersion between different classes) and minimizes the denominator (the dispersion within same classes).

5.3.3. Nonlinear Trace Ratio Formulation of WDA. By (5.13) and the Frobenius inner product, the inter-class distance between data matrices P^TX^c and $P^TX^{c'}$ is

$$\begin{aligned}
W_\lambda(P^TX^c, P^TX^{c'}) &= \langle T^{c,c'}(P), M_{P^TX^c,P^T,X^{c'}}\rangle \\
&= \sum_{ij} T^{c,c'}_{ij}(P)\|P^Tx^c_i - P^Tx^{c'}_j\|_2^2 \qquad (5.17)\\
&= \sum_{ij} T^{c,c'}_{ij}(P)\mathrm{Tr}(P^T(x^c_i - x^{c'}_j)(x^c_i - x^{c'}_j)^TP) \\
&= \mathrm{Tr}(P^T[\sum_{ij} T^{c,c'}_{ij}(P)(x^c_i - x^{c'}_j)(x^c_i - x^{c'}_j)^T]P) \\
&=: \mathrm{Tr}(P^TC^{c,c'}(P)P), \qquad (5.18)
\end{aligned}$$

where

$$C^{c,c'}(P) := \sum_{ij} T^{c,c'}_{ij}(P)(x^c_i - x^{c'}_j)(x^c_i - x^{c'}_j)^T. \tag{5.19}$$

Similarly, the intra-class distance between P^TX^c to itself is a trace operator

$$W_\lambda(P^TX^c, P^TX^c) =: \mathrm{Tr}(P^TC^{c,c}(P)P), \tag{5.20}$$

where

$$C^{c,c}(P) := \sum_{ij} T^{c,c}_{ij}(P)(x^c_i - x^c_j)(x^c_i - x^c_j)^T. \tag{5.21}$$

Thus, by (5.18) and (5.20), WDA (5.16) can be reformulated as the nonlinear trace ratio optimization (NTRopt), which we refer to as **NTRopt-WDA**:

$$\max_{P^TP=I_p} \left\{ q(P) := \frac{\mathrm{Tr}(P^TC_b(P)P)}{\mathrm{Tr}(P^TC_w(P)P)} \right\}, \tag{5.22}$$

where $C_b(P)$ and $C_w(P)$ are defined as the between and within cross-covariance matrices, respectively:

$$C_b(P) := \sum_{c,c'>c} C^{c,c'}(P) = \sum_{c,c'>c}\sum_{ij} T_{ij}^{c,c'}(P)(x_i^c - x_j^{c'})(x_i^c - x_j^{c'})^T \in \mathbb{R}^{d\times d}, \tag{5.23}$$

$$C_w(P) := \sum_{c} C^{c,c}(P) = \sum_{c}\sum_{ij} T_{ij}^{c,c}(P)(x_i^c - x_j^{c})(x_i^c - x_j^{c})^T \in \mathbb{R}^{d\times d}. \tag{5.24}$$

We emphasize that NTRopt-WDA (5.22) is a bi-level optimization. In order to obtain the cross-covariance matrices $C_b(P)$ and $C_w(P)$, the OT matrices $\{T^{c,c'}\}_{c,c'>c}$ and $\{T^{c,c}\}_c$ need to be computed from the minimization problems (5.14).

5.3.4. Advantages of WDA. Among the number of advantages of WDA, two outstanding ones are that WDA is a generalization of LDA and that WDA can dynamically consider both global and local relations of the data points.

- When the regularization parameter $\lambda = 0$, each relation between data points is treated equally and NTRopt-WDA (5.22) reduces to LDA. Specifically, by (5.11), the OT matrix is

  $$T^{c,c'} := \underset{T\in U_{N_c,N_{c'}}}{\mathrm{argmax}}\ h(T).$$

  The entropy maximizing matrix is simply

  $$T^{c,c'} = \frac{1}{N_cN_{c'}} 1_{N_c,N_{c'}},$$

  and the cross-covariance matrices become

  $$C^{c,c'} = \frac{1}{N_cN_{c'}} \sum_{ij}(x_i^c - x_j^{c'})(x_i^c - x_j^{c'})^T.$$

Consequently, NTRopt-WDA reduces to LDA:

$$\max_{P^TP=I_p} \frac{\mathrm{Tr}(P^TS_bP)}{\mathrm{Tr}(P^TS_wP)}$$

where

$$S_b := \sum_{c,c'>c} \frac{1}{N_cN_{c'}} \sum_{ij}(x_i^c - x_j^{c'})(x_i^c - x_j^{c'})^T \quad \text{and} \quad S_w := \sum_c \frac{1}{N_c^2} \sum_{ij}(x_i^c - x_j^c)(x_i^c - x_j^c)^T. \tag{5.25}$$

The between and within cross-covariance matrices S_b and S_w, respectively, indicate that all relations between data points carry equal weight. Therefore, LDA strictly enforces global relations among data points.

- Unlike other linear DR methods that are strictly based on either global relations (such as LDA) or local relations (such as Large Margin Nearest Neighbor [**117**]) of the data points, WDA can dynamically consider both global and local relations by varying the regularization parameter λ in the regularized Wasserstein distances.

In the regularized Wasserstein distance (5.17), the transportation weight $T_{ij}^{c,c'}(P)$ quantifies the importance of the quantity $\|P^Tx_i^c - P^Tx_j^{c'}\|_2^2$, i.e., the Euclidean distance between the projected data point x_i^c in class c and the projected data point $x_j^{c'}$ in class c'. By varying the strength of λ, the regularized Wasserstein distance is able to control the value of the transportation weights and thus the relations between data points. Small λ puts more emphasis on the global coherency and large λ puts more emphasis on the local structure of the class manifold.

Figure 5.1 illustrates the impact that λ has on the transportation weights. Shown in the figure is a synthetic dataset consisting of a class that displays a half-moon shape and another class that is a bi-modal Gaussian distribution with a mode on each side of the half-moon shape. The transportation weights are represented as the edges between the data points such that their magnitudes are reflected by the visibility of the edges. The left column shows the inter-class relations and the right column shows the intra-class relations. Each row corresponds to a different λ value, with the first row corresponding to $\lambda = 0.1$, the second row to $\lambda = 0.5$, and the third row to $\lambda = 1$. For both inter-class and intra-class relations, we observe that as λ increases from $\lambda = 0.1$ to $\lambda = 1$ the structure of the class manifold shifts from globality to locality. For instance, when $\lambda = 0.1$, if we focus on one of the red data points on the top mode we observe that it has strong inter-class relations with almost all the blue points and strong intra-class relations with almost all the red points. However, when $\lambda = 1$, the same red data point has strong inter-class relations only with the blue points that are close-by and strong intra-class relations only with the red points that are in the top modal.

The impact of λ on the global and local relations between the projected data points can be explained by examining the formulation of the transportation weights and the sum constraint imposed on them. Suppose that we have three data points $x_i^c, x_j^{c'}, x_k^{c'}$ such that $P^T x_i^c$ is closer to $P^T x_j^{c'}$ than it is to $P^T x_k^{c'}$, i.e,

$$\|P^T x_i^c - P^T x_j^{c'}\|_2^2 < \|P^T x_i^c - P^T x_k^{c'}\|_2^2. \tag{5.26}$$

According to Theorem 5.2.1, the transportation weights $T_{ij}^{c,c'}(P)$ and $T_{ik}^{c,c'}(P)$ between $P^T x_i^c$ and $P^T x_j^{c'}$ and between $P^T x_i^c$ and $P^T x_k^{c'}$, respectively, are equal to

$$T_{ij}^{c,c'}(P) = u_i^{c,c'} e^{-\lambda\|P^T x_i^c - P^T x_j^{c'}\|_2^2} v_j^{c,c'} \quad \text{and} \quad T_{ik}^{c,c'}(P) = u_i^{c,c'} e^{-\lambda\|P^T x_i^c - P^T x_k^{c'}\|_2^2} v_k^{c,c'}, \tag{5.27}$$

where $u_i^{c,c'}, v_j^{c,c'}, v_k^{c,c'}$ are found by an algorithm for matrix balancing problems such as the Sinkhorn-Knopp (SK) algorithm or the accelerated SK (Acc-SK) algorithm. According to (5.27), the condition (5.26) suggests that $T_{ij}^{c,c'}(P) > T_{ik}^{c,c'}(P)$, with the value of λ determining the magnitude difference between $T_{ij}^{c,c'}(P)$ and $T_{ik}^{c,c'}(P)$. Specifically, as λ increases, the disparity in size between $T_{ij}^{c,c'}(P)$ and $T_{ik}^{c,c'}(P)$ also increases. This indicates that the relationship between closely-distanced data points is emphasized more than the relationship between widely-distanced data points. Furthermore, the sum constraint on the OT matrix $T^{c,c'}(P)$, expressed as

$$T^{c,c'}(P) 1_{N_{c'}} = \frac{1}{N_c} 1_{N_c} \quad \text{and} \quad T^{c,c'}(P)^T 1_{N_c} = \frac{1}{N_{c'}} 1_{N_{c'}}, \tag{5.28}$$

implies that satisfying the constraint (5.28) requires an inverse relationship between transportation weights. When one transportation weight becomes small, the other transportation weights must become large. In other words, for large λ, the weaker data relations between distantly located data points contribute to the stronger data relations between closely located data points. This leads to a more pronounced locality in the relationships within the class manifold.

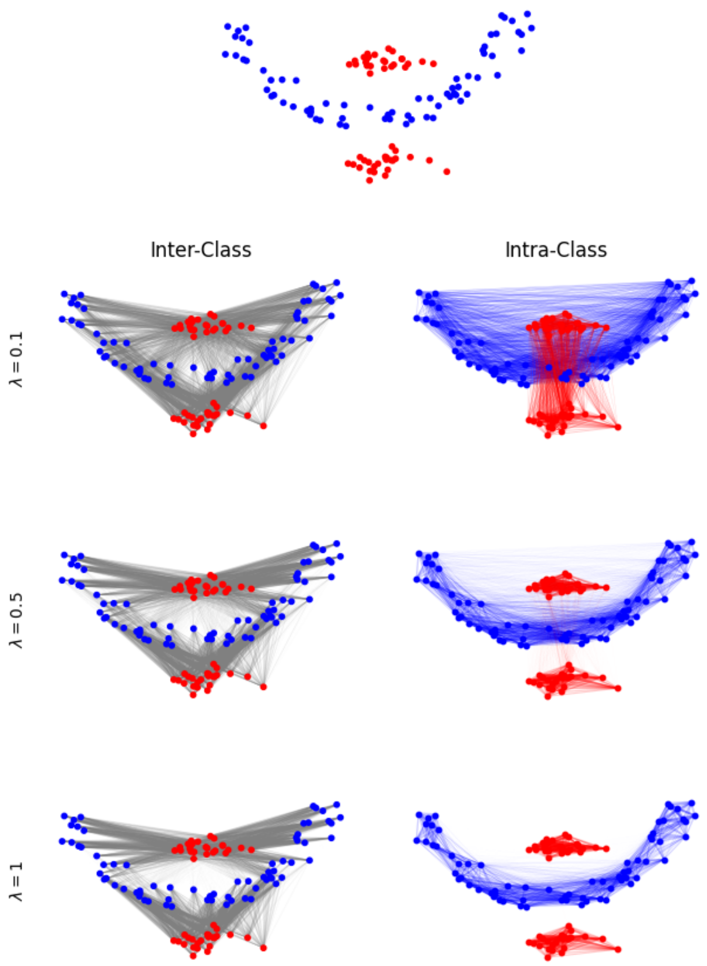


FIGURE 5.1. Illustration of globality and locality relations of two class datasets as the regularization parameter takes on the values $\lambda = 0.1, 0.5, 1$. The first plot displays the shape of two classes, and the following left column and the following right column illustrate inter-class relations and intra-class relations, respectively, with each row corresponding to different λ.

CHAPTER 6

Wasserstein Discriminant Analysis: Algorithms and Numerical Results

6.1. Existing Algorithms

To the best of our knowledge, there are two algorithms that are specifically designed for solving NTRopt-WDA (5.22). In this section, we provide a summary of these two algorithms and point out their shortcomings.

6.1.1. WDA-gd. In [**45**], NTRopt-WDA (5.22) is solved using the projected gradient descent method. The derivative of the OT matrix is computed as the derivative of the matrix $T^L(P) = D(u^L)KD(v^L)$ obtained from a preset L^{th} iteration of SK algorithm (see Section 6.2.2 for the discussion on SK algorithm). A recursion occurs in computing the derivative of $T^L(P)$ as u^L depends on v^L which in turn depends on u^{L-1}. An automatic differentiation is used to compute the derivative of $T^L(P)$ in order to utilize the recursion scheme of SK algorithm. This derivative-based algorithm is referred to as WDA-gd. However, WDA-gd is subject to some practical shortcomings.

- A large number of SK iterations may be needed to obtain the OT matrices and their derivatives (see Example 6.2.2). If the iteration number L is not large enough, both the OT matrices and their derivatives may be inaccurate.
- Computing the derivative of the objective function of NTRopt-WDA (5.22) comes with a heavy cost that scales quadratically in the dimension size, the number of data points, and the number of data classes. Assuming that each class has N number of data points, at each iteration of WDA-gd, the costs of computing the derivatives of the OT matrices from automatic differentiation are $O(LpN^2d^2)$, where L is the number of SK iterations, p is the size of subspace dimension, and d is the size of the original dimension. The other computation costs in the derivative of the objective function are $O(pC^2N^2d)$ where C is

the number of classes. Therefore, the overall cost for computing the derivative at each iteration is $O(LpC^2N^2d^2)$.

6.1.2. WDA-eig. In [**72**], the authors proposed to approximate NTRopt-WDA (5.22) with the following ratio trace surrogate model

$$\max_{P^TP=I_p} \operatorname{Tr}\Big((P^TC_w(P)P)^{-1}(P^TC_b(P)P) \Big), \tag{6.1}$$

then further approximate this surrogate model using the NEPv:

$$C_b(P)P = C_w(P)P\Lambda, \tag{6.2}$$

where P is the eigenvector corresponding to the p largest eigenvalues of $(C_b(P), C_w(P))$. The NEPv (6.2) was solved using the self-consistent field (SCF) iteration:

$$P_{k+1} \leftarrow \text{eigenvectors of the } p \text{ largest eigenvalues of } (C_b(P_k), C_w(P_k)). \tag{6.3}$$

The approach was named WDA-eig. There are major issues with this approach.

- The OT matrices needed in $C_b(P_k)$ and $C_w(P_k)$ are computed using SK algorithm. As a result, WDA-eig suffers from the same shortcomings as WDA-gd of needing a large number of SK iterations.
- As discussed in Section 2.2.1.1, (RTopt) can be a poor approximation to (TRopt), often leading to suboptimal solutions and inferior classification outcomes compared to (TRopt). Similarly, in the case of their nonlinear extensions, the surrogate trace ratio model (6.1) may also serve as a poor approximation to NTRopt-WDA (5.22). In Figure 6.6, we present an illustration demonstrating that the surrogate model (6.1) leads to suboptimal solutions.
- There is no proper quantification for the quality of the surrogate model (6.1) to the NEPv (6.2). It is not known if the solution P of the model (6.1) corresponds to the p largest eigenvectors of the NEPv (6.2). Therefore, the converged projection from the SCF iteration (6.3) may not be the solution to the surrogate model (6.1).

6.2. A Bi-Level Nonlinear Eigenvector Algorithm

In this section, we present a new algorithm, called WDA-nepv, to address the shortcomings of WDA-gd and WDA-eig. Unlike WDA-gd, WDA-nepv eliminates the need for computing the derivatives of the objective function $q(P)$ in NTRopt-WDA (5.22). Meanwhile, unlike WDA-eig, WDA-nepv directly tackles the objective function $q(P)$ without using a surrogate function. Therefore, WDA-nepv is *derivative-free* and *surrogate-model-free.* WDA-nepv fully leverages the bi-level structure of NTRopt-WDA (5.22) by formulating both the inner and outer optimizations as NEPvs. This approach presents a unified framework for addressing NTRopt-WDA (5.22).

6.2.1. Algorithm Outline. WDA-nepv follows a bi-level or inner-outer iteration scheme in which at each iteration the projection dependence on the cross-covariance matrices is fixed and the projection is updated as the solution of the resulting trace ratio optimization (TRopt). There lie two NEPvs in WDA-nepv: one for computing the OT matrices of the cross-covariance matrices and the other for computing (TRopt). Here is an outline of WDA-nepv:

(1) Start with an initial projection P_0 with $P_0^T P_0 = I_p$.

(2) At the kth iteration:

  (a) Compute the OT matrices $T^{c,c'}(P_k)$ and $T^{c,c}(P_k)$ by an NEPv (Sec. 6.2.2).

  (b) Use the OT matrices $T^{c,c'}(P_k)$ and $T^{c,c}(P_k)$ to form between and within cross-covariances $C_b(P_k)$ and $C_w(P_k)$ by level-3 BLAS (Basic Linear Algebra Subprograms) (see Sec. 6.2.3).

  (c) Compute P_{k+1}:

$$P_{k+1} = \operatorname*{argmax}_{P^T P = I_p} \frac{\mathrm{Tr}(P^T C_b(P_k) P)}{\mathrm{Tr}(P^T C_w(P_k) P)} \tag{6.4}$$

  by another NEPv for (TRopt) (see Sec. 6.2.4);

(3) Terminate when P_k and P_{k+1} are sufficiently close (see remarks of Algorithm 4).

We solve both NEPvs using the self-consistent field (SCF) iteration.

6.2.2. Computing OT Matrices by NEPv. In NTRopt-WDA (5.22), a key computation step involves computing the OT matrices defined in (5.14). The OT matrices take an integral part in quantifying the global and local relations of data points. Therefore, accurate and efficient computations of OT matrices are crucial. According to Theorem 5.2.1, the computation of the OT matrices can be recast as a matrix balancing problem. In this section, we discuss Sinkhorn-Knopp (SK) algorithm, one of the most well-known algorithms for the matrix balancing problem, followed by its accelerated variant (named Acc-SK) that can converge in a smaller number of iterations. In contrast to WDA-gd and WDA-eig which use SK algorithm directly, we use Acc-SK to compute the OT matrices. We conclude the section with two examples to demonstrate the efficiency and accuracy of Acc-SK in comparison to SK.

6.2.2.1. *Matrix Balancing Problem.* For the sake of simplicity, we denote the OT matrices $T^{c,c'}(P_k)$ and Euclidean distance matrix $M_{P_k^T X^c, P_k^T X^{c'}}$ as $T^\lambda \in \mathbb{R}_+^{n\times m}$ and $M \in \mathbb{R}_+^{n\times m}$, respectively. According to Theorem 5.2.1,

$$T^\lambda = D(u)KD(v)$$

where $K := e^{-\lambda M}$ is an element-wise exponential, and $u \in \mathbb{R}_+^n$ and $v \in \mathbb{R}_+^m$ satisfy the relation

$$\begin{cases} D(u)KD(v)1_m = \frac{1}{n}1_n \\ D(v)K^T D(u)1_n = \frac{1}{m}1_m. \end{cases} \tag{6.5}$$

Computing $u \in \mathbb{R}_+^n$ and $v \in \mathbb{R}_+^m$ from K to satisfy the relation (6.5) is known as the matrix balancing problem.

There is a vast literature on the matrix balancing problem and one of its earliest works can be dated back as far as 1937 [**66**] where the problem arose in the calculation of traffic flow. Its applications lie in many different areas such as balancing the matrix to improve the sensitivity of the eigenvalue problem [**85**] or balancing the matrix pencil to improve the sensitivity of the generalized eigenvalue problem [**39**, **70**, **116**] in order to compute more accurate eigenvalues, and in optimal transport [**34**] to compute the distance between probability vectors. See [**53**] for further discussions on the applications and the historical remarks of the matrix balancing problems.

The most notable work on the matrix balancing problem is Sinkhorn's paper in 1964 [**97**] that showed the matrix balancing problem is solvable for a positive square matrix. In 1967 [**98**], Sinkhorn further extended his result to prove the existence of a matrix with prescribed row and column sums from a positive rectangular matrix, and with Knopp, derived conditions for the existence of a doubly stochastic matrix from a non-negative square matrix [**99**].

6.2.2.2. *Sinkhorn-Knopp (SK) Algorithm.* Sinkhorn-Knopp (SK) algorithm [**97**, **98**, **99**], also known as RAS method or Bregman's balancing method, is one of the most well-known algorithms for the matrix balancing problem. SK iteratively scales the matrix involving only matrix-vector multiplications, and converges linearly [**47**, **56**, **62**]. Applied to computing the OT matrix T^λ, SK obtains the vectors v and u satisfying (6.5) by an alternating updating scheme on the matrix K:

$$\begin{cases} v_{k+1} = \frac{1}{m}1_m./(K^T u_k) \\ u_{k+1} = \frac{1}{n}1_n./(Kv_{k+1}). \end{cases} \tag{6.6}$$

6.2.2.3. *Reformulation of SK as Nonlinear Mapping and NEPv.* While simple, the SK algorithm is subject to slow convergence [**4**,**62**]. There are variants of the SK algorithm for acceleration. They include a different updating scheme for scaling the matrix [**83**], an inner-outer iteration algorithm based on Newton's method [**63**], a greedy Sinkhorn algorithm called Greenkhorn algorithm [**2**, **3**], and the SCF iteration for solving the NEPv formulation [**4**].

In our algorithm WDA-nepv, we consider accelerating the SK algorithm by solving the NEPv formulation. We note that the SK iteration (6.6) for obtaining the vector v can be rewritten as the following iteration

$$v_{k+1} = R(v_k), \tag{6.7}$$

where

$$R(v) := D^{-1}\left(K^T D^{-1}\left(Kv\right)\frac{1_n}{n}\right)\frac{1_m}{m} \tag{6.8}$$

is a nonlinear mapping. R is a contraction mapping on the space of non-negative vectors [**22**]. Thus, the fixed point of R exists and is unique.

In the following, we show that the fixed point v of the mapping R is an eigenvector corresponding to the largest eigenvalue of $J_R(v) \in \mathbb{R}^{m\times m}$, the Jacobian of the mapping R. First, we have the following theorem due to [**4**, Theorem 5].[1]

[1]Although only positive square matrices were subject to discussion in [**4**], we can extend their results to positive rectangular matrices.

THEOREM 6.2.1. *For a positive vector v, it holds*

$$R(v) = J_R(v)v, \tag{6.9}$$

where J_R is the Jacobian of the mapping R defined as in (6.8).

PROOF. The equation (6.8) can be written as the composition of mappings

$$R(v) = \frac{n}{m}U(K^TS(v)) \tag{6.10}$$

where U and S are defined as

$$U(v) = 1./v \quad \text{and} \quad S(v) = U(Kv). \tag{6.11}$$

Note that we define U as a component-wise reciprocal operator for vectors of any size. In particular, since K is $n \times m$, U in the operator S operates on the vector of size n.

The Jacobian of U is

$$J_U(v) = -D^{-2}(v) = -D^2(1./v) = -D^2(U(v)), \tag{6.12}$$

where the last equality holds by (6.11). The Jacobian of S is

$$J_S(v) = -D^{-2}(Kv)K = -D^2(1_n./(Kv))K = -D^2(U(Kv))K = -D^2(S(v))K, \tag{6.13}$$

where the last two equalities hold by (6.11).

By the chain rule, the Jacobian matrix of the mapping R is

$$J_R(v) = J_{\frac{n}{m}UK^TS}(v) = \frac{n}{m}J_U(K^TS(v)) \cdot K^T \cdot J_S(v).$$

Utilizing the results (6.12), (6.13) and substituting the mappings R (6.10), $J_R(v)$ is further simplified as

$$\begin{aligned} J_R(v) &= \frac{n}{m}[-D^2(U(K^TS(v)))][K^T][-D^2(S(v))K] \\ &= \frac{m}{n}D^2(R(v))K^TD^2(S(v))K \end{aligned} \tag{6.14}$$

Then,

$$\begin{aligned}
J_R(v)v &= \frac{m}{n}D^2(R(v))K^TD^2(S(v))Kv = \frac{m}{n}D^2(R(v))K^TD^2(U(Kv))Kv \\
&= \frac{m}{n}D^2(R(v))K^TD^{-2}(Kv)Kv = \frac{m}{n}D^2(R(v))K^TD^{-1}(Kv)1_n \\
&= \frac{m}{n}D^2(R(v))K^TD(U(Kv))1_n = \frac{m}{n}D^2(R(v))K^TS(v) \\
&= \frac{n}{m}D^{-2}(K^TS(v))K^TS(v) = \frac{n}{m}D^{-1}(K^TS(v))1_m \\
&= \frac{n}{m}D(U(K^TS(v)))1_m = D(R(v))1_m = R(v)
\end{aligned}$$

□

By Theorem 6.2.1, the fixed point v of the mapping R satisfies

$$J_R(v)v = v, \tag{6.15}$$

i.e., v is an eigenvector of $J_R(v)$ corresponding to the eigenvalue $\mu = 1$. Note that the fixed point v is strictly positive and the exponential Euclidean distance matrix K is strictly positive. Consequently, it is clear from the equation (6.14) that $J_R(v)$ is a positive matrix. Therefore, by the Perron-Frobenius theorem [**49**], v must be the eigenvector corresponding to the largest eigenvalue of the NEPv:

$$J_R(v)v = \mu v \tag{6.16}$$

6.2.2.4. *Accelerated Sinkhorn-Knopp (Acc-SK) Algorithm.* The above discussion indicates that the SK algorithm (6.6) is equivalent to finding the eigenvector v corresponding to the largest eigenvalue of $J_R(v)$. To accelerate the SK iteration, we can apply the following SCF iteration for solving the NEPv (6.16):

$$v_{k+1} \leftarrow \text{eigenvector of } \mu_{\max}(J_R(v_k)). \tag{6.17}$$

The SCF iteration (6.17) is referred to as Acc-SK. Once Acc-SK converges to v, u is computed by an extra update (6.6), i.e.,

$$u = \frac{1}{n}1_n./(Kv).$$

In summary, Acc-SK for computing the OT matrix T^{λ} is shown in Algorithm 3. The eigenvalue problem in line 4 of Algorithm 3 can be efficiently solved by a Krylov-subspace eigensolver such as the implicitly restarted Arnoldi method [**100**]. The stopping criteria in line 5 checks whether the distance between v_{k+1} and v_k is smaller than a preset tolerance *tol*.

Algorithm 3 Computation of T^{λ} via NEPv

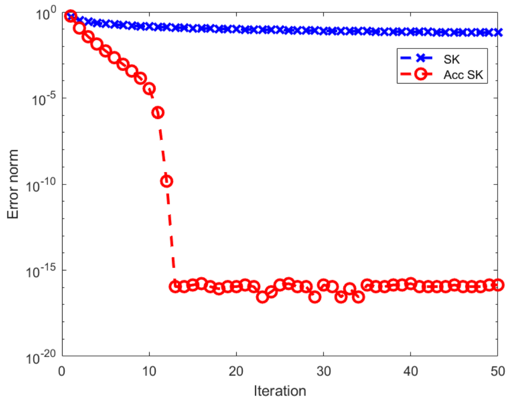
```
Input: Matrix $K \in \mathbb{R}_{+}^{n\times m}$, tolerance tol
Output: the OT matrix $T^{\lambda}$ in (5.9)
 1: Create a starting $v_0 > 0$
 2: for $k = 0, 1, \dots$ do
 3:     Set $J_k = J_R(v_k)$
 4:     Set $v_{k+1}$ as the eigenvector of the largest eigenvalue of $J_k$
 5:     if $d(v_{k+1}, v_k) < tol$ then
 6:         return $v = v_{k+1}$
 7:     end if
 8: end for
 9: Compute $u = \frac{1}{n}1_n./(Kv)$
10: Return $T^{\lambda} = D(u)KD(v)$
```

6.2.2.5. *Advantages of Acc-SK.* Acc-SK significantly improves over SK in number of iterations when the magnitude of the components of an input matrix is small [**4**]. In the context of NTRopt-WDA (5.22), this situation can occur when the regularization parameter λ is sufficiently large, i.e., when the local relationships between data points are emphasized. We illustrate the acceleration of Acc-SK over SK in the following two examples: the first example for general synthetic matrices and the second example for computing the OT matrices of NTRopt-WDA (5.22). For both SK and Acc-SK, the starting vector is taken as $v_0 = \frac{1}{m}1_m$ and the convergence behavior is displayed by the error $\|v_{k+1} - v_k\|_2$. We show that while SK is subject to slow convergence, Acc-SK converges efficiently in a small number of iterations.

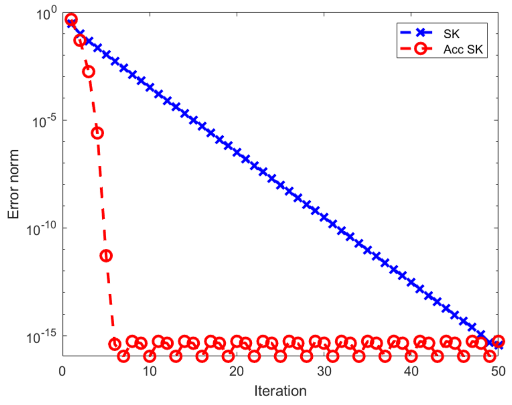
Example 6.2.1. Consider the following synthetic matrices

$$K_1 = \begin{bmatrix} 1 & \epsilon \\ 1 & 1 \end{bmatrix}, \quad K_2 = \begin{bmatrix} 1 & \epsilon \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

where ϵ is a small positive number. The convergence behaviors of SK and Acc-SK for the matrices K_1 and K_2 with $\epsilon = 10^{-8}$ in Figure 6.1 shows that SK is unable to converge within 50 iterations for matrix K_1 and that while it converges for the matrix K_2, it does so slowly. Meanwhile, Acc-SK converges in around 10 iterations for both matrices.



(a) Matrix K_1



(b) Matrix K_2

FIGURE 6.1. Convergence behaviors of SK and Acc-SK on matrices K_1 and K_2. Each plot displays the error $\|v_{k+1} - v_k\|_2$ of SK and Acc-SK.

EXAMPLE 6.2.2. Using the synthetic dataset (see Sec. 6.3.1 for its description), we create data matrices $X^1 \in \mathbb{R}^{2\times 60}, X^2 \in \mathbb{R}^{2\times 100}$. We choose a sufficiently large regularization parameter $\lambda = 1$ to emphasize the local relations of the data points and form exponential Euclidean distance matrices $K^{1,2} \in \mathbb{R}_+^{60\times 100}$ and $K^{1,1} \in \mathbb{R}_+^{60\times 60}$. With this choice of the regularization parameter, some of the components of $K^{1,2}$ and $K^{1,1}$ are small. Figure 6.2 depicts the convergence behavior. We observe that SK exhibits slow convergence for $K^{1,2}$ and does not converge below the error 10^{-5} for $K^{1,1}$. On the other hand, Acc-SK displays fast and accurate convergence, converging in just two iterations for both $K^{1,2}$ and $K^{1,1}$.

6.2.3. Computation of Cross-covariance Matrices $C_b(P_k)$ and $C_w(P_k)$. Recall that the between and within cross-covariance matrices $C_b(P_k)$ and $C_w(P_k)$ defined in (5.23) and (5.24)

$$C_b(P_k) = \sum_{c,c'>c} \sum_{ij} T_{ij}^{c,c'}(P_k)(x_i^c - x_j^{c'})(x_i^c - x_j^{c'})^T,$$

$$C_w(P_k) = \sum_{c} \sum_{ij} T_{ij}^{c,c}(P_k)(x_i^c - x_j^c)(x_i^c - x_j^c)^T$$

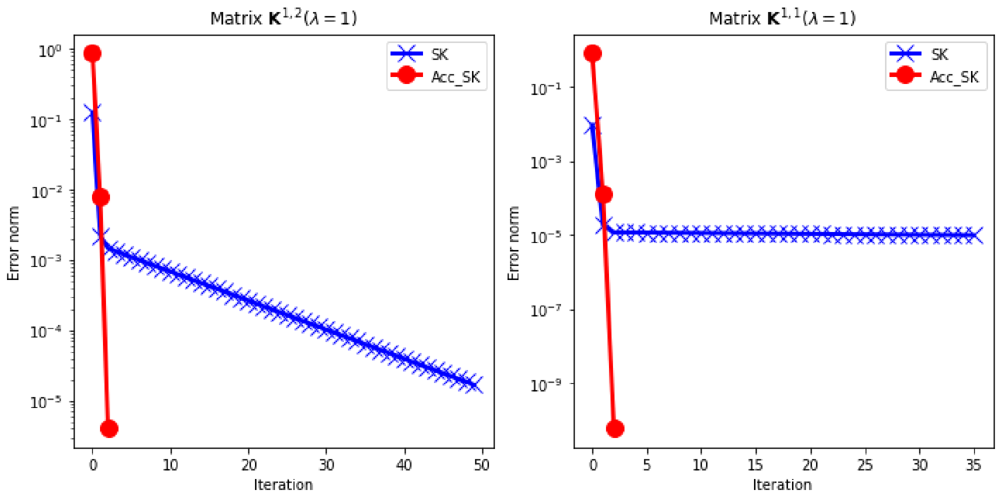


FIGURE 6.2. Convergence behaviors of SK and Acc-SK on matrices $K^{1,2}$ and $K^{1,1}$. Each plot displays the error $\|v_{k+1} - v_k\|_2$ of SK and Acc-SK.

are the weighted sums of the outer products of the data points, where the weights are the components of the OT matrices. The number of double sums in $C_b(P_k)$ and $C_w(P_k)$ are

$$N_b := \sum_{c,c'>c}^{C} N_c N_{c'} \quad \text{and} \quad N_w := \sum_{c}^{C} N_c^2,$$

respectively. Overall, the number of double sums grow quadratically in the number of classes or data points.

A straightforward implementation for computing these cross-covariance matrices is by a double for-loop with N_b and N_w calls to level-2 BLAS in forming the outer products of $C_b(P_k)$ and $C_w(P_k)$, respectively. However, memory access is the bottleneck on modern computing platforms. Unlike level-2 BLAS, level-3 BLAS makes full reuse of data and avoids excessive movement of data to and from memory [**38**]. Therefore, we propose to compute the cross-covariance matrices using level-3 BLAS to improve computational efficiency. Specifically, we reformulate the sum of the outer products as a matrix-matrix multiplication:

$$C_b(P_k) = \widehat{C}_b(P_k)\widehat{C}_b^T(P_k) \quad \text{and} \quad C_w(P_k) = \widehat{C}_w(P_k)\widehat{C}_w^T(P_k) \tag{6.18}$$

where $\widehat{\boldsymbol{C}}_b(P_k) \in \mathbb{R}^{d\times N_b}$ and $\widehat{C}_w(P_k) \in \mathbb{R}^{d\times N_w}$ have columns

$$\sqrt{T_{ij}^{c,c'}(P_k)}(x_i^c - x_j^{c'}) \quad \text{and} \quad \sqrt{T_{ij}^{c,c}(P_k)}(x_i^c - x_j^c),$$

respectively. As level-3 BLAS, the matrix-matrix multiplications (6.18) are far more efficient than forming the sum of outer products. In particular, the improvement in efficiency becomes more prominent as the data dimension gets large. For an instance of the synthetic dataset (see Section 6.3 for its description) with $d = 2000$, the running time of the sum of outer products took 127 seconds while the level-3 BLAS formulation only took 0.63 seconds.

We note that efficient computations of the cross-covariance matrices using level-3 BLAS is a novelty of this work. In WDA-gd, cross-covariance matrices are not formed explicitly. Instead, each regularized Wasserstein distance is computed directly as the weighted sum (5.17) and the reciprocal of the objective function (5.16) is an input in Pymanopt [**108**], an optimizer on matrix manifold. In WDA-eig, the cross-covariance matrices are implemented directly from their double sum formulations using tensor operations.

6.2.4. Solving TRopt by NEPv. Now let us consider the outer iteration for solving the TRopt (6.4). For notation convenience, set

$$A = C_b(P_k), \quad B = C_w(P_k).$$

Note that by (6.18), A and B are symmetric positive semi-definite. Furthermore, given that the number of data points is typically larger than the dimension size, we can safely assume that A and B are positive definite. Then, the corresponding TRopt (6.4) can be solved by Algorithm 1, the SCF iteration on the NEPv (2.7). Algorithm 1 is monotonic, globally convergent to the global maximizer for any initial projection P_0, and has a local quadratic convergence.

6.2.5. WDA-nepv. Combining Algorithm 3 and Algorithm 1, WDA-nepv is presented in Algorithm 4. A possible choice of the initial projection P_0 is using a random orthogonal projection or using the projection obtained as the solution to PCA or as the solution to LDA. In line 4, the initial for Algorithm 3 can be chosen as the uniformly distributed probability vector, i.e., $v_0 = 1_{N_{c'}}/N_{c'}$ and $v_0 = 1_{N_c}/N_c$ for $T^{c,c'}(P_k)$ and $T^{c,c}(P_k)$, respectively. In line 6, the initial

Algorithm 4 WDA-nepv

Input: Data matrices $X^1 \in \mathbb{R}^{d\times N_1}, \ldots, X^C \in \mathbb{R}^{d\times N_c}$, regularization $\lambda \geqslant 0$, tolerance tol
Output: solution P to NTRopt-WDA (5.22)

```
1: Create a starting $P_0$ with $P_0^T P_0 = I_p$
2: for $k = 0, 1, \ldots$ do
3:    Compute $K^{c,c'}(P_k) \in \mathbb{R}^{N_c \times N_{c'}}$ with $[K^{c,c'}(P_k)]_{ij} = e^{-\lambda \|P_k^T x_i^c - P_k^T x_j^{c'}\|_2^2}$ and $K^{c,c}(P_k) \in \mathbb{R}^{N_c \times N_c}$
      with $[K^{c,c}(P_k)]_{ij} = e^{-\lambda \|P_k^T x_i^c - P_k^T x_j^c\|_2^2}$
4:    Compute OT matrices $T^{c,c'}(P_k) \in \mathbb{R}^{N_c \times N_{c'}}$ and $T^{c,c}(P_k) \in \mathbb{R}^{N_c \times N_c}$ by Alg. 3 with $K^{c,c'}(P_k)$
      and $K^{c,c}(P_k)$
5:    Compute $C_b(P_k)$ and $C_w(P_k)$ by level-3 BLAS (6.18)
6:    Compute $P_{k+1}$ by Alg. 1 with $C_b(P_k)$ and $C_w(P_k)$
7:    if $d(P_{k+1}, P_k) < tol$ then
8:       Return $P = P_{k+1}$
9:    end if
10: end for
```

projection for Algorithm 1 can be chosen as P_k, the current kth projection. The stopping criteria $d(P_{k+1}, P_k) < tol$ in line 7 is measured as the largest principal angle between the subspaces spanned by the matrices P_{k+1} and P_k [**49**].

Regarding the complexity of Algorithm 4, suppose that the numbers of data points of classes are the same, i.e., $N = N_c$ for all $c = 1, 2, \ldots, C$. Then, for one iteration of Algorithm 4: $O(C^2)$ many OT matrices are computed, and each OT matrix costs $O(N^2)$ at each iteration of Algorithm 3. The formation of the cross-covariance matrices $C_b(P_k)$ and $C_w(P_w)$ are $O(N^2 d^2)$. The other cost is a Krylov subspace eigensolver for SCF in Algorithm 1, whose leading cost is $O(d^2)$ for the matrix-vector products on the $d \times d$ covariance matrices. In summary, WDA-nepv (Algorithm 4) is quadratic in the number of data points N and the dimension d.

6.2.6. Convergence of WDA-nepv. In this section, we provide the convergence analysis of the proposed WDA-nepv. We first note that given P_k, the inner optimization (Algorithm 3) to compute the OT matrices $T^{c,c'}(P_k)$ is globally convergent. Furthermore, for applications of WDA, the desired accuracy is typically low, say at the range of $tol = 10^{-3}$ to 10^{-5}. Therefore, for simplicity, in our analysis we assume that the OT matrices are accurately computed so that they can be regarded as "*exact*". Similarly, the SCF iteration for TRopts (Algorithm 1) in the outer optimization is also globally convergent so that P_{k+1} is accurately computed to be regarded as "*exact*". Therefore, we have

$$\underbrace{\left[C_b(P_k) - \frac{\mathrm{Tr}(P_{k+1}^T C_b(P_k) P_{k+1})}{\mathrm{Tr}(P_{k+1}^T C_w(P_k) P_{k+1})} C_w(P_k)\right]}_{=:\widetilde{H}_k} P_{k+1} = P_{k+1}\Lambda_k, \tag{6.19}$$

where the eigenvalues of Λ_k consist of the p largest eigenvalues of $\widetilde{H}_k$.

Next, we make the following so-called monotonicity assumption whose rigorous verification remains open but is numerically demonstrated throughout all our numerical tests.

Assumption 6.2.1. *For orthonormal $P, \widehat{P} \in \mathbb{R}^{d\times p}$ and the function $q(P)$ defined in NTRopt-WDA* (5.22)*, if*

$$\frac{\mathrm{Tr}(\widehat{P}^T C_b(P)\widehat{P})}{\mathrm{Tr}(\widehat{P}^T C_w(P)\widehat{P})} \geqslant q(P) + \eta, \tag{6.20}$$

where $\eta \in \mathbb{R}$, then

$$q(\widehat{P}) \geqslant q(P) + c\eta \tag{6.21}$$

for some constant $c > 0$, independent of $P, \widehat{P}$.

The convergence of WDA-nepv is stated in the following theorem. It provides a theoretical justification of the SCF framework for solving NTRopt-WDA (5.22).

Theorem 6.2.2. *Suppose Assumption* (6.2.1) *holds and that the sum of the p smallest eigenvalues of $C_w(P)$ is uniformly bounded below for all orthonormal $P \in \mathbb{R}^{d\times p}$. Similarly, suppose that the sum of the p largest eigenvalues of $C_w(P)$ is uniformly bounded above for all orthonormal $P \in \mathbb{R}^{d\times p}$. Let the sequence $\{P_k\}_{k=0}^{\infty}$ be generated by WDA-nepv (Algorithm 4). The following statements hold.*

(a) $\{q(P_k)\}_{k=0}^{\infty}$ is monotonically increasing and convergent.

(b) $\{P_k\}_{k=0}^{\infty}$ has a convergent subsequence $\{P_k\}_{k\in\mathbb{I}}$, converging to P_*, and

$$\lim_{k\to\infty} q(P_k) = q(P_*).$$

(c) P_* is an orthonormal eigenbasis matrix of $H_* := H(P_*) = C_b(P_*) - q(P_*)\, C_w(P_*)$ associated with its p largest eigenvalues.

(d) If $\lambda_p(H_*) - \lambda_{p+1}(H_*) > 0$, then

$$P_* = \arg \max_{P^T P = I_p} \frac{\mathrm{Tr}(P^T C_b(P_*) P)}{\mathrm{Tr}(P^T C_w(P_*) P)}, \tag{6.22}$$

where $\lambda_i(H_*)$ is the i^{th} largest eigenvalues of H_*.

Proof. (a) Recall that the projection P_{k+1} is computed exactly to form an orthonormal eigenbasis matrix corresponding to the p largest eigenvalues of $\widetilde{H}_k$. Therefore, by the Ky Fan trace theorem (Theorem 2.1.2) P_{k+1} is a solution to the following optimization:

$$\max_{P^T P = I_p} \mathrm{Tr}(P^T \widetilde{H}_k P), \tag{6.23}$$

and we have from (6.19) that

$$0 = \mathrm{Tr}(P_{k+1}^T \widetilde{H}_k P_{k+1}) = \mathrm{Tr}(\Lambda_k) \geqslant \mathrm{Tr}(P_k^T \widetilde{H}_k P_k),$$

which yields (6.20) with $P = P_k$, $\widehat{P} = P_{k+1}$, and $\eta = 0$. Hence, by assumption 6.2.1, we conclude $q(P_{k+1}) \geqslant q(P_k)$.

According to (2.6), we know that $\mathrm{Tr}(P^T C_w(P) P)$ is not smaller than the sum of the p smallest eigenvalues of $C_w(P)$. Since the sum of the p smallest eigenvalues of $C_w(P)$ is assumed to be uniformly bounded from below for any orthonormal P, $\{q(P_k)\}_{k=0}^{\infty}$ is monotonically increasing and bounded and hence convergent.

(b) Since $\{P_k\}_{k=0}^{\infty}$ is a bounded sequence, it has a convergent subsequence $\{P_k\}_{k\in\mathbb{I}}$, converging to an orthonormal P_*. As $q(P)$ is continuous with respect to P, by item (a), we have

$$\lim_{k\to\infty} q(P_k) = \lim_{\mathbb{I}\ni k\to\infty} q(P_k) = q(\lim_{\mathbb{I}\ni k\to\infty} P_k) = q(P_*).$$

(c) Let ω be the uniform upper bound of the sum of the p largest eigenvalues of $C_w(P)$.

Recall that we have (6.19). $\{P_{k+1}\}_{k\in\mathbb{I}}$, as a bounded sequence, has a convergent subsequence $\{P_{k+1}\}_{k\in\mathbb{I}_1}$, converging to an orthonormal $\widehat{P}_*$, where $\mathbb{I}_1 \subset \mathbb{I}$. Letting $\mathbb{I}_1 \ni k \to \infty$

in (6.19) yields

$$\underbrace{\left[C_b(P_*) - \frac{\operatorname{Tr}(\widehat{P}_*^T C_b(P_*)\widehat{P}_*)}{\operatorname{Tr}(\widehat{P}_*^T C_w(P_*)\widehat{P}_*)} C_w(P_*)\right]}_{=:\widetilde{H}_*} \widehat{P}_* = \widehat{P}_* \Lambda_*,$$

where the eigenvalues of Λ_* consist of the p largest eigenvalues of $\widetilde{H}_*$. We claim that P_* is an orthonormal eigenbasis matrix of $\widetilde{H}_*$ associated with its p largest eigenvalues as well, just like $\widehat{P}_*$ is. Otherwise, according to (2.5),

$$\eta = \operatorname{Tr}(\widehat{P}_*^T \widetilde{H}_* \widehat{P}_*) - \operatorname{Tr}(P_*^T \widetilde{H}_* P_*) > 0,$$

yielding

$$\frac{\operatorname{Tr}(\widehat{P}_*^T C_b(P_*)\widehat{P}_*)}{\operatorname{Tr}(\widehat{P}_*^T C_w(P_*)\widehat{P}_*)} = q(P_*) + \frac{\eta}{\operatorname{Tr}(P_*^T C_w(P_*)P_*)} \geqslant q(P_*) + \frac{\eta}{\omega}.$$

By continuity, there is an $k_0 \in \mathbb{I}_1$ such that

$$\frac{\operatorname{Tr}(P_{k_0+1}^T C_b(P_{k_0})P_{k_0+1})}{\operatorname{Tr}(P_{k_0+1}^T C_w(P_{k_0})P_{k_0+1})} \geqslant \frac{\operatorname{Tr}(\widehat{P}_*^T C_b(P_*)\widehat{P}_*)}{\operatorname{Tr}(\widehat{P}_*^T C_w(P_*)\widehat{P}_*)} - \frac{1}{3}\frac{\eta}{\omega},$$

and

$$q(P_{k_0}) \geqslant q(P_*) - \frac{1}{3}\frac{c\eta}{\omega}.$$

Therefore,

$$\frac{\operatorname{Tr}(P_{k_0+1}^T C_b(P_{k_0})P_{k_0+1})}{\operatorname{Tr}(P_{k_0+1}^T C_w(P_{k_0})P_{k_0+1})} \geqslant q(P_*) + \frac{\eta}{\omega} - \frac{1}{3}\frac{\eta}{\omega} = q(P_*) + \frac{2}{3}\frac{\eta}{\omega} \geqslant q(P_{k_0}) + \frac{2}{3}\frac{\eta}{\omega}.$$

By assumption 6.2.1, we get

$$q(P_{k_0+1}) \geqslant q(P_{k_0}) + c\frac{2}{3}\frac{\eta}{\omega} \geqslant q(P_*) - \frac{1}{3}\frac{c\eta}{\omega} + c\frac{2}{3}\frac{\eta}{\omega} = q(P_*) + \frac{1}{3}\frac{c\eta}{\omega} > q(P_*),$$

contradicting all $q(P_k) \leqslant q(P_*)$.

The contradiction indicates that P_* is an orthonormal eigenbasis matrix of $\widetilde{H}_*$ associated with its p largest eigenvalues, and hence $\eta = 0$, i.e.,

$$\frac{\operatorname{Tr}(\widehat{P}_*^T C_b(P_*)\widehat{P}_*)}{\operatorname{Tr}(\widehat{P}_*^T C_w(P_*)\widehat{P}_*)} = q(P_*),$$

implying $\widetilde{H}_* = H_*$. Therefore, in conclusion, P_* is an orthonormal eigenbasis matrix of H_* associated with its p largest eigenvalues.

(d) In proving item (c), we concluded that $\widetilde{H}_* = H_*$ and so we have $H_*\widehat{P}_* = \widehat{P}_*\Lambda_*$ upon letting $\mathbb{I}_1 \ni k \to \infty$ in (6.19), where the eigenvalues of Λ_* consist of the p largest eigenvalue of H_*. By item (c), P_* and $\widehat{P}_*$ are two orthonormal eigenbasis matrices of H_* associated with its p largest eigenvalues. By the assumption that $\lambda_p(H_*) - \lambda_{p+1}(H_*) > 0$, we conclude $\widehat{P}_* = P_*Q$ where $Q \in \mathbb{R}^{p\times p}$ is an orthogonal matrix. Hence, we have

$$P_* = \arg\max_{P^TP=I_p} \frac{\mathrm{Tr}(P^TC_b(P_*)P)}{\mathrm{Tr}(P^TC_w(P_*)P)}.$$

□

6.3. Numerical Experiments

In this section, we demonstrate the convergence, classification accuracy, and scalability of WDA-nepv. The outline of the section is as follows: first, we provide a brief summary of datasets that we perform numerical experiments on. Then, we demonstrate the convergence behaviors of WDA-nepv, followed by a discussion on non-optimal convergence. Using the K-Nearest-Neighbors (KNN) algorithm to measure the classification accuracy, we demonstrate that WDA-nepv performs either competitively or better than WDA-gd and WDA-eig. Finally, we demonstrate that WDA-nepv scales linearly in subspace dimension p, and quadratically in dimension d and in number of data points N.

The experiments were conducted using Python on a PC with an Intel Core i7-7500U processor with 16GB of RAM. To advocate for reproducible research, we share our Python implementation of WDA-nepv for the experiment results presented[2]. The codes of WDA-gd and WDA-eig are provided by their respective authors[3]. For all experiments, the initial projection P_0 is chosen as a random orthogonal matrix. Unless otherwise stated, the stopping tolerance parameter is preset at 10^{-5}.

6.3.1. Datasets.

[2]Github page for WDA-nepv: `https://github.com/gnodking7/WDAnepv`

[3]Code for WDA-gd: `https://pythonot.github.io/auto_examples/others/plot_WDA.html`.
Github page for WDA-eig: `https://github.com/HexuanLiu/WDA_eig`

6.3.1.1. *Synthetic Dataset.* We consider the synthetic dataset used in [**45**, **72**]. The dataset consists of three bi-modal classes such that the corresponding three data matrices are

$$X^1 \in \mathbb{R}^{d \times N_1}, \quad X^2 \in \mathbb{R}^{d \times N_2}, \quad X^3 \in \mathbb{R}^{d \times N_3}. \tag{6.24}$$

Each data point is a d dimensional vector whose first two components are discriminative and the remaining components are Gaussian noise, i.e., drawn from the standard normal distribution $\mathcal{N}(0, 1)$. In particular, each class consists of two separate modes in its discriminative components such that the number of data points is equally split among the two modes.

The discriminative behavior is shown in Figure 6.3 for an example $N_1 = 30, N_2 = 40, N_3 = 30$: the left subplot shows the first two components and the right subplot the next two components.

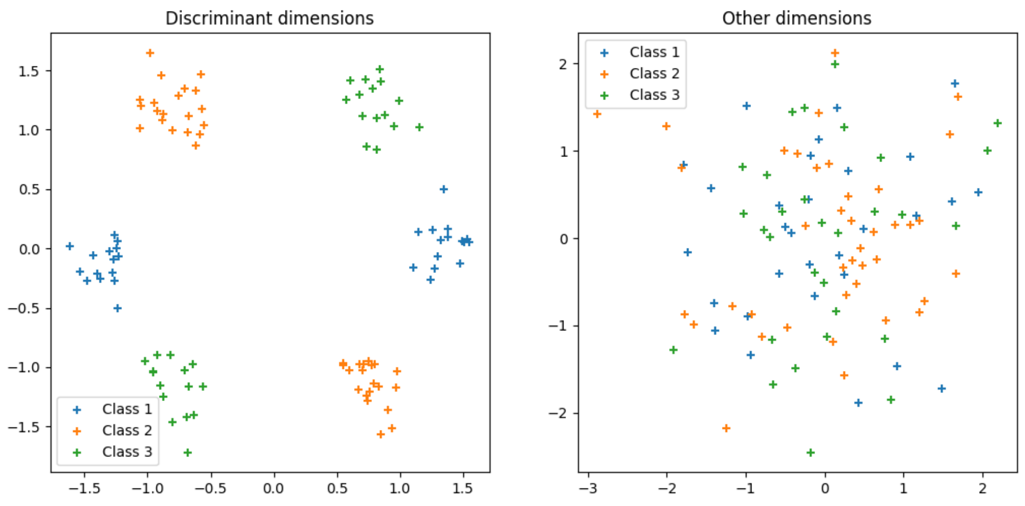


FIGURE 6.3. Each of the three classes is displayed in a different color. The left figure is the plot of the first two components of the data points, and the right figure is the plot of the next two components.

6.3.1.2. *UCI Datasets.* The UCI Repository [**42**] is a collection of datasets that are widely used by the machine learning community for the empirical analysis of machine learning algorithms. In particular, the repository provides a wide variety of datasets that are suitable for clustering and classification tasks. Among the datasets, we choose the real-life datasets named Iris, Wine, Ionosphere, LSVT, and Parkinson's Disease. In Table 6.1, their dimension size, number of data points, and number of classes are displayed.

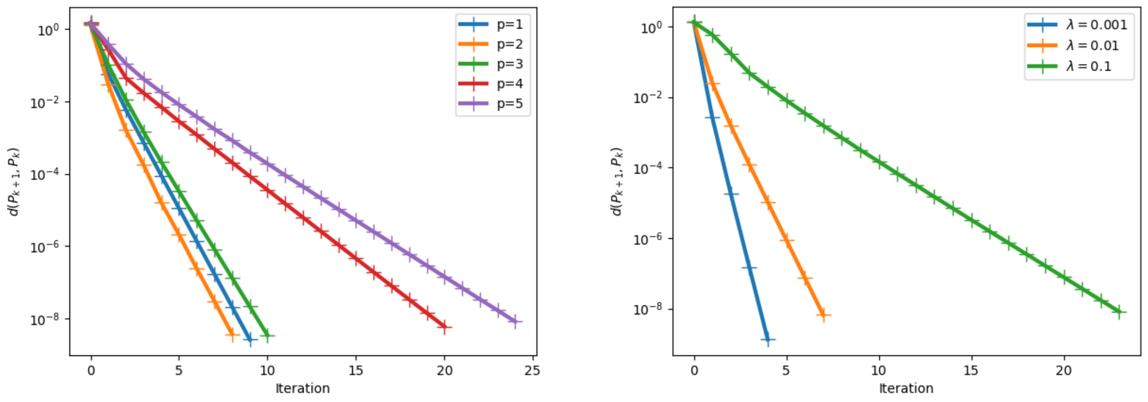
	Iris	**Wine**	**Ionosphere**	**LSVT**	**Parkinson**
Dimension (d)	4	13	34	309	754
Data points ($\sum N_c$)	150	178	351	126	756
Classes (C)	3	3	2	2	2

Table 6.1. UCI datasets description.

6.3.2. WDA-nepv: Convergence Behavior.

Example 6.3.1. To investigate the convergence behavior of WDA-nepv, we use a synthetic dataset with a dimension of $d = 10$ and three classes of data points of sizes $(N_1, N_2, N_3) = (30, 40, 30)$. We perform two experiments to investigate the convergence behavior of WDA-nepv:

- With fixed regularization parameter $\lambda = 0.01$, the convergence behavior of WDA-nepv is depicted for various subspace dimensions $p \in \{1, 2, 3, 4, 5\}$; see Figure 6.4(a).
- With fixed subspace dimension $p = 2$, the convergence behavior of WDA-nepv is reported for various regularization parameters $\lambda \in \{0.001, 0.01, 0.1\}$; see Figure 6.4(b).



(a) Varying p (b) Varying λ

Figure 6.4. Convergence behavior of WDA-nepv on the synthetic dataset.

Both plots in Figure 6.4 illustrate linear convergence for WDA-nepv regardless of the choice of the regularization parameter or the subspace dimension. WDA-nepv achieves the fastest convergence rate for the subspace dimension $p = 2$, the dimension size of the true discriminative subspace of the dataset. For a similar subspace dimension size $p = 1$ WDA-nepv achieves a similar convergence rate as $p = 2$, while as the subspace dimension p increases the convergence rate slows down. We also observe that the convergence rate slows down as the regularization parameter λ increases.

When λ is small, NTRopt-WDA (5.22) can be considered as LDA with a small perturbation to the cross-covariance matrices. In this case, as LDA is known to have a local quadratic convergence [**25**], WDA-nepv converges faster than the cases where λ is larger.

6.3.2.1. *Many Local Maxima and Non-optimal Convergence.* NTRopt-WDA (5.22) is a highly nonlinear, non-convex bi-level optimization that often has many local optimizers. The following example illustrates this.

Example 6.3.2. Let us consider the Synthetic dataset with $d = 2$ and the number of data points $N_1 = 30, N_2 = 40, N_3 = 30$. We desire to compute a projection P that projects these data points onto a subspace of dimension $p = 1$, i.e., P is a normalized vector $p \in \mathbb{R}^{2\times 1}$. We define $q(P)$ as the value of the objective function of NTRopt-WDA at a normalized vector p. That is,

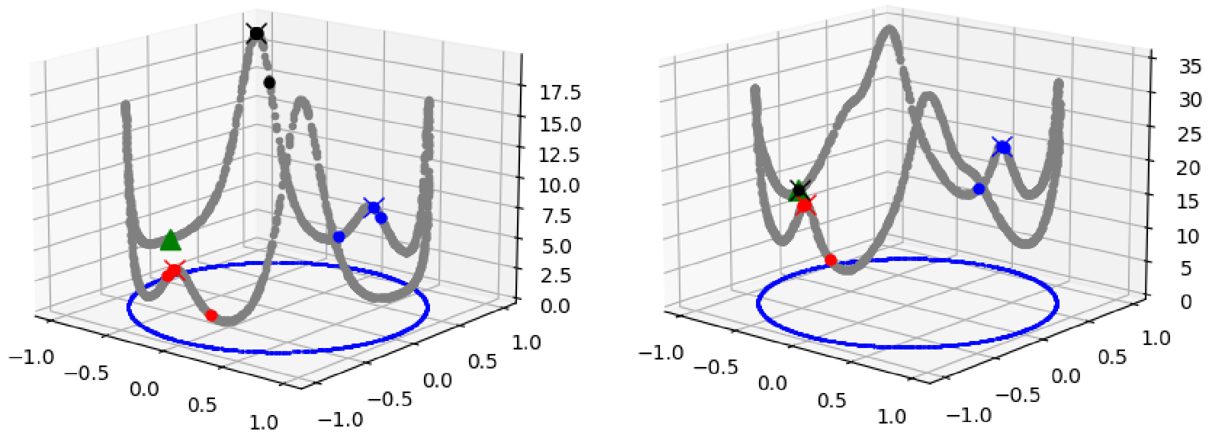
$$q(p) := \frac{\mathrm{Tr}(p^T C_b(p) p)}{\mathrm{Tr}(p^T C_w(p) p)}. \tag{6.25}$$

Figure 6.5 plots the values $q(P)$ (gray points) at 2000 random normalized vectors p, where the regularization parameter is chosen as $\lambda = 0.1$ for plot (a) and $\lambda = 1$ for plot (b). We observe that for both regularization parameters, there are many local maxima.

Finding a global optimizer of a non-convex problem is NP-hard. Algorithms for non-convex problems are prone to converge towards a local optimizer, and consequently, lead to a suboptimal solution. A suboptimal solution is not desirable in optimization and for the task at hand. For instance, in a classification task, relying on a suboptimal projection often proves insufficient for identifying the subspace that effectively discriminates the projected data points.

Example 6.3.3. In order to analyze the convergence behaviors of WDA-gd, WDA-eig and WDA-nepv, we examine the history of their NTRopt-WDA values for the problem described in Example 6.3.2. That is, for projections p_k obtained at each iteration of an algorithm, we examine how the values $q(p_k)$ (6.25) are changing. Figure 6.5 illustrates the local convergence behavior for WDA-nepv, WDA-gd, and WDA-eig.

For both regularization parameters $\lambda \in \{0.1, 1\}$, the same initial projection p_0 corresponding to a local minimizer is used and shared by all three algorithms. In the figure, the value $q(p_0)$ is



(a) $\lambda = 0.1$ (b) $\lambda = 1$

FIGURE 6.5. Plots of NTRopt-WDA values $q(p_k)$ (gray points) at random normalized vectors, with regularization parameter $\lambda = 0.1$ in (a) and $\lambda = 1$ in (b). For all three considered algorithms, the initial projection p_0 (a green triangle) is chosen as a local minimizer. The values $q(p_k)$ of WDA-nepv (red circles), WDA-gd (black circles), and WDA-eig (blue circles) are displayed for the projections p_k obtained at each iteration of an algorithm. For each algorithm, the value $q(p_*)$ for the converged projection p_* is displayed as the '$\times$' symbol.

represented as a green triangle, while the values $q(p_k)$ are shown as red circles for WDA-nepv, black circles for WDA-gd, and blue circles for WDA-eig. The final value $q(p_*)$ for the converged projection p_* is displayed as the '$\times$' symbol, with each color corresponding to the respective algorithm.

In Figure 6.5(a), we observe that for a regularization parameter $\lambda = 0.1$, WDA-nepv and WDA-eig converge towards a local maximizer while WDA-gd successfully finds a global maximizer. However, in Figure 6.5(b), when $\lambda = 1$, we also observe that WDA-gd fails to find a global maximizer and remains trapped in the initial local minimizer.

EXAMPLE 6.3.4. In addition, we use the UCI datasets Wine and Iris to further illustrate the convergence towards non-global local optimizers. We set the regularization parameter as $\lambda = 0.01$ for the Wine dataset and $\lambda = 1$ for the Iris dataset. For both datasets, we solve the NTRopt-WDA problem for subspace dimensions $p \in \{2, 3\}$. The history of the NTRopt-WDA values, denoted as $q(P_k)$, is plotted in Figure 6.6. All three algorithms share the same initial projection P_0, resulting in the same value of $q(P_0)$.

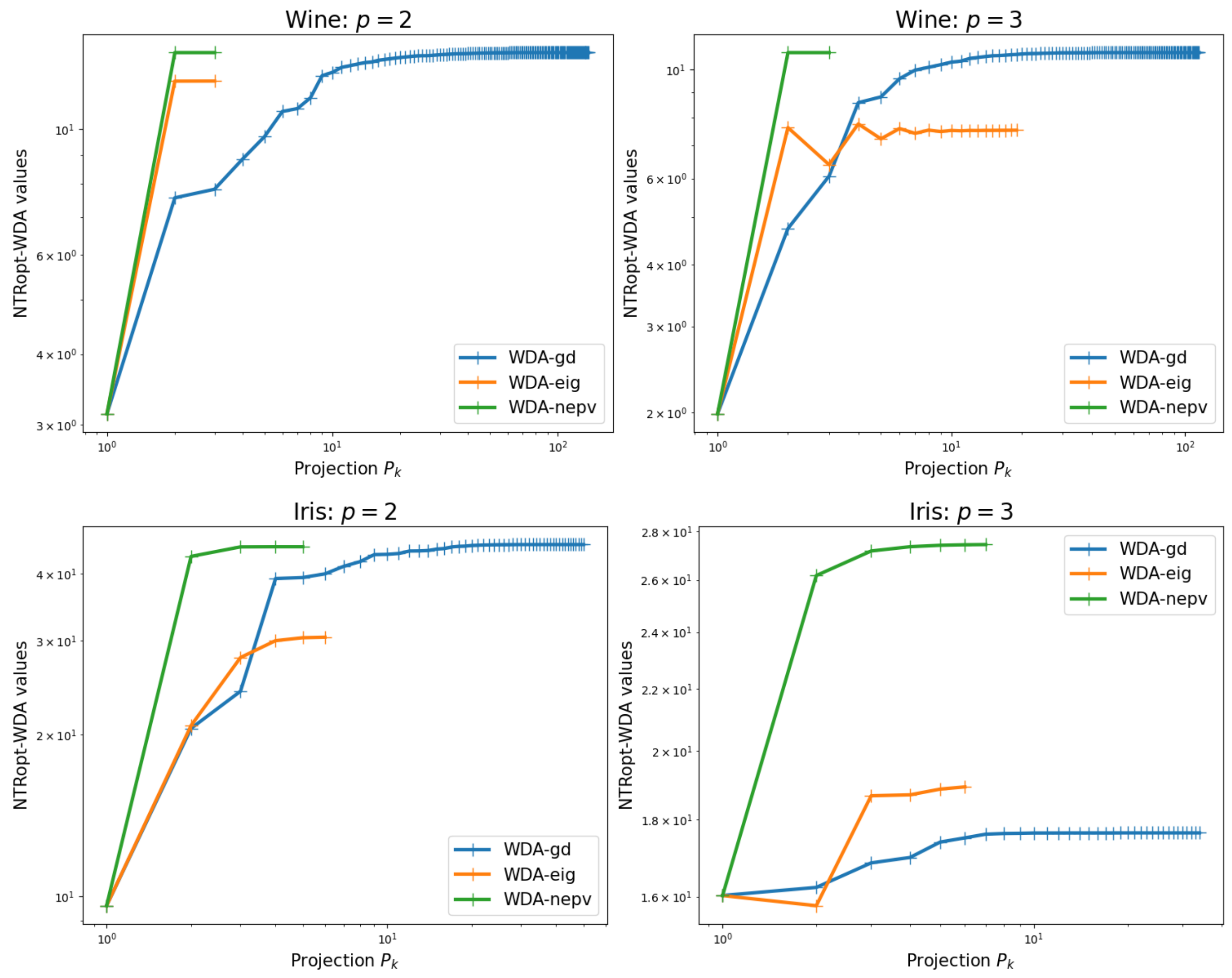


FIGURE 6.6. History of the NTRopt-WDA values for Wine and Iris. Subspace dimensions $p \in \{1, 2, 3\}$ are considered and regularization parameters are set at $\lambda = 0.01$ and $\lambda = 1$ for Wine and Iris, respectively.

For the Wine dataset, we observe that both WDA-gd and WDA-nepv achieve the same optimal value, $q(P_*)$, while WDA-eig's optimal value, $q(P_*)$, is smaller for both $p = 2$ and $p = 3$, indicating that WDA-eig found a suboptimal solution. This highlights the suboptimality of WDA-eig's surrogate ratio trace model (6.1).

Regarding the Iris dataset, WDA-nepv obtains the largest optimal value, $q(P_*)$, among the three algorithms for all considered subspace dimensions. Moreover, WDA-nepv demonstrates faster convergence, requiring significantly fewer iterations compared to WDA-gd.

Both Example 6.3.3 and Example 6.3.4 illustrate that convergence to non-global optima can be a prevalent issue for all three algorithms. The convergence behavior varies across datasets, with instances where some or all three algorithms successfully converge to the same optimizer, as well as instances where they converge to different optimizers. Notably, when the initial projection is chosen close to the global maximizer, it can lead to global convergence. However, since the global maximizer is typically unknown in advance, obtaining an initial projection close to it may require multiple attempts.

6.3.3. WDA-nepv: Classification Accuracy. One of the many goals of DR methods is to derive a projection such that the projected data vectors maintain or amplify the coherent structure of the original dataset. For a supervised linear DR method, one hopes to obtain an optimal projection matrix P such that the class structure of the original data vectors in the projected subspace is more pronounced. For this reason, the effectiveness of a supervised linear DR method is often measured by the accuracy of classification on the projected data vectors. Measuring the accuracy of NTRopt-WDA (5.22) is no exception. To evaluate the classification accuracy of an algorithm, we follow the following conventional steps:

(1) Randomly divide a given dataset into a training dataset and a testing dataset, with equal size of 50% each.
(2) Compute the optimal projection matrix P_* using the training dataset.
(3) Project the testing dataset onto the lower-dimensional subspace using the optimal projection matrix P_*.
(4) Employ the K-Nearest-Neighbors classifier (KNN) on the projected testing dataset to compute the classification accuracy.

The classification accuracy is quantified in terms of prediction error. A smaller error indicates better performance.

Example 6.3.5. Using the UCI datasets Wine, Ionosphere, LSVT, and Parkinson's Disease, the classification accuracy of WDA-nepv is compared with the classification accuracy of WDA-gd and WDA-eig. As suggested in WDA-eig [**72**], we add a small perturbation term ϵI_d with $\epsilon = 1$ to the matrix $C_w(P_k)$. This enforces the positive definiteness of $C_w(P_k)$ in practice and helps the robustness of the computation of the eigenvalue problems. The regularization parameter $\lambda = 0.01$ is fixed throughout and the stopping tolerance parameter is set at 10^{-5}. Each experiment is repeated 20 times, and the average and the min-max interval of classification errors are reported. Two different experiments are considered; one to observe the algorithm behavior in KNN number and another in subspace dimension.

- With fixed subspace dimension $p = 5$, various KNN numbers $K \in \{1, 3, 5, 7, 9, 11, 13, 15, 17, 19\}$ are considered; see left column of Figure 6.7.
- With fixed KNN number $K = 11$, various subspace dimensions $p \in \{1, 2, 3, 4, 5\}$ are considered; see right column of Figure 6.7.

For Wine and Ionosphere, we observe that WDA-nepv performs comparable to WDA-gd and WDA-eig, achieving similar accuracy as the other two WDA algorithms - when looking at the averages, less than 1% classification accuracy difference exists between WDA-nepv and the best performing algorithm for all considered values of KNN numbers and subspace dimensions.

For higher dimensional datasets LSVT and Parkinson, WDA-nepv achieves lowest average prediction errors and has the smallest min-max error intervals among the three algorithms. For LSVT, WDA-nepv achieves, on average, at least 2% to 4% higher classification accuracy for considered KNN numbers, and at least 1% to 4% higher classification accuracy for considered subspace dimensions. For Parkinson, WDA-nepv achieves, on average, at least 2% to 4% higher classification accuracy for considered KNN numbers, and at least 3% to 5% higher classification accuracy for considered subspace dimensions.

6.3.4. WDA-nepv: Timing and Scalability.

We demonstrate the efficiency of WDA-nepv by reporting its running time and its scalability.

Example 6.3.6. We compare the running time of WDA-gd, WDA-eig, and WDA-nepv on the UCI datasets Wine, Ionosphere, LSVT, and Parkinson. For the fixed regularization parameter $\lambda = 0.01$, we measure the running time for different subspace dimensions $p \in \{3, 4, 5\}$ for Wine and Ionosphere, and $p \in \{15, 20, 25\}$ for larger dimensional datasets LSVT and Parkinson. We repeat the experiment 20 times and report the average running times are shown in the following table:

	$p = 3$	$p = 4$	$p = 5$
WDA-gd	1.854	2.354	2.078
WDA-eig	**0.019**	**0.018**	**0.017**
WDA-nepv	0.031	0.028	0.028

(A) Wine

	$p = 3$	$p = 4$	$p = 5$
WDA-gd	12.234	12.530	11.896
WDA-eig	0.868	0.902	0.891
WDA-nepv	**0.242**	**0.241**	**0.284**

(B) Ionosphere

	$p = 15$	$p = 20$	$p = 25$
WDA-gd	11.332	12.965	14.106
WDA-eig	13.689	14.007	14.153
WDA-nepv	**4.683**	**4.811**	**5.068**

(C) LSVT

	$p = 15$	$p = 20$	$p = 25$
WDA-gd	1060.76	1000.56	1001.29
WDA-eig	258.33	285.68	281.66
WDA-nepv	**180.32**	**189.82**	**205.89**

(D) Parkinson

For a small dimensional Wine, we observe that while WDA-eig has the shortest running time, the running time of WDA-nepv only differs from WDA-eig by around 0.01 seconds. For larger dimensional Ionosphere, LSVT, and Parkinon, however, WDA-nepv achieves a running time that is more than half of the running time of WDA-gd and WDA-eig. Unlike WDA-gd, which incurs heavy computational costs from computing the derivatives, WDA-nepv is derivative-free and has a short running time. Additionally, the efficient use of level-3 BLAS for the cross-covariance matrices gives WDA-nepv an edge over WDA-eig in terms of running time.

Example 6.3.7. We demonstrate that WDA-nepv scales linearly in subspace dimension p and quadratically in data dimension d, and number of data points N using the Synthetic dataset. The regularization parameters $\lambda = 0.1, 0.01, 0.001$ are considered, and the reported scalability results are averaged over 100 trials. Three experiments are performed:

- With dimension $d = 10$ and number of total data points $N = 100$ with $(N_1, N_2, N_3) = (30, 40, 30)$, various subspace dimensions $p \in \{1, 2, 3, 4, 5\}$ are considered.
- With subspace dimension $p = 2$ and number of total data points $N = 100$ with $(N_1, N_2, N_3) = (30, 40, 30)$, various dimensions $d \in \{80, 160, 320, 640, 1280, 2560\}$ are considered.
- With dimension $d = 50$ and subspace dimension $p = 2$, various number of total data points $N \in \{100, 200, 300, 500, 1000\}$ are considered.

Figure 6.8 illustrates that larger λ incurs more running time. As discussed previously, as λ approaches zero, WDA-nepv becomes increasingly similar to LDA, which has a local quadratic convergence. Therefore, convergence speed is generally faster for small values of λ. The left plot indicates that the running time of WDA-nepv scales linearly in subspace dimension p. This linear scalability in p is due to the matrix-vector multiplications involved in the exponential Euclidean distance matrices K and the number of dominant eigenvectors in (TRopt) computations. The scalability in data dimension d and number of data points N is displayed in the center and right log-log plots, respectively, in order to exemplify their quadratic scalability. Moreover, a line of quadratic function is included in the log-log plots. We observe that the slope of the lines corresponding to the running time of WDA-nepv match closely with the line of a quadratic function, indicating that the running time of WDA-nepv scales quadratically in data dimension d and number of data points N. These quadratic scalability are due to the level-3 BLAS in the evaluation of the cross-covariance matrices $C_b(P)$ and $C_w(P)$ (6.18).

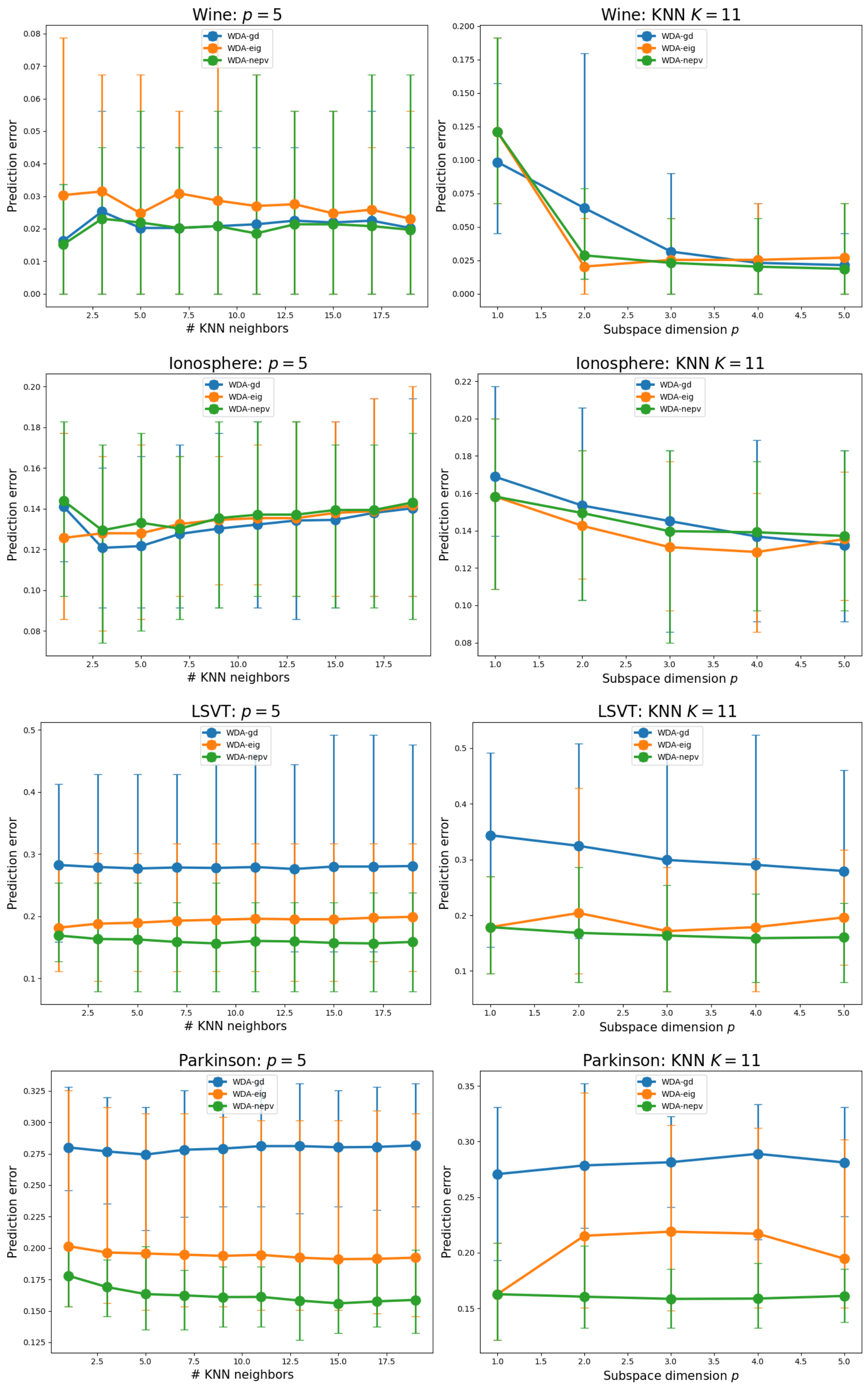


FIGURE 6.7. Prediction errors for WDA-gd, WDA-eig, and WDA-nepv.

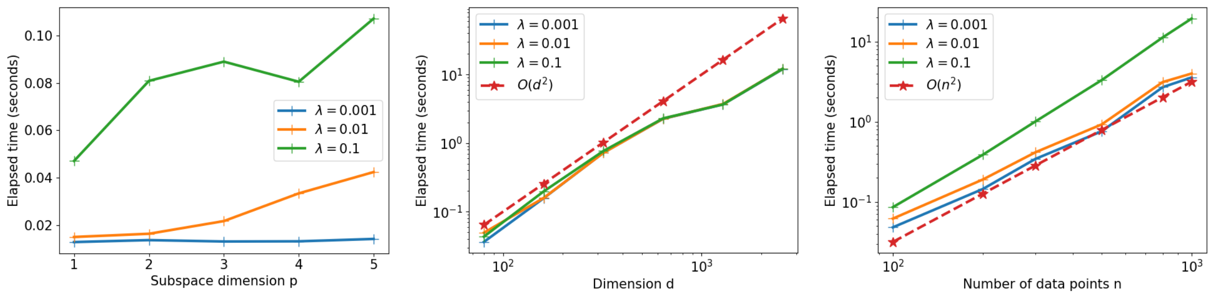


FIGURE 6.8. Scalability of WDA-nepv on the synthetic dataset: varying subspace dimension p (left); varying dimension d (middle); varying number of data points N (right).

CHAPTER 7

Conclusion

The dissertation presented eigenvector algorithms to solve the nonlinear Rayleigh quotient optimization and the nonlinear trace ratio optimization. As preliminaries, we discussed the classical result of the connection between the Rayleigh quotient optimization and a linear eigenvalue problem. In addition, we showed that the trace ratio optimization is equivalent to an eigenvector-dependent nonlinear eigenvalue problem. Afterwards, we investigated the robust common spatial pattern, a signal processing method in brain-computer interface system. In this context, we encountered the nonlinear Rayleigh quotient optimization, which we characterized as an eigenvector-dependent nonlinear eigenvalue problem. The effectiveness of the self-consistent field iteration in solving this nonlinear eigenvalue problem was demonstrated through numerous numerical results using real-world data, highlighting its advantages over existing algorithms. Finally, we explored Wasserstein discriminant analysis, a dimensionality reduction method in machine learning that can dynamically capture global and local relations among the data. The bi-level optimization structure of Wasserstein discriminant analysis is formulated into a nonlinear trace ratio optimization. We proposed an eigenvector algorithm that solves the eigenvector-dependent nonlinear eigenvalue problems in both the inner and outer optimizations of Wasserstein discriminant analysis. Furthermore, we provided numerical results showcasing the convergence, classification rates, and scalability of the proposed eigenvector algorithm for Wasserstein discriminant analysis.

Mathematically, a main open question remains regarding the nonlinear Rayleigh quotient optimization and the nonlinear trace ratio optimization. Given their linear counterparts are linked to the linear eigenvalue problem and the eigenvector-dependent nonlinear eigenvalue problem, respectively, a natural question arises:

> *Under what conditions can we characterize the nonlinear Rayleigh quotient optimization and the nonlinear trace ratio optimization with an eigenvalue problem?*

Specifically, we would like to understand the conditions of the nonlinear matrices $A(x), B(x)$, for the nonlinear Rayleigh quotient optimization, and $A(X), B(X)$, for the nonlinear trace ratio optimization, that impact the answer to the above question.

From an algorithmic perspective, understanding the behavior of the self-consistent field iteration for solving the eigenvector-dependent nonlinear eigenvalue problem is critical. We aim to study the conditions under which the convergence of the self-consistent field iteration is guaranteed for the related eigenvector-dependent nonlinear eigenvalue problems. Additionally, we plan to explore various strategies to accelerate its convergence.

APPENDIX A

Vector and Matrix Calculus

Let x be a vector of size n,

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n,$$

and let X be a matrix of size $n \times p$

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \in \mathbb{R}^{n \times p}.$$

The gradient of a differentiable function $a(x) : \mathbb{R}^n \to \mathbb{R}$ with respect to x is defined as

$$\nabla a(x) := \begin{bmatrix} \frac{\partial a(x)}{\partial x_1} \\ \frac{\partial a(x)}{\partial x_2} \\ \vdots \\ \frac{\partial a(x)}{\partial x_n} \end{bmatrix} \in \mathbb{R}^n. \tag{A.1}$$

Similarly, the gradient of a differentiable vector-valued function $y(x) : \mathbb{R}^n \to \mathbb{R}^m$ with respect to x is defined as

$$\nabla y(x) := \begin{bmatrix} \frac{\partial y_1(x)}{\partial x_1} & \frac{\partial y_2(x)}{\partial x_1} & \cdots & \frac{\partial y_m(x)}{\partial x_1} \\ \frac{\partial y_1(x)}{\partial x_2} & \frac{\partial y_2(x)}{\partial x_2} & \cdots & \frac{\partial y_m(x)}{\partial x_2} \\ \vdots & \vdots & & \vdots \\ \frac{\partial y_1(x)}{\partial x_n} & \frac{\partial y_2(x)}{\partial x_n} & \cdots & \frac{\partial y_m(x)}{\partial x_n} \end{bmatrix}$$

$$= \begin{bmatrix} \nabla y_1(x) & \nabla y_2(x) & \cdots & \nabla y_m(x) \end{bmatrix} \in \mathbb{R}^{n\times m}, \tag{A.2}$$

where $\nabla y_i(x) \in \mathbb{R}^n$ is the gradient of the i^{th} component of $y(x)$.

In the following two lemmas, the first lemma consists of basic vector and matrix gradients involving a symmetric matrix whose proofs can be found in a textbook such as [**35**, Appendix D]. For the second lemma, we provide the proofs.

LEMMA A.0.1. *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix. Then, the following gradients hold:*

(a) $\nabla(Ax) = A$.

(b) $\nabla(x^T Ax) = 2Ax$.

(c) $\nabla \mathrm{Tr}(X^T AX) = 2AX$.

LEMMA A.0.2. *Let $a(x) : \mathbb{R}^n \to \mathbb{R}$ be a differentiable real-valued function of x. Let $y(x) : \mathbb{R}^n \to \mathbb{R}^m$ be a differentiable vector-valued function of x. Let $B(x) : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ be a differentiable matrix-valued function of x, where the column vectors of $B(x)$ are represented as $b_i(x)$ for $i = 1, 2, \ldots, m$.*

(a) Given a diagonal matrix $W \in \mathbb{R}^{m\times m}$, $\nabla(y(x)^T Wy(x)) = 2\nabla y(x)Wy(x) \in \mathbb{R}^n$.

(b) $\nabla(B(x)y(x)) = \sum_{i=1}^m y_i(x)\nabla b_i(x) + \nabla y(x)B(x)^T \in \mathbb{R}^{n\times n}$.

(c) $\nabla(a(x)y(x)) = \nabla a(x)y(x)^T + a(x)\nabla y(x) \in \mathbb{R}^{n\times m}$.

PROOF. (a) We have $y(x)^T Wy(x) = \sum_{i=1}^m w_i(y_i(x))^2$ where w_i is the i^{th} diagonal element of W. Then, by the chain rule,

$$\nabla(y(x)^T Wy(x)) = \sum_{i=1}^m 2w_i y_i(x)\nabla y_i(x),$$

which is precisely $2\nabla y(x)Wy(x)$.

(b) We have

$$B(x)y(x) = \begin{bmatrix} b_{11}(x) & \cdots & b_{1m}(x) \\ \vdots & & \vdots \\ b_{n1}(x) & \cdots & b_{nm}(x) \end{bmatrix} \begin{bmatrix} y_1(x) \\ \vdots \\ y_m(x) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^m b_{1j}(x)y_j(x) \\ \vdots \\ \sum_{j=1}^m b_{nj}(x)y_j(x) \end{bmatrix} \in \mathbb{R}^n,$$

so that the i^{th} column of the gradient $\nabla(B(x)y(x)) \in \mathbb{R}^{n\times n}$ is the gradient of $B(x)y(x)$'s i^{th} element:

$$\nabla(\sum_{j=1}^m b_{ij}(x)y_j(x)) = \sum_{j=1}^m y_j(x)\nabla b_{ij}(x) + \sum_{j=1}^m b_{ij}(x)\nabla y_j(x).$$

$\nabla(B(x)y(x))$ can then be written as a sum of two matrices

$$\nabla(B(x)y(x)) = \mathcal{B}(x) + \mathcal{Y}(x),$$

where

$$\mathcal{B}(x) := \begin{bmatrix} \sum_{j=1}^m y_j(x)\nabla b_{1j}(x) & \sum_{j=1}^m y_j(x)\nabla b_{2j}(x) & \cdots & \sum_{j=1}^m y_j(x)\nabla b_{nj}(x) \end{bmatrix} \in \mathbb{R}^{n\times n}$$

and

$$\mathcal{Y}(x) := \begin{bmatrix} \sum_{j=1}^m b_{1j}(x)\nabla y_j(x) & \sum_{j=1}^m b_{2j}(x)\nabla y_j(x) & \cdots & \sum_{j=1}^m b_{nj}(x)\nabla y_j(x) \end{bmatrix} \in \mathbb{R}^{n\times n}.$$

By decomposing, $\mathcal{B}(x)$ is equal to the sum of m many $n \times n$ matrices:

$$\begin{aligned}
\mathcal{B}(x) &= \begin{bmatrix} y_1(x)\nabla b_{11}(x) & y_1(x)\nabla b_{21}(x) & \cdots & y_1(x)\nabla b_{n1}(x) \end{bmatrix} \\
&\quad + \begin{bmatrix} y_2(x)\nabla b_{12}(x) & y_2(x)\nabla b_{22}(x) & \cdots & y_2(x)\nabla b_{n2}(x) \end{bmatrix} \\
&\quad + \cdots + \begin{bmatrix} y_m(x)\nabla b_{1m}(x) & y_m(x)\nabla b_{2m}(x) & \cdots & y_m(x)\nabla b_{nm}(x) \end{bmatrix} \\
&= y_1(x)\begin{bmatrix} \nabla b_{11}(x) & \nabla b_{21}(x) & \cdots & \nabla b_{n1}(x) \end{bmatrix} \\
&\quad + y_2(x)\begin{bmatrix} \nabla b_{12}(x) & \nabla b_{22}(x) & \cdots & \nabla b_{n2}(x) \end{bmatrix} \\
&\quad + \cdots + y_m(x)\begin{bmatrix} \nabla b_{1m}(x) & \nabla b_{2m}(x) & \cdots & \nabla b_{nm}(x) \end{bmatrix} \\
&= \sum_{i=1}^{m} y_i(x)\nabla b_i(x)
\end{aligned}$$

where $\nabla b_i(x)$ is the gradient of the i^{th} column of $B(x)$.

For $\mathcal{Y}(x)$, note that

$$\nabla y(x)B(x)^T = \sum_{j=1}^{m} \nabla y_j(x)b_j(x)^T.$$

This indicates that the i^{th} column of $\nabla y(x)B(x)^T$ corresponds to $\sum_{j=1}^{m} b_{ij}(x)\nabla y_j(x)$, which is the i^{th} column of $\mathcal{Y}(x)$. Hence, we have $\mathcal{Y}(x) = \nabla y(x)B(x)^T$.

(c) We have

$$a(x)y(x) = \begin{bmatrix} a(x)y_1(x) \\ a(x)y_2(x) \\ \vdots \\ a(x)y_m(x) \end{bmatrix}$$

so that

$$\begin{aligned}
\nabla(a(x)y(x)) &= \Big[\nabla(a(x)y_1(x)) \quad \cdots \quad \nabla(a(x)y_m(x))\Big] \\
&= \Big[y_1(x)\nabla a(x) + a(x)\nabla y_1(x) \quad \cdots \quad y_m(x)\nabla a(x) + a(x)\nabla y_m(x)\Big] \\
&= \Big[y_1(x)\nabla a(x) \quad \cdots \quad y_m(x)\nabla a(x)\Big] + \Big[a(x)\nabla y_1(x) \quad \cdots \quad a(x)\nabla y_m(x)\Big] \\
&= \nabla a(x)y(x)^T + a(x)\nabla y(x).
\end{aligned}$$

□

APPENDIX B

Dinkelbach's Algorithm

Introduced in 1967, the Dinkelbach's algorithm [**37**] is an approach used to solve nonlinear fractional programmings through the utilization of a parametrization technique. It is based on the observation that the global optimum of the nonlinear fractional programming is the unique root of the closely related parametric function, which is defined as a nonlinear parametric programming. Moreover, the solution of the parametric programming at the root of the parametric function corresponds to the global optimal solution of the nonlinear fractional programming. By solving the parametric programming, the Dinkelbach's algorithm obtains a parameter sequence that exhibits monotonic behavior and converges globally to the root of the parametric function.

B.1. Preliminaries

Let $N(X)$ and $D(X)$ be continuous and real-valued functions defined over a compact, connected subset $S \subseteq \mathbb{R}^{n\times p}$ with $D(X) > 0$ for all $X \in S$. We are interested in solving the following nonlinear fractional programming (NFP):

$$\text{(NFP)} \qquad \max_{X\in S} \frac{N(X)}{D(X)}.$$

A closely related problem to (NFP) is the following nonlinear parametric programming (NPP):

$$\text{(NPP)} \qquad f(q) = \max_{X\in S}\{N(X) - qD(X)\}\ , \qquad q \in \mathbb{R}.$$

In both (NFP) and (NPP), the functions $N(X)$ and $D(X)$ are continuous on the compact subset S, and $D(X) > 0$. This guarantees the existence of solutions for both problems.

Lemma B.1.1 ([**37**]). *For the parametric function $f(q)$ of* (NPP)*, the following properties hold:*

(a) $f(q)$ is a convex function over $\mathbb{R}$.

(b) $f(q)$ is a continuous function over $\mathbb{R}$.

(c) *$f(q)$ is a strictly monotonic decreasing function, i.e., $f(q'') < f(q')$ if $q' < q''$.*

(d) *$f(q)$ has a unique root, denoted by q_*.*

(e) *If $q_0 = \frac{N(X_0)}{D(X_0)}$ for some $X_0 \in S$, then $f(q_0) \geqslant 0$.*

Proof. (a) Let X_t maximize (NPP) corresponding to $f(tq' + (1-t)q'')$ with $q' \neq q''$ and $0 \leqslant t \leqslant 1$. Then,

$$\begin{aligned} f(tq' + (1-t)q'') &= N(X_t) - (tq' + (1-t)q'')D(X_t) \\ &= t[N(X_t) - q'D(X_t)] + (1-t)[N(X_t) - q''D(X_t)] \\ &\leqslant tf(q') + (1-t)f(q''). \end{aligned}$$

(b) Since f is finite and convex over an open set $\mathbb{R}$, it is continuous.

(c) Let X'' maximize (NPP) corresponding to $f(q'')$. Then,

$$f(q'') = N(X'') - q''D(X'') < N(X'') - q'D(X'') \leqslant f(q').$$

(d) This follows since f is continuous, monotonically decreasing, and $\lim_{q\to-\infty} f(q) = \infty$ and $\lim_{q\to\infty} f(q) = -\infty$.

(e) Let $q_0 = \frac{N(X_0)}{D(X_0)}$ for some $X_0 \in S$. Then,

$$f(q_0) = \max_{X \in S}\{N(X) - q_0D(X)\} \geqslant N(X_0) - q_0D(X_0) = 0.$$

□

The following theorem establishes that the global maximum of (NFP) corresponds to the root q_* of the parametric function $f(q)$. Additionally, the global maximizer of (NFP) is also the global maximizer of the (NPP) corresponding to $f(q_*)$.

Theorem B.1.1 ([**37**]). *$q_* = \frac{N(X_*)}{D(X_*)} = \max_{X\in S} \frac{N(X)}{D(X)}$ if and only if $f(q_*) = N(X_*) - q_*D(X_*) = \max_{X\in S}\{N(X) - q_*D(X)\} = 0$.*

Proof. Let X_* be the global maximizer to (NFP). Then, it holds that $q_* = \frac{N(X_*)}{D(X_*)} \geqslant \frac{N(X)}{D(X)}$ for all $X \in S$ so that

$$N(X) - q_*D(X) \leqslant 0 \quad \text{for all} \quad X \in S.$$

Since $N(X_*) - q_* D(X_*) = 0$, X_* maximizes (NPP) corresponding to $f(q_*)$.

Conversely, let X_* be the global maximizer of (NPP) corresponding to $f(q_*)$, where q_* is the root of $f(q)$. Thus, $N(X_*) - q_* D(X_*) = 0$. Then, we have

$$N(X) - q_* D(X) \leqslant N(X_*) - q_* D(X_*) = 0 \quad \text{for all} \quad X \in S.$$

Therefore, $\frac{N(X)}{D(X)} \leqslant q_*$ for all $X \in S$, indicating that q_* is the global maxima of (NFP). Moreover, $q_* = \frac{N(X_*)}{D(X_*)}$, which implies that X_* is the global maximizer of (NFP). □

B.2. Algorithm

We introduce the Dinkelbach's algorithm [**37**] for solving (NFP). The algorithm generates a monotonically increasing sequence $\{q_k\}$ that approaches the root q_* of the parametric function $f(q)$. The sequence is generated by iteratively solving (NPP).

Algorithm 5 Dinkelbach's Algorithm

Input: Initial $X_0 \in S$, tolerance *tol*.

Output: Global maximizer X_* of (NFP).

```
1: for $k = 0, 1, \dots$ do
2:     Set $q_k = \frac{N(X_k)}{D(X_k)}$.
3:     Compute the global maximizer $X_{k+1}$ to $f(q_k) = \max_{X \in S}\{N(X) - q_k D(X)\}$.
4:     if $f(q_k) \leqslant \text{tol}$ then
5:         return $X_* = X_{k+1}$.
6:     end if
7: end for
```

The stopping criteria in line 4 of Algorithm 5 checks whether q_k is the root of the parametric function $f(q)$. If so, X_{k+1} computed in line 3 is the global maximizer of (NFP).

We emphasize that an accurate computation of the global maximizer of (NPP) in line 3 is crucial for the success of the Dinkelbach's algorithm. When $N(X)$ is concave and $D(X)$ is convex, concave programming methods can be employed to solve (NPP). However, it is important to note that (NPP) can be a challenging problem in general, requiring careful consideration and specialized techniques.

B.2.1. Convergence proof of $\{q_k\}$ of Algorithm 5. In the following analysis, we consider X_{k+1} to be accurately computed and regarded as "exact". We show that the sequence $\{q_k\}$ generated by Algorithm 5 converges monotonically to q_*, the root of the parametric function $f(q)$.

We first show that the sequence $\{q_k\}$ is strictly monotonically increasing, i.e., $q_{k+1} > q_k$. Here, we assume that $q_k \neq q_*$ since Algorithm 5 would have converged otherwise. Then, $f(q_k) > 0$ according to Lemma B.1.1(e) and Lemma B.1.1(d). Consequently, we obtain

$$0 < f(q_k) = N(X_{k+1}) - q_k D(X_{k+1}) = q_{k+1} D(X_{k+1}) - q_k D(X_{k+1}) = (q_{k+1} - q_k) D(x_{k+1}). \tag{B.1}$$

Since $D(X) > 0$ for all $X \in S$, this inequality shows that $q_{k+1} > q_k$.

For $q_k \neq q_*$, we have $f(q_k) > 0$. According to Lemma B.1.1(c), this indicates $q_k < q_*$. Hence, we have a strictly monotonic increasing sequence $\{q_k\}$ that is bounded above by q_*. Consequently, the sequence must converge to some limit point $\widetilde{q}$. We show that $\widetilde{q} = q_*$.

Suppose on the contrary that $\widetilde{q} \neq q_*$ so that $\widetilde{q} < q_*$ and $f(\widetilde{q}) > 0$. Since q_k monotonically increases towards $\widetilde{q}$, for any $\epsilon > 0$, there exists a large enough k such that $\widetilde{q} - q_k < \epsilon$. Specifically, we choose

$$0 < \epsilon = \frac{f(\widetilde{q})}{\widetilde{D}} \leqslant \frac{f(\widetilde{q})}{D(X_{k+1})},$$

where $\widetilde{D} := \max_{X \in S} D(X)$. Then, by $f(q_k) = (q_{k+1} - q_k) D(x_{k+1})$ from (B.1) and the fact that $f(q_k) \geqslant f(\widetilde{q})$, we have

$$q_{k+1} = q_k + \frac{f(q_k)}{D(X_{k+1})} \geqslant q_k + \frac{f(\widetilde{q})}{D(X_{k+1})} \geqslant q_k + \epsilon > \widetilde{q},$$

which contradicts the assumption that $\widetilde{q}$ is the limit point of $\{q_k\}$. Therefore, we conclude that $\widetilde{q} = q_*$.

Bibliography

[1] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, 2008.

[2] M. Z. Alaya, M. Berar, G. Gasso, and A. Rakotomamonjy, *Screening Sinkhorn algorithm for regularized optimal transport*, Advances in Neural Information Processing Systems, 32 (2019).

[3] J. Altschuler, J. Niles-Weed, and P. Rigollet, *Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration*, Advances in neural information processing systems, 30 (2017).

[4] A. Aristodemo and L. Gemignani, *Accelerating the Sinkhorn–Knopp iteration by Arnoldi-type methods*, Calcolo, 57 (2020), pp. 1–17.

[5] F. Arrigo, D. J. Higham, and F. Tudisco, *A framework for second-order eigenvector centralities and clustering coefficients*, Proceedings of the Royal Society A, 476 (2020), p. 20190724.

[6] M. Arvaneh, C. Guan, K. K. Ang, and C. Quek, *Optimizing spatial filters by minimizing within-class dissimilarities in electroencephalogram-based brain–computer interface*, IEEE transactions on neural networks and learning systems, 24 (2013), pp. 610–619.

[7] A. M. Azab, L. Mihaylova, H. Ahmadi, and M. Arvaneh, *Robust common spatial patterns estimation using dynamic time warping to improve bci systems*, in ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2019, pp. 3897–3901.

[8] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: a Practical Guide*, SIAM, 2000.

[9] Z. Bai, R.-C. Li, and D. Lu, *Optimal convergence rate of self-consistent field iteration for solving eigenvector-dependent nonlinear eigenvalue problems*, arXiv preprint arXiv:2009.09022, (2020).

[10] ———, *Sharp estimation of convergence rate for self-consistent field iteration to solve eigenvector-dependent nonlinear eigenvalue problems*, SIAM Journal on Matrix Analysis and Applications, 43 (2022), pp. 301–327.

[11] Z. Bai, D. Lu, and B. Vandereycken, *Robust rayleigh quotient minimization and nonlinear eigenvalue problems*, SIAM Journal on Scientific Computing, 40 (2018), pp. A3495–A3522.

[12] W. Bao and Q. Du, *Computing the ground state solution of bose–einstein condensates by a normalized gradient flow*, SIAM Journal on Scientific Computing, 25 (2004), pp. 1674–1697.

[13] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*, Princeton university press, 2009.

[14] B. Blankertz, M. Kawanabe, R. Tomioka, F. U. Hohlefeld, V. V. Nikulin, and K.-R. Müller, *Invariant common spatial patterns: Alleviating nonstationarities in brain-computer interfacing.*, in NIPS, 2007, pp. 113–120.

[15] B. Blankertz, K.-R. Muller, D. J. Krusienski, G. Schalk, J. R. Wolpaw, A. Schlogl, G. Pfurtscheller, J. R. Millan, M. Schroder, and N. Birbaumer, *The bci competition iii: Validating alternative approaches to actual bci problems*, IEEE transactions on neural systems and rehabilitation engineering, 14 (2006), pp. 153–159.

[16] B. Blankertz, C. Sannelli, S. Halder, E. M. Hammer, A. Kübler, K.-R. Müller, G. Curio, and T. Dickhaus, *Neurophysiological predictor of smr-based bci performance*, Neuroimage, 51 (2010), pp. 1303–1309.

[17] B. Blankertz, R. Tomioka, S. Lemm, M. Kawanabe, and K.-R. Muller, *Optimizing spatial filters for robust eeg single-trial analysis*, IEEE Signal processing magazine, 25 (2007), pp. 41–56.

[18] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre, *Manopt, a Matlab toolbox for optimization on manifolds*, Journal of Machine Learning Research, 15 (2014), pp. 1455–1459.

[19] S. Brandl and B. Blankertz, *Motor imagery under distraction—an open access bci dataset*, Frontiers in Neuroscience, 14 (2020).

[20] S. Brandl, L. Frølich, J. Höhne, K.-R. Müller, and W. Samek, *Brain–computer interfacing under distraction: an evaluation study*, Journal of neural engineering, 13 (2016), p. 056012.

[21] S. Brandl, J. Höhne, K.-R. Müller, and W. Samek, *Bringing bci into everyday life: Motor imagery in a pseudo realistic environment*, in 2015 7th International IEEE/EMBS Conference on Neural Engineering (NER), IEEE, 2015, pp. 224–227.

[22] R. A. Brualdi, S. V. Parter, and H. Schneider, *The diagonal equivalence of a nonnegative matrix to a stochastic matrix*, Journal of Mathematical Analysis and Applications, 16 (1966), pp. 31–50.

[23] C. J. Burges, *Dimension Reduction: A Guided Tour*, Now Publishers Inc, 2010.

[24] D. Cai, X. He, K. Zhou, J. Han, and H. Bao, *Locality sensitive discriminant analysis.*, in IJCAI, vol. 2007, 2007, pp. 1713–1726.

[25] Y. Cai, L.-H. Zhang, Z. Bai, and R.-C. Li, *On an eigenvector-dependent nonlinear eigenvalue problem*, SIAM Journal on Matrix Analysis and Applications, 39 (2018), pp. 1360–1382.

[26] E. Cancès, *Self-consistent field algorithms for Kohn–Sham models with fractional occupation numbers*, The Journal of Chemical Physics, 114 (2001), pp. 10616–10622.

[27] E. Cancès, G. Kemlin, and A. Levitt, *Convergence analysis of direct minimization and self-consistent iterations*, SIAM Journal on Matrix Analysis and Applications, 42 (2021), pp. 243–274.

[28] E. Cances and C. Le Bris, *On the convergence of SCF algorithms for the Hartree-Fock equations*, ESAIM: Mathematical Modelling and Numerical Analysis, 34 (2000), pp. 749–774.

[29] H. Chen, X. Dai, X. Gong, L. He, and A. Zhou, *Adaptive finite element approximations for Kohn–Sham models*, Multiscale Modeling & Simulation, 12 (2014), pp. 1828–1869.

[30] H.-T. Chen, H.-W. Chang, and T.-L. Liu, *Local discriminant embedding and its variants*, in 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol. 2, IEEE, 2005, pp. 846–853.

[31] S. H. Cheng and N. J. Higham, *A modified cholesky algorithm based on a symmetric indefinite factorization*, SIAM Journal on Matrix Analysis and Applications, 19 (1998), pp. 1097–1110.

[32] H. Cho, M. Ahn, S. Ahn, M. Kwon, and S. C. Jun, *Eeg datasets for motor imagery brain–computer interface*, GigaScience, 6 (2017), p. gix034.

[33] J. P. Cunningham and Z. Ghahramani, *Linear dimensionality reduction: Survey, insights, and generalizations*, The Journal of Machine Learning Research, 16 (2015), pp. 2859–2900.

[34] M. Cuturi, *Sinkhorn distances: Lightspeed computation of optimal transport*, Advances in neural information processing systems, 26 (2013).

[35] J. Dattorro, *Convex optimization & Euclidean distance geometry*, Lulu. com, 2010.

[36] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, 1997.

[37] W. Dinkelbach, *On nonlinear fractional programming*, Management science, 13 (1967), pp. 492–498.

[38] J. J. Dongarra, J. Du Croz, S. Hammarling, and I. S. Duff, *A set of level 3 basic linear algebra subprograms*, ACM Transactions on Mathematical Software (TOMS), 16 (1990), pp. 1–17.

[39] F. M. Dopico, M. C. Quintana, and P. V. Dooren, *Diagonal scalings for the eigenstructure of arbitrary pencils*, SIAM Journal on Matrix Analysis and Applications, 43 (2022), pp. 1213–1237.

[40] G. Dornhege, M. Krauledat, K.-R. Müller, and B. Blankertz, *13 general signal processing and machine learning tools for bci analysis*, Toward Brain-Computer Interfacing, (2007), p. 207.

[41] G. Dornhege, J. d. R. Millán, T. Hinterberger, D. J. McFarland, K.-r. Muller, et al., *Toward brain-computer interfacing*, vol. 63, Citeseer, 2007.

[42] D. Dua and C. Graff, *UCI machine learning repository*, 2017.

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

[44] Z. Fan, Y. Xu, and D. Zhang, *Local linear discriminant analysis framework using sample neighbors*, IEEE Transactions on Neural Networks, 22 (2011), pp. 1119–1132.

[45] R. Flamary, M. Cuturi, N. Courty, and A. Rakotomamonjy, *Wasserstein discriminant analysis*, Machine Learning, 107 (2018), pp. 1923–1945.

[46] I. K. Fodor, *A survey of dimension reduction techniques*, tech. rep., Lawrence Livermore National Lab., CA (US), 2002.

[47] J. Franklin and J. Lorenz, *On the scaling of multidimensional matrices*, Linear Algebra and its applications, 114 (1989), pp. 717–735.

[48] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Elsevier, 2013.

[49] G. H. Golub and C. F. Van Loan, *Matrix Computations*, JHU press, 2013.

[50] J. Gu, M. Wei, Y. Guo, and H. Wang, *Common spatial pattern with l21-norm*, Neural Processing Letters, 53 (2021), pp. 3619–3638.

[51] Y.-F. Guo, S.-J. Li, J.-Y. Yang, T.-T. Shu, and L.-D. Wu, *A generalized Foley–Sammon transform based on generalized fisher discriminant criterion and its application to face recognition*, Pattern Recognition Letters, 24 (2003), pp. 147–158.

[52] J. Han, J. Pei, and H. Tong, *Data Mining: Concepts and Techniques*, Morgan kaufmann, 2022.

[53] M. Idel, *A review of matrix scaling and Sinkhorn's normal form for matrices and positive maps*, arXiv preprint arXiv:1609.06349, (2016).

[54] E. Jarlebring, S. Kvaal, and W. Michiels, *An inverse iteration method for eigenvalue problems with eigenvector nonlinearities*, SIAM Journal on Scientific Computing, 36 (2014), pp. A1978–A2001.

[55] Y. Jia, F. Nie, and C. Zhang, *Trace ratio problem revisited*, IEEE Transactions on Neural Networks, 20 (2009), pp. 729–735.

[56] B. Kalantari and L. Khachiyan, *On the complexity of nonnegative-matrix scaling*, Linear Algebra and its applications, 240 (1996), pp. 87–103.

[57] H. Kasai, *Multi-view Wasserstein discriminant analysis with entropic regularized Wasserstein distance*, in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2020, pp. 6039–6043.

[58] T. Kato, *Perturbation theory for linear operators*, vol. 132, Springer Science & Business Media, 2013.

[59] M. Kawanabe, W. Samek, K.-R. Müller, and C. Vidaurre, *Robust common spatial filters with a maxmin approach*, Neural computation, 26 (2014), pp. 349–376.

[60] M. Kawanabe, C. Vidaurre, S. Scholler, and K.-R. Muuller, *Robust common spatial filters with a maxmin approach*, in 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2009, pp. 2470–2473.

[61] S.-J. Kim, A. Magnani, and S. Boyd, *Robust fisher discriminant analysis*, in Advances in neural information processing systems, 2006, pp. 659–666.

[62] P. A. Knight, *The Sinkhorn–Knopp algorithm: convergence and applications*, SIAM Journal on Matrix Analysis and Applications, 30 (2008), pp. 261–275.

[63] P. A. Knight and D. Ruiz, *A fast algorithm for matrix balancing*, IMA Journal of Numerical Analysis, 33 (2013), pp. 1029–1047.

[64] A. V. Knyazev, *Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method*, SIAM journal on scientific computing, 23 (2001), pp. 517–541.

[65] Z. J. Koles, *The quantitative extraction and topographic mapping of the abnormal components in the clinical eeg*, Electroencephalography and clinical Neurophysiology, 79 (1991), pp. 440–447.

[66] J. Kruithof, *Telefoonverkeersrekening*, De Ingenieur, 52 (1937), pp. 15–25.

[67] B. Kulis et al., *Metric learning: A survey*, Foundations and Trends® in Machine Learning, 5 (2013), pp. 287–364.

[68] O. Ledoit and M. Wolf, *A well-conditioned estimator for large-dimensional covariance matrices*, Journal of multivariate analysis, 88 (2004), pp. 365–411.

[69] S. Lemm, B. Blankertz, T. Dickhaus, and K.-R. Müller, *Introduction to machine learning for brain imaging*, Neuroimage, 56 (2011), pp. 387–399.

[70] D. Lemonnier and P. Van Dooren, *Balancing regular matrix pencils*, SIAM journal on matrix analysis and applications, 28 (2006), pp. 253–263.

[71] X. Li, X. Lu, and H. Wang, *Robust common spatial patterns with sparsity*, Biomedical Signal Processing and Control, 26 (2016), pp. 52–57.

[72] H. Liu, Y. Cai, Y.-L. Chen, and P. Li, *Ratio trace formulation of Wasserstein discriminant analysis*, Advances in Neural Information Processing Systems, 33 (2020).

[73] X. Liu, X. Wang, Z. Wen, and Y. Yuan, *On the convergence of the self-consistent field iteration in kohn–sham density functional theory*, SIAM Journal on Matrix Analysis and Applications, 35 (2014), pp. 546–558.

[74] X. Liu, Z. Wen, X. Wang, M. Ulbrich, and Y. Yuan, *On the analysis of the discretized Kohn–Sham density functional theory*, SIAM Journal on Numerical Analysis, 53 (2015), pp. 1758–1785.

[75] F. Lotte and C. Guan, *Regularizing common spatial patterns to improve bci designs: unified theory and new algorithms*, IEEE Transactions on biomedical Engineering, 58 (2010), pp. 355–362.

[76] O. L. Mangasarian and E. W. Wild, *Multisurface proximal support vector machine classification via generalized eigenvalues*, IEEE transactions on pattern analysis and machine intelligence, 28 (2005), pp. 69–74.

[77] R. M. Martin, *Electronic structure: basic theory and practical methods*, Cambridge university press, 2020.

[78] J. Müller-Gerking, G. Pfurtscheller, and H. Flyvbjerg, *Designing optimal spatial filters for single-trial eeg classification in a movement task*, Clinical neurophysiology, 110 (1999), pp. 787–798.

[79] T. T. Ngo, M. Bellalij, and Y. Saad, *The trace ratio optimization problem*, SIAM review, 54 (2012), pp. 545–569.

[80] F. Nie, S. Xiang, Y. Jia, C. Zhang, and S. Yan, *Trace ratio criterion for feature selection.*, in AAAI, vol. 2, 2008, pp. 671–676.

[81] F. Nie, S. Xiang, and C. Zhang, *Neighborhood minmax projections.*, in IJCAI, 2007, pp. 993–998.

[82] J. Nocedal and S. Wright, *Numerical Optimization*, Springer Science & Business Media, 2006.

[83] B. Parlett and T. Landis, *Methods for scaling to doubly stochastic form*, Linear Algebra and its Applications, 48 (1982), pp. 53–79.

[84] B. N. Parlett, *The Symmetric Eigenvalue Problem*, SIAM, 1998.

[85] B. N. Parlett and C. Reinsch, *Balancing a matrix for calculation of eigenvalues and eigenvectors*, in Handbook for Automatic Computation, Springer, 1971, pp. 315–326.

[86] O. Pele and M. Werman, *Fast and robust earth mover's distances*, in 2009 IEEE 12th international conference on computer vision, IEEE, 2009, pp. 460–467.

[87] G. Peyré, M. Cuturi, et al., *Computational optimal transport: With applications to data science*, Foundations and Trends® in Machine Learning, 11 (2019), pp. 355–607.

[88] G. Pfurtscheller and F. L. Da Silva, *Event-related eeg/meg synchronization and desynchronization: basic principles*, Clinical neurophysiology, 110 (1999), pp. 1842–1857.

[89] B. Reuderink and M. Poel, *Robustness of the common spatial patterns algorithm in the bci-pipeline*, University of Twente, Tech. Rep, (2008).

[90] D. M. Roh and Z. Bai, *A bi-level nonlinear eigenvector algorithm for wasserstein discriminant analysis*, arXiv preprint arXiv:2211.11891, (2022).

[91] Y. Rubner, L. J. Guibas, and C. Tomasi, *The earth mover's distance, multi-dimensional scaling, and color-based image retrieval*, in Proceedings of the ARPA image understanding workshop, vol. 661, 1997, p. 668.

[92] Y. Saad, J. R. Chelikowsky, and S. M. Shontz, *Numerical methods for electronic structure calculations of materials*, SIAM review, 52 (2010), pp. 3–54.

[93] W. Samek, D. Blythe, K.-R. Müller, and M. Kawanabe, *Robust spatial filtering with beta divergence*, Advances in Neural Information Processing Systems, 26 (2013).

[94] W. Samek, K.-R. Müller, M. Kawanabe, and C. Vidaurre, *Brain-computer interfacing in discriminative and stationary subspaces*, in 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2012, pp. 2873–2876.

[95] W. Samek, C. Vidaurre, K.-R. Müller, and M. Kawanabe, *Stationary common spatial patterns for brain–computer interfacing*, Journal of neural engineering, 9 (2012), p. 026013.

[96] C. Sannelli, C. Vidaurre, K.-R. Müller, and B. Blankertz, *A large scale screening study with a smr-based bci: Categorization of bci users and differences in their smr activity*, PLoS One, 14 (2019), p. e0207351.

[97] R. Sinkhorn, *A relationship between arbitrary positive matrices and doubly stochastic matrices*, The annals of mathematical statistics, 35 (1964), pp. 876–879.

[98] ——, *Diagonal equivalence to matrices with prescribed row and column sums*, The American Mathematical Monthly, 74 (1967), pp. 402–405.

[99] R. Sinkhorn and P. Knopp, *Concerning nonnegative matrices and doubly stochastic matrices*, Pacific Journal of Mathematics, 21 (1967), pp. 343–348.

[100] D. C. Sorensen, *Implicitly restarted Arnoldi/Lanczos methods for large scale eigenvalue calculations*, in Parallel Numerical Algorithms, Springer, 1997, pp. 119–165.

[101] G. W. Stewart, *Matrix Algorithms: Volume II: Eigensystems*, SIAM, 2001.

[102] ———, *A Krylov–Schur algorithm for large eigenproblems*, SIAM Journal on Matrix Analysis and Applications, 23 (2002), pp. 601–614.

[103] B. Su and G. Hua, *Order-preserving Wasserstein distance for sequence matching*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1049–1057.

[104] B. Su, J. Zhou, and Y. Wu, *Order-preserving Wasserstein discriminant analysis*, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9885–9894.

[105] A. Szabo and N. S. Ostlund, *Modern quantum chemistry: introduction to advanced electronic structure theory*, Courier Corporation, 2012.

[106] R. Tomioka and K. Aihara, *Classifying matrices with a spectral regularization*, in Proceedings of the 24th international conference on Machine learning, 2007, pp. 895–902.

[107] R. Tomioka, K. Aihara, and K.-R. Müller, *Logistic regression for single trial eeg classification*, Advances in neural information processing systems, 19 (2006).

[108] J. Townsend, N. Koep, and S. Weichwald, *Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation*, Journal of Machine Learning Research, 17 (2016), p. 1–5.

[109] F. Tudisco and D. J. Higham, *A nonlinear spectral method for core–periphery detection in networks*, SIAM Journal on Mathematics of Data Science, 1 (2019), pp. 269–292.

[110] ———, *Core-periphery detection in hypergraphs*, SIAM Journal on Mathematics of Data Science, 5 (2023), pp. 1–21.

[111] P. Upadhyaya, E. Jarlebring, and E. H. Rubensson, *A density matrix approach to the convergence of the self-consistent field iteration*, arXiv preprint arXiv:1809.02183, (2018).

[112] L. Van Der Maaten, E. Postma, J. Van den Herik, et al., *Dimensionality reduction: a comparative*, J Mach Learn Res, 10 (2009), p. 13.

[113] C. Villani, *Optimal Transport: Old and New*, vol. 338, Springer, 2009.

[114] H. Wang, Q. Tang, and W. Zheng, *L1-norm-based common spatial patterns*, IEEE Transactions on Biomedical Engineering, 59 (2011), pp. 653–662.

[115] H. Wang, S. Yan, D. Xu, X. Tang, and T. Huang, *Trace ratio vs. ratio trace for dimensionality reduction*, in 2007 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–8.

[116] R. C. Ward, *Balancing the generalized eigenvalue problem*, SIAM Journal on Scientific and Statistical Computing, 2 (1981), pp. 141–152.

[117] K. Q. Weinberger and L. K. Saul, *Distance metric learning for large margin nearest neighbor classification.*, Journal of machine learning research, 10 (2009).

[118] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, *Brain–computer interfaces for communication and control*, Clinical neurophysiology, 113 (2002), pp. 767–791.

[119] P. Xanthopoulos, M. R. Guarracino, and P. M. Pardalos, *Robust generalized eigenvalue classifier with ellipsoidal uncertainty*, Annals of Operations Research, 216 (2014), pp. 327–342.

[120] S. Yan, D. Xu, B. Zhang, and H.-J. Zhang, *Graph embedding: A general framework for dimensionality reduction*, in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, IEEE, 2005, pp. 830–837.

[121] C. Yang, W. Gao, and J. C. Meza, *On the convergence of the self-consistent field iteration for a class of nonlinear eigenvalue problems*, SIAM journal on matrix analysis and applications, 30 (2009), pp. 1773–1788.

[122] C. Yang, J. C. Meza, and L.-W. Wang, *A trust region direct constrained minimization algorithm for the kohn–sham equation*, SIAM Journal on Scientific Computing, 29 (2007), pp. 1854–1875.

[123] X. Yong, R. K. Ward, and G. E. Birch, *Robust common spatial patterns for eeg signal preprocessing*, in 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2008, pp. 2087–2090.

[124] L.-H. Zhang, L.-Z. Liao, and M. K. Ng, *Fast algorithms for the generalized Foley–Sammon discriminant analysis*, SIAM journal on matrix analysis and applications, 31 (2010), pp. 1584–1605.

[125] L.-H. Zhang and W. H. Yang, *Perturbation analysis for the trace quotient problem*, Linear and Multilinear Algebra, 61 (2013), pp. 1629–1640.

[126] L.-H. Zhang, W. H. Yang, and L.-Z. Liao, *A note on the trace quotient problem*, Optimization Letters, 8 (2014), pp. 1637–1645.

[127] T. Zhang, Y. Wang, Z. Cui, C. Zhou, B. Cui, H. Huang, and J. Yang, *Deep Wasserstein graph discriminant learning for graph classification.*, in AAAI, 2021, pp. 10914–10922.

[128] M. Zhao, Z. Zhang, and T. W. Chow, *Trace ratio criterion based generalized discriminative learning for semi-supervised dimensionality reduction*, Pattern Recognition, 45 (2012), pp. 1482–1499.