INCREMENTAL AND REGULARIZED LINEAR DISCRIMINANT ANALYSIS

WANG XIAOYAN

(M.Sc., ECNU, China)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF MATHEMATICS

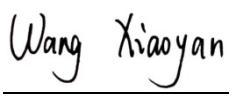
NATIONAL UNIVERSITY OF SINGAPORE

2012

DECLARATION

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.



Wang Xiaoyan

19 August 2012

Acknowledgements

First and foremost I offer my sincerest gratitude to my supervisor, Professor Chu Delin, for his support and encouragement, guidance and assistance in my studies and my research work, especially, for his patience and advice on the improvement of my skill in both of research and writing. It would not have been possible to write this doctoral thesis without the help of him. Prof. Chu Delin who has the highest degree of professionalism and integrity is a model of meticulous scholarship in my mind.

I would like to give special thanks to Professor Ching Wai-Ki (at the University of Hong Kong) and Professor Liao Li-Zhi (at Hong Kong Baptist University) for their assistance and support in my research work.

I would like to thank Department of Mathematics and National University of Singapore for providing me excellent research conditions and scholarship to complete my PhD study. I also feel grateful for the facility of Center for Computational Science and Engineering that enable us to run programmes.

I would also like to thank all my friends in Singapore for their kindness help. With special thanks to Goh Siong Thye and Zhang Xiaowei for their helpful discussion and assistance in my research work.

This thesis is dedicated to my family, for their encouragement and support.

Contents

Summary

This thesis focuses on the theory, implementation and applications of linear discriminant analysis (LDA). LDA is a well-known supervised dimensionality reduction technique, which has been applied successfully in many important applications such as pattern recognition, information retrieval, face recognition, micro-array data analysis and text classification.

The original LDA is a batch method that needs all training data to be available in advance in order to construct the transformation matrix. However, in many applications, not all data is available at the same time. In order to avoid storing the complete data it is necessary to process learning samples as soon as they become available and discard them immediately afterwards. Consequently, instead of learning data from scratch, incremental dimensionality reduction algorithm that directly updates the current transformation matrix whenever a new data is inserted, is desirable. In this thesis, an LDA-based incremental dimensionality reduction algorithm, ILDA/QR, has been developed. ILDA/QR produces exact transformation matrix as its batch version, in addition, it is very fast and always achieves comparative classification accuracy compared with ULDA algorithm and existing incremental LDA algorithms. More importantly, it can easily handle not only the case that only one new sample is inserted but also the case that a chunk

of new samples are added.

As an extension of classical LDA to deal with the undersampled problem, regularized LDA is frequently used by adding a regularized perturbation to the scatter matrix. However, the major issue of regularized LDA involved in existing methods is how to choose an appropriate regularization parameter. In this thesis, by deriving the mathematical relationship between orthogonal linear discriminant analysis (OLDA) and regularized orthogonal linear discriminant analysis (ROLDA), we find a mathematical criterion for selecting the regularization parameter in ROLDA. Unlike other regularized LDA methods, no candidate set of regularization parameter is needed in our new proposed method.

List of Tables

List of Figures

Chapter 1

Introduction

Many applications including machine learning, data mining and bioinformatics require us to deal with high dimensional data efficiently. Advances in data collection and storage capabilities during the past decades have led to an information overload in lots of applications. Researchers working in domains as diverse as engineering, astronomy, biology, remote sensing, economics, and consumer transactions, face larger and larger observations and simulations on a daily basis. Such datasets, in contrast with smaller, more traditional datasets that have been studied extensively in the past, present new challenges in data analysis. Searching for intrinsic data structure embedded in high dimensional data can give a low dimensional representation which can preserve essential information in the original data, and it is often necessary to reduce the dimension of the datasets significantly in order to achieve higher efficiency in manipulating the data. As a consequence, methods like dimensionality reduction are important.

Dimensionality reduction, which transforms the high-dimensional data into a lower-dimensional space with limited loss of information, studies methods that effectively reduce data dimensionality for efficient data processing tasks. Its general purposes are to remove irrelevant and redundant data to reduce the computational cost and avoid data over-fitting, and to improve the quality of data for efficient data-intensive processing tasks such as face recognition and data mining. Once the

high-dimensional data is transformed to a low dimensional space, some indexing techniques [110, 58, 68] can be effectively applied to facilitate efficient retrieval of data. Dimensionality reduction is an effective solution to the problem of "curse of dimensionality" [9, 35, 30, 14, 51], that is, an enormous number of samples is required to perform accurate prediction on problems with high dimensionality, this is because in high-dimensional spaces, data become extremely sparse and apart from each other. Hence, the problem of data dimensionality reduction has received broad attention, see [39, 73, 89, 14, 103, 40, 96, 49, 99, 24, 48, 6, 29, 52, 57, 30, 55, 56, 18, 65, 83, 12, 33, 35, 34, 53, 28] for instance.

Many different techniques for dimensionality reduction have been developed in the past. Among them, principle component analysis (PCA) [57] and linear discriminant analysis (LDA) [35, 30, 51] are two of the most popular linear subspace learning methods. PCA is an unsupervised learning method [47, 30], which performs dimensionality reduction by projecting the original high dimensional data onto the low dimensional linear subspace spanned by the leading eigenvectors of the data's covariance matrix. PCA deals with data in its entirety for the principal components analysis without paying any particular attention to the underlying class structure. On the other hand, LDA is a supervised learning method [51, 3], which aims to find the optimal low-dimensional representation to the original dataset by minimizing the within-class distance and maximizing the between-class distance simultaneously, thus achieving maximum class discrimination. In the sense of classification, LDA is substantially optimized than PCA. Due to the ability of PCA to shrink down the problem size, PCA is frequently used as a pre-processing technique [83, 7] before applying LDA. LDA has been applied successfully for decades in many important applications of diverse fields including pattern recognition [35, 86, 30, 14], information retrieval [33, 65], face recognition [83, 55], micro-array data analysis [6, 29], and text classification [53].

The overall objective of this thesis is to study LDA-based dimensionality reduction approaches.

1.1 Classical LDA

Given a data matrix

$$A = \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} = \begin{bmatrix} \mathcal{A}_1 & \cdots & \mathcal{A}_k \end{bmatrix} \in \mathbf{R}^{m\times n}, \quad m > n > k,$$

from a high dimensional space being grouped into k classes, where each a_i $(1 \leq i \leq n)$ is a data point in an m dimensional space and each block matrix $\mathcal{A}_i \in \mathbf{R}^{m\times n_i}$ $(1 \leq i \leq k)$ is a collection of data items in the i-th class, n_i $(1 \leq i \leq k)$ is the size of the class i and the total number of data items in data set A is $n = \sum_{i=1}^{k} n_i$. Let $\mathcal{N}_i$ denote the set of column indices that belong to the class i. The global centroid c of A and the local centroid c_i of each class $\mathcal{A}_i$ are given by

$$c = \frac{1}{n}Ae, \quad c_i = \frac{1}{n_i}\mathcal{A}_i e_i, \quad i = 1, \cdots, k,$$

respectively, where

$$e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^n, \quad e_i = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^{n_i}, \quad i = 1, \cdots, k.$$

Let

$$\begin{aligned} S_b &= \sum_{i=1}^{k} \sum_{j\in\mathcal{N}_i} (c_i - c)(c_i - c)^T = \sum_{i=1}^{k} n_i (c_i - c)(c_i - c)^T, \\ S_w &= \sum_{i=1}^{k} \sum_{j\in\mathcal{N}_i} (a_j - c_i)(a_j - c_i)^T, \\ S_t &= \sum_{j=1}^{n} (a_j - c)(a_j - c)^T. \end{aligned} \tag{1.1}$$

Here S_b, S_w and S_t are called the between-class scatter matrix, the within-class scatter matrix and the total scatter matrix, respectively. It is well known that [48]

$$S_t = S_b + S_w. \tag{1.2}$$

Denote

$$\begin{aligned} H_b &= \left[\sqrt{n_1}(c_1 - c) \quad \cdots \quad \sqrt{n_k}(c_k - c)\right] \in \mathbf{R}^{m\times k}, \\ H_w &= \left[\mathcal{A}_1 - c_1 e_1^T \quad \cdots \quad \mathcal{A}_k - c_k e_k^T\right] \in \mathbf{R}^{m\times n}, \\ H_t &= \left[a_1 - c \quad \cdots \quad a_n - c\right] = A - ce^T \in \mathbf{R}^{m\times n}, \end{aligned} \tag{1.3}$$

where H_t is called the centered data matrix. Then scatter matrices S_b, S_w and S_t can be expressed as:

$$S_b = H_b H_b^T, \quad S_w = H_w H_w^T, \quad S_t = H_t H_t^T. \tag{1.4}$$

It follows from the properties of matrix trace that

$$\operatorname{trace}(S_w) = \sum_{i=1}^{k} \sum_{j\in\mathcal{N}_i} (a_j - c_i)^T (a_j - c_i) = \sum_{i=1}^{k} \sum_{j\in\mathcal{N}_i} \|a_j - c_i\|_2^2 ,$$

and

$$\operatorname{trace}(S_b) = \sum_{i=1}^{k} n_i (c_i - c)^T (c_i - c) = \sum_{i=1}^{k} n_i \|c_i - c\|_2^2 .$$

Thus $\operatorname{trace}(S_b)$ measures the distance between the class local centroids and the global centroid, while $\operatorname{trace}(S_w)$ measures the distance between the data points and their corresponding class local centroid. Note that when data points within each class are tightly located around their local class centroid, the value of $\operatorname{trace}(S_w)$ will be small, while when the local centroids are remote from the global centroid, the value of $\operatorname{trace}(S_b)$ will be large. So the class quality can be measured by the values of $\operatorname{trace}(S_w)$ and $\operatorname{trace}(S_b)$. When $\operatorname{trace}(S_b)$ is large while $\operatorname{trace}(S_w)$ is small, the different classes will be separated well and the data points within each class will be related tightly. This leads to high class quality.

In the lower dimensional space mapped upon using the linear transformation $G \in \mathbf{R}^{m\times l}$, the between-class, within-class and total scatter matrices are of the forms

$$S_b^L = G^T S_b G, \quad S_w^L = G^T S_w G, \quad S_t^L = G^T S_t G.$$

Ideally, the optimal transformation G should maximize $\operatorname{trace}(S_b^L)$ and minimize $\operatorname{trace}(S_w^L)$ simultaneously, equivalently, maximize $\operatorname{trace}(S_b^L)$ and minimize $\operatorname{trace}(S_t^L)$

simultaneously, which leads to optimization in classical LDA for determining the optimal linear transformation G, namely the classical Fisher criterion:

$$G = \arg \max_{G \in \mathbf{R}^{m \times l}} \text{trace}((S_t^L)^{-1} S_b^L). \tag{1.5}$$

In the classical LDA [35], the above optimization problem is solved by computing all the generalized eigen-pairs

$$S_b x = \lambda S_t x, \quad \lambda \neq 0.$$

When S_t is nonsingular, it reduces to the following regular eigenvalue problem

$$S_t^{-1} S_b x = \lambda x, \quad \lambda \neq 0.$$

Thus, the solution G can be characterized explicitly through the eigen-decomposition of the matrix $S_t^{-1} S_b$. It is easy to know that $\text{rank}(S_b) = \text{rank}(H_b) \leq k - 1$, and so, the reduced dimension by the classical LDA is at most $k - 1$.

1.2 Generalized LDA

Classical LDA has a critical drawback, that is, the total scatter matrix S_t must be nonsingular. However, when the data points are from a very high-dimensional space and thus usually the number of the data samples is much smaller than the data dimension, i.e., $m > n$, the total scatter matrix S_t is singular. This is known as the undersampled problem [35] and it is also commonly called the small sampled size problem. Thus, we cannot apply the classical LDA to undersampled problems directly.

To make LDA applicable for undersampled problems, various extensions of the classical LDA can be found in the literature. These extensions can be roughly categorized into three categories. The first approach, known as the two-stage LDA, is to apply an intermediate dimensionality reduction stage to reduce the dimension of the original data before classical LDA is applied, such as, PCA+LDA [83, 7],

LSI+LDA [87, 50], and PLS+LDA [82]. Two-stage LDA implements LDA by projecting the data into a subspace, whereby it is also known as subspace LDA [107]. Although this approach is simple, the intermediate dimensionality reduction stage may remove some important information.

The second approach applies the pseudoinverse [38] to avoid the singularity problem. Orthogonal LDA (OLDA)[96, 20, 71] belongs to this group, wherein the optimal transformation matrix G has orthonormal columns, i.e., $G^TG = I$. Uncorrelated LDA (ULDA) [98, 101, 94, 22] is another popular approach, the features in the reduced space of which are uncorrelated, accomplished by adding a constraint, $G^TS_tG = I$, to LDA. In addition, null space LDA (NLDA)[18, 52, 21] performs LDA by maximizing the between-class distance in the null space of the within-class scatter matrix.

Later on, Wang et al. [90, 91] pointed out that both subspace LDA and NLDA discard some useful discriminative information and encounter the overfitting problem. In PCA+LDA, the components with small eigenvalues are removed by the PCA pre-processing. When the PCA subspace dimension is relatively high, the constructed LDA classifier is often biased and unstable. With the existence of noise, the null space of the within-class scatter matrix becomes small when the data sample size is large, hence much discriminative information outside this null space will be lost. The constructed classifier in NLDA may also be over tuned to the training set.

The third approach to bear on the undersampled problem is the regularized LDA [34, 24, 39, 104, 105]. The basic idea of the regularized LDA is to add a multiple of identity matrix λI to the total scatter matrix S_t, where $\lambda > 0$ is the regularization parameter, so the classical LDA methodologies can be applied.

The main disadvantage of the regularized LDA is that the optimal regularization parameter λ is difficult to determine [24, 99], since if λ is large then we lose information on the scatter matrix S_t, while if it is too small the regularization may not be sufficiently effective. So the main task of the regularized LDA is to

choose an appropriate regularization parameter. In the existing regularized LDA, a candidate set of the regularization parameter is given, then the cross-validation method [31, 67], which is a classification method, is used to choose an "optimal" regularization parameter from the given candidate set. The procedure of selecting an optimal value for a parameter such as λ is called as model selection [45]. Obviously, the limitation of such regularized LDA methods is that it is not clear how to choose an appropriate candidate set. The model selection problem in the regularized LDA has been addressed in a number of work, see [34, 39, 104, 105, 54]. For the regularized LDA, there are an infinite number of choices for the value of the regularization parameter, to achieve satisfactory performance in practice, a large set of candidate values are usually used. Recently it has been shown in [54] that the matrix computations involved in the regularized LDA can be simplified so that the cross-validation procedure can be performed efficiently. However, there is still no solid mathematical theory for selecting an appropriate regularization parameter.

One of the purposes of this thesis is to fill this gap. As we show in Chapter 5, there is a close mathematical relationship between solutions of OLDA:

$$G = \arg \max_{G \in \mathbf{R}^{m \times l},\ G^T G = I} \operatorname{trace}((G^T S_t G)^{(+)} G^T S_b G)^1 \tag{1.6}$$

and solutions of regularized OLDA (ROLDA):

$$G^\lambda = \arg \max_{G \in \mathbf{R}^{m \times l},\ G^T G = I} \operatorname{trace}((G^T (S_t + \lambda I) G)^{-1} G^T S_b G), \tag{1.7}$$

with $\lambda > 0$. Actually, given a regularization parameter, the distance between solutions of OLDA and ROLDA is bounded; and conversely, given the distance between solutions of OLDA and ROLDA (also, it is the tolerance of the approximation error when using the solution of ROLDA to approximate the solution of OLDA), there is a bound of the regularization parameter. Note that for any data items x and y,

$$\left| \left\| G^\lambda x - G^\lambda y \right\|_F - \left\| Gx - Gy \right\|_F \right| \leq \left\| G^\lambda - G \right\|_F (\|x\|_F + \|y\|_F),$$

[1]For the definition and properties of Moore-Penrose inverse and trace operator, please refer to Appendix A.

which indicates that $\left\|G^{\lambda}x - G^{\lambda}y\right\|_F$ is a good approximation of $\|Gx - Gy\|_F$ provided that G^{λ} is close to G. Hence, it is expected that if the regularization parameter is selected by calibrating the solution of ROLDA to the solution of OLDA, ROLDA can achieve a satisfactory classification performance similar to OLDA, which is verified by our numerical results in Chapter 5. This observation leads to a mathematical way to select the regularization parameter in ROLDA: with a given tolerance of the approximation error defined by users, the regularization parameter is set as its upper bound.

In this thesis, all solutions of OLDA (1.6) and ROLDA (1.7) are characterized first and then the intrinsic relationship between OLDA and ROLDA is exploited. More importantly, by means of this relationship we find a mathematical criterion for selecting the regularization parameter in ROLDA and consequently we develop a new regularized orthogonal linear discriminant analysis method, in which no candidate set of regularization parameter is needed. The effectiveness of our proposed ROLDA is demonstrated by real-world data sets.

Regularization is a powerful tool in remarkably wide areas of many disciplines, for example, the kernel machines area, [78, 44, 13, 43] and the references therein. The methodology developed in the present work is expected to have impact on the further study for regularization problems in these areas.

1.3 Incremental LDA

The original LDA is a batch algorithm, which requires that the data must be available in advance and be given once altogether. However, in most real-world applications, data are incrementally received from various data sources presented as a data stream. For streaming data, the input data to be learned are not available all at once, but rather arrive as continuous data sequences. The approach for batch LDA to deal with streaming data is to collect data whenever a new one is presented, and learn all the data from scratch. However, it is obvious that large memory and

high computational cost are involved in this batch learning, because the system would need to maintain a huge memory to store the data either previously learned, or newly presented. For large and high-dimensional datasets, the lack of available space becomes a critical issue.

To solve the above problem, incremental learning is more suitable. Learning from new data without forgetting prior knowledge is known as incremental learning. In this learning scheme, a system retains the knowledge acquired in the past without keeping a large number of training samples and incorporates new data items as they become available. Compared to non-incremental or batch learning, incremental learning has the advantages of being more widely adaptive and reactive. Such as, it can be applied to dynamic scenario where input data come only in sequence and a timely updating model is crucial for performance. Different from batch learning, incremental learning deals with theory revision when a new data is inserted dynamically instead of theory creation from the beginning. In this regard, incremental methods require less memory demand and computational cost than batch methods.

Several methods for incremental LDA learning have been proposed in the past. In [102], the IDR/QR algorithm that applies regularized LDA in a projected subspace spanned by the class means is developed. As the dimension of the subspace which is equal to the number of classes k is low, IDR/QR algorithm is very fast, whereas, the potential limitation of this algorithm is that much information is lost in the first projection as well as minor components are discarded in the updating process. Moreover, an appropriate regularization parameter should be given in advance. In [62, 63], a new proposed incremental LDA algorithm (we call it ILDA/SSS in this thesis) uses the concept of sufficient spanning set approximation to update the between-class scatter matrix S_b and total scatter matrix S_t, where the principal eigenvectors and eigenvalues of both matrices are kept and updated and minor components are removed in each updating. The big issue of ILDA/SSS is that three eigenvalue thresholds for respectively determining the

principal components of between-class scatter matrix, total scatter matrix and the optimal transformation matrix should be given in advance. As illustrated in [69], ILDA/SSS suffers from a problem, that is, it is difficult to determine to which degree the performance should be traded off for computational efficiency. If too many minor components are discarded, the performance will deteriorate, otherwise the efficiency will be impaired. Moreover, the performance is sensitive to the setting of the approximation parameter, while tuning the parameters is not easy.

Unlike IDR/QR and ILDA/SSS, two recently proposed incremental algorithms, LS-ILDA [69] and ICLDA [70], perform exact incremental updating whenever a new data sample is inserted. LS-ILDA is based on the idea of LS-LDA [97] which gives the least square solution to LDA. The core step of LS-ILDA is the updating of the pseudo-inverse of the centered data matrix H_t. ICLDA is another incremental approach by incrementally implementing CLDA [95] exactly. However, the dimension of the reduced space of ICLDA which is equal to the rank of the total scatter matrix is high. There are too many items that need to be updated when a new sample is inserted as well as to be stored for subsequent learning, which increase both the computational cost and the memory cost.

In this thesis, a new, efficient and simple implementation of LDA, called LDA/QR, which is eigen-decomposition-free and SVD-free, is first proposed in Chapter 3. Due to the simplicity of LDA/QR, we design a novel incremental dimensionality reduction algorithm, called ILDA/QR. It can easily handle not only the case that only one new sample is inserted but also the case that a chunk of new samples are added. Crucially, ILDA/QR is proved to accurately incrementally update the discriminant vectors of LDA/QR instead of recomputing the LDA/QR again. More importantly, only some matrix-vector multiplications and one matrix addition are involved in sequential ILDA/QR, while only economic QR factorization of a small size matrix and matrix multiplications are involved in chunk ILDA/QR, thus, our new proposed LDA-based incremental algorithm is very fast. Extensive experiments show that ILDA/QR performs favorably in classification

and execution time, in comparison with some existing LDA-based incremental algorithms.

The relationship of LDA/QR to ULDA [98, 101, 94, 22] is also analyzed. Numerical experiments conducted on a variety of real-world datasets show that LDA/QR is competitive with ULDA/QR [22] in terms of classification accuracy while less execution time is needed.

1.4 Outline of the Thesis

In addition to the Introduction, this thesis is organized into five chapters and two appendices as follows. Four existing incremental LDA algorithms are summarized in Chapter 2. Our new proposed fast LDA algorithm LDA/QR and an efficient LDA-based incremental algorithm ILDA/QR are introduced in Chapter 3. Chapter 4 reviews three existing regularized LDA methods. Chapter 5 shows a new implementation of OLDA and a new regularized OLDA is proposed. We finish the thesis with a conclusion and an overview of possible future work in Chapter 6. Some properties of Moore-Penrose inverse and trace operator are shown in Appendix A. Furthermore, the computational complexity of some matrix computation used in this thesis is given in Appendix B and the description of datasets used in our experiments is summarized in Appendix C.

Chapter 2

Existing Incremental LDA

In this chapter, we briefly review four existing incremental LDA algorithms: incremental dimension reduction via QR decomposition (IDR/QR) [102], incremental LDA using sufficient spanning set approximation (ILDA/SSS) [62, 63], least square incremental LDA (LS-ILDA) [69] and a recently proposed incremental complete LDA (ICLDA) [70]. Computational complexity of each algorithm is also summarized. In the incremental learning, updated version of any variable X after the insertion of new samples is denoted by $\tilde{X}$.

2.1 Incremental Dimension Reduction via QR Decomposition (IDR/QR)

IDR/QR [102] is a very efficient dimensionality reduction algorithm proposed by Ye et al. The batch version of IDR/QR mainly consists of two stages. The first stage is projecting scatter matrices into the subspace, in which the between class difference are maximized, precisely, the first stage of IDR/QR aims to solve the following optimization problem:

$$G = \arg \max_{G^T G = I} \text{trace}(G^T S_b G),$$

by computing the economic QR factorization of local centroid $C = \begin{bmatrix} c_1 & \cdots & c_k \end{bmatrix} \in \mathbf{R}^{m\times k}$, $C = \mathcal{Q}\mathcal{R}$, where $\mathcal{Q} \in \mathbf{R}^{m\times k}$ is column orthogonal and $\mathcal{R} \in \mathbf{R}^{k\times k}$ is upper triangular. Secondly, apply regularized LDA on the reduced scatter matrices resulting from the first stage, that is, the second stage of IDR/QR is achieved by solving the eigenvalue problem of

$$(W + \mu I)^{-1} B,$$

where $W \in \mathbf{R}^{k\times k}$ and $B \in \mathbf{R}^{k\times k}$ are scatter matrices projected in the reduced space, i.e., $W = \mathcal{Q}^T S_w \mathcal{Q}$, $B = \mathcal{Q}^T S_b \mathcal{Q}$.

The algorithm for batch IDR/QR is shown as follows:

Algorithm 2.1. *(Batch IDR/QR)*

Input: *Data matrix $A \in \mathbf{R}^{m\times n}$ with cluster label, regularization parameter μ.*

Output: *Transformation matrix $G \in \mathbf{R}^{m\times k}$.*

Step 1. *Construct centroid matrix $C \in \mathbf{R}^{m\times k}$, $H_w \in \mathbf{R}^{m\times n}$, and $H_b \in \mathbf{R}^{m\times k}$.*

Step 2. *Compute economic QR factorization of C as $C = \mathcal{Q}\mathcal{R}$, where $\mathcal{Q} \in \mathbf{R}^{m\times k}$ and $\mathcal{R} \in \mathbf{R}^{k\times k}$.*

Step 3. *Compute $Z = H_w^T \mathcal{Q}$, $Y = H_b^T \mathcal{Q}$.*

Step 4. *Compute reduced scatter matrices $W = Z^T Z$ and $B = Y^T Y$.*

Step 5. *Compute eigenvectors X of $(W + \mu I)^{-1} B$.*

Step 6. $G = \mathcal{Q}X$.

Based on this batch algorithm, the incremental IDR/QR proceeds in two steps: QR-updating of the centroid matrix C and updating of the reduced scatter matrices W and B with each new data being inserted. If the new inserted data belongs to an existing class, rank-one updating and updating of adding a row are involved in the QR-updating step. If the new inserted data belongs to a new class, QR-updating of centroid matrix C is accomplished by the Gram-Schmidt procedure [38]. In both cases, the authors made an approximation for the updating of reduced

scatter matrix W. Thus, IDR/QR is an approximated incremental algorithm. The implementation of incremental IDR/QR for two distinct cases are shown in the following two algorithms:

Algorithm 2.2. *(IDR/QR: Updating Existing Class)*

Input: *Centroid matrix* $C = [c_1 \quad \cdots \quad c_k]$, *its QR factorization* $C = \mathcal{Q}\mathcal{R}$, *matrix* W, *size of each class* $N = [n_1 \quad \cdots \quad n_k]$, *new data* x *with class label* $\ell \leq k$.

Output: *Updated* $\tilde{W}$, $\tilde{B}$, $\tilde{N}$, $\tilde{C}$ *and its QR factorization* $\tilde{C} = \tilde{\mathcal{Q}}\tilde{\mathcal{R}}$, *and transformation matrix* $\tilde{G}$.

Step 1. $\tilde{n}_j = n_j$ $(j \neq \ell)$, $\tilde{n}_\ell = \tilde{n}_\ell + 1$, $f = \frac{x - c_\ell}{\tilde{n}_\ell}$, $\tilde{n} = n + 1$.

Step 2. $\tilde{c}_j = c_j$ $(j \neq \ell)$, $\tilde{c}_\ell = c_\ell + f$.

Step 3. $f_1 = \mathcal{Q}^T f$, $f_2 = f - \mathcal{Q} f_1$, $g = [0 \quad \cdots \quad 1 \quad \cdots \quad 0]^T$.

Step 4. *Do rank one QR-updating*

$$\mathcal{Q}(\mathcal{R} + f_1 g^T) = \mathcal{Q}_1 \mathcal{R}_1, \quad \mathcal{Q}_1 \in \mathbf{R}^{m \times k}, \quad \mathcal{R}_1 \in \mathbf{R}^{k \times k}.$$

Step 5. *If* $\|f_2\|_2 = 0$, $\tilde{\mathcal{Q}} = \mathcal{Q}_1$, $\tilde{\mathcal{R}} = \mathcal{R}_1$,
If $\|f_2\|_2 \neq 0$, $q = f_2 / \|f_2\|_2$, *do QR-updating*

$$[\mathcal{Q}_1 \quad q] \begin{bmatrix} \mathcal{R}_1 \\ \|f_2\|_2 g^T \end{bmatrix} = \tilde{\mathcal{Q}}\tilde{\mathcal{R}}.$$

Step 6. $u = x - c_\ell$, $v = \tilde{c}_\ell - c_\ell$.

Step 7. $\tilde{u} = \tilde{\mathcal{Q}}^T u$, $\tilde{v} = \tilde{\mathcal{Q}}^T v$.

Step 8. $\tilde{W} = W + (\tilde{u} - \tilde{v})(\tilde{u} - \tilde{v})^T + n_\ell \tilde{v}\tilde{v}^T$.

Step 9. $h = \left[\sqrt{\tilde{n}_1} \quad \cdots \quad \sqrt{\tilde{n}_k}\right]^T$, $D = \operatorname{diag}\left(\sqrt{\tilde{n}_1} \quad \cdots \quad \sqrt{\tilde{n}_k}\right)$.

Step 10. $r = [\tilde{n}_1 \quad \cdots \quad \tilde{n}_k]^T$, $\tilde{r} = \frac{1}{\tilde{n}} \tilde{\mathcal{R}} r$.

Step 11. $\tilde{B} = (\tilde{\mathcal{R}} D - \tilde{r} h^T)(\tilde{\mathcal{R}} D - \tilde{r} h^T)^T$.

Step 12. *Compute eigenvectors* $\tilde{X}$ *of* $(\tilde{W} + \mu I)^{-1} \tilde{B}$.

Step 13. $\tilde{G} = \tilde{\mathcal{Q}} \tilde{X}$.

Algorithm 2.3. *(IDR/QR: Updating New Class)*

Input: *Centroid matrix* $C = [c_1 \quad \cdots \quad c_k]$, *its QR factorization* $C = \mathcal{QR}$, *matrix* W, *size of each class* $N = [n_1 \quad \cdots \quad n_k]$, *new data* x *with class label* $\ell \leq k$.

Output: *Updated* $\tilde{W}$, $\tilde{B}$, $\tilde{N}$, $\tilde{C}$ *and its QR factorization* $\tilde{C} = \tilde{\mathcal{Q}}\tilde{\mathcal{R}}$, *and transformation matrix* $\tilde{G}$.

Step 1. $\tilde{n}_j = n_j$ $(1 \leq j \leq k)$, $\tilde{n}_{k+1} = 1$, $\tilde{n} = n + 1$.

Step 2. *Do QR-updating of* $\tilde{C} = [C \quad x]$ *as* $\tilde{C} = \tilde{\mathcal{Q}}\tilde{\mathcal{R}}$.

Step 3. *Update*

$$\tilde{W} = \begin{bmatrix} W & 0 \\ 0 & 0 \end{bmatrix}.$$

Steps 4-8. *The same as Steps 9-13 of Algorithm 2.2.*

The incremental IDR/QR algorithm for the case that new samples are inserted in a chunk is shown as follows:

Algorithm 2.4. *(Chunk IDR/QR)*

Input: *Centroid matrix* $C = [c_1 \quad \cdots \quad c_k]$, *its QR factorization* $C = \mathcal{QR}$, *matrix* W, *size of each class* $N = [n_1 \quad \cdots \quad n_k]$, *new data* $\mathcal{X} = [x_1 \quad \cdots \quad x_s]$ *with class labels* ℓ_i $(i = 1, \cdots, s)$.

Output: *Updated* $\tilde{W}$, $\tilde{B}$, $\tilde{N}$, $\tilde{C}$ *and its QR factorization* $\tilde{C} = \tilde{\mathcal{Q}}\tilde{\mathcal{R}}$, *and transformation matrix* $\tilde{G}$.

Step 1. *For each data item* x_i $(i = 1, \cdots, r)$, *determine whether it is from an existing class or a new class.*

Step 2. *If* x_i *is from an existing class, update* $\tilde{C} = \tilde{Q}\tilde{R}$, $\tilde{W}$ *and* $\tilde{N}$ *following Steps 1-8 of Algorithm 2.2.*

Step 3. *If* x_i *is from a new class, update* $\tilde{C} = \tilde{Q}\tilde{R}$, $\tilde{W}$ *and* $\tilde{N}$ *following Steps 1-3 of Algorithm 2.3.*

Steps 4. *After all data items have been learned, update* $\tilde{B}$ *and compute* $\tilde{G}$ *following Steps 9-13 of Algorithm 2.2.*

By using the computational cost of some matrix computation given in Appendix B, we summarize in Table 2.1 the computational complexity of Algorithms 2.2 and 2.3 for a single insertion. Note that the updating algorithms for IDR/QR shown above are different from those given in [102]. Specifically, Algorithms 2.2 and 2.3 include the computation of transformation matrix G while this step is not enclosed in [102]. Thus, the computational complexity summarized in Table 2.1 also includes the computation of G. It can be observed from Table 2.1 that incremental IDR/QR is very fast as its main cost for a single insertion is $2mk^2 + 26mk + 91/3k^3$ and $2mk^2 + 26mks + 91/3k^3$ for the insertion of a chunk of s samples. However, since IDR/QR performs LDA in the range space of local centroid C and the reduced scatter matrix W is updated by its approximation in each updating, IDR/QR may suffer from the low classification problem.

Table 2.1: Computational complexity (flops) of algorithm IDR/QR

The computational cost of incremental IDR/QR (Updating Existing Class)	
Steps 1-3:	$4mk$
Step 4:	$12mk + 6k^2$
Step 5:	$6mk + 3k^2$
Steps 6-8:	$4mk + 4k^2$
Steps 9-11:	$6k^2$
Steps 12-13:	$2mk^2 + 91/3k^3$

The computational cost of incremental IDR/QR (Updating New Class)	
Steps 1-3:	$4mk$
Steps 4-8:	$2mk^2 + 91/3k^3$

2.2 Incremental LDA using Sufficient Spanning Set (ILDA/SSS)

In this section, we briefly review the main idea of ILDA/SSS summarized from Kim et. al. [62, 63] and the code [61] of algorithm ILDA/SSS provided by Kim. The principal discriminant of batch ILDA/SSS is obtained by maximizing between-class distance in the range space of total scatter matrix S_t.

Let the singular value decomposition of the total scatter matrix S_t and the between-class scatter matrix S_b be

$$S_t = U^T \Lambda U, \quad S_b = V^T \Gamma V,$$

respectively, where $U,\ V \in \mathbf{R}^{m\times m}$ are orthogonal and $\Lambda,\ \Gamma \in \mathbf{R}^{m\times m}$ are diagonal. Denote $\Lambda_1 \in \mathbf{R}^{\tau_1\times\tau_1}$ and $\Gamma_1 \in \mathbf{R}^{\tau_2\times\tau_2}$ as the principal eigenvalues of S_t and S_b, respectively, determined by an eigenvalue threshold, and $U_1 \in \mathbf{R}^{m\times\tau_1}$ and $V_1 \in \mathbf{R}^{m\times\tau_2}$as the corresponding principal eigenvectors. Let $Z = U_1\Lambda_1^{-1/2}$, then the optimization problem of batch ILDA/SSS is to find the components that maximize trace$(Z^T S_b Z)$. To reduce the computational cost of computing the principal components, sufficient spanning set approximation is introduced. Firstly, compute the economic QR factorization of $Z^T V_1$ as

$$Z^T V_1 = ST,$$

where $S \in \mathbf{R}^{\tau_1\times\tau_2}$ is column orthogonal and $T \in \mathbf{R}^{\tau_2\times\tau_2}$ is upper triangular. Then the optimization problem of batch ILDA/SSS is reduced to solve

$$\mathcal{G} = \arg \max_{\mathcal{G}\in\mathbf{R}^{\tau_2\times\tau},\ \mathcal{G}^T\mathcal{G}=I} \text{trace}(\mathcal{G}^T S^T Z^T V_1 \Gamma_1 V_1^T Z S \mathcal{G}),$$

where the scalability of $\mathcal{G}$, τ, is determined by an threshold, and the final optimal transformation is given by $G = U_1\Lambda_1^{-1/2}\mathcal{G}$.

Kim et. al. did not give the batch algorithm of ILDA/SSS in [62, 63], after omitting some minor processes like the QR factorization thresholding, etc, we summarize the algorithm from the website [61] provided by Kim as follows:

Algorithm 2.5. *(Batch ILDA/SSS)*

Input: *Data matrix* $A \in \mathbf{R}^{m\times n}$ *with class label, eigenvalue threshold.*

Output: *Transformation matrix* $G \in \mathbf{R}^{m\times\tau}$.

Step 1. *Construct local centroid* $C = [c_1 \ \cdots \ c_k] \in \mathbf{R}^{m\times k}$, *global centroid* $c \in \mathbf{R}^m$, $H_t \in \mathbf{R}^{m\times n}$ *and* $H_b \in \mathbf{R}^{m\times k}$.

Step 2. *Compute eigenvalue decomposition of* $H_t^T H_t$ *as* $H_t^T H_t = U\Lambda U^T$, *where* $U,\ \Lambda \in \mathbf{R}^{n\times n}$. *Denote* $U_1 \in \mathbf{R}^{n\times\tau_1}$ *and* $\Lambda_1 \in \mathbf{R}^{\tau_1\times\tau_1}$ *as the principal components of* U *and* Λ, *respectively, determined by eigenvalue threshold.*

Step 3. $U_1 := H_t U_1 \Lambda_1^{-1/2}$.

Step 4. *Compute eigenvalue decomposition of* $H_b^T H_b$ *as* $H_b^T H_b = V\Gamma V^T$, *where* $V,\ \Gamma \in \mathbf{R}^{k\times k}$. *Denote* $V_1 \in \mathbf{R}^{k\times\tau_2}$ *and* $\Gamma_1 \in \mathbf{R}^{\tau_2\times\tau_2}$ *as the principal components of* V *and* Γ, *respectively, determined by eigenvalue threshold.*

Step 5. $V_1 := H_b V_1 \Gamma_1^{-1/2}$.

Step 6. *Compute* $Z = U_1\Lambda_1^{-1/2}$ *and economic QR factorization* $Z^T V_1 = ST$, *where* $S \in \mathbf{R}^{\tau_1\times\tau_2}$, $T \in \mathbf{R}^{\tau_2\times\tau_2}$.

Step 7. *Compute the eigenvalue decomposition* $S^T Z^T V_1 \Gamma_1 V_1^T Z S = H\Sigma H^T$, *where* $H \in \mathbf{R}^{\tau_2\times\tau_2}$ *and* $\Sigma \in \mathbf{R}^{\tau_2\times\tau_2}$. *Denote* $H_1 \in \mathbf{R}^{\tau_2\times\tau}$ *as the principal components of* H *determined by eigenvalue threshold.*

Step 8. $G = ZSH_1$.

Based on this batch algorithm, the incremental ILDA/SSS mainly contains three parts: update the eigenvalue decomposition of total scatter matrix S_t, update the eigenvalue decomposition of between-class scatter matrix S_b and update the discriminant components G. Like the batch version, the concept of sufficient spanning set approximation is employed in each of these three updating parts.

For convenience, in the incremental learning, we denote $D \in \mathbf{R}^{\tau_2\times k}$ as

$$D = \begin{bmatrix} d_1 & \cdots & d_k \end{bmatrix} := V_1^T \begin{bmatrix} c_1 - c & \cdots & c_k - c \end{bmatrix},$$

where the columns of $V_1 \in \mathbf{R}^{m\times\tau_2}$ are the principal eigenvectors of between-scatter matrix S_b, c_i $(i = 1, \cdots, k)$ are class local centroids and c is the global centroid.

Kim et. al. only provided a brief structure of incremental ILDA/SSS algorithm in [62, 63], like the above batch algorithm, we summarize the MATLAB code in [61] by removing some minor processes in the following for the case that data sample is inserted one at a time.

Algorithm 2.6. *(ILDA/SSS: Updating Existing Class)*

Input: *Principal eigenvectors and eigenvalues, U_1, Λ_1. Γ_1 and V_1, global centroid c and $D = [d_1 \ \cdots \ d_k]$, total class number k, size of each class $N = [n_1 \ \cdots \ n_k]$, number of total samples n, new data x with class label $1 \le \ell \le k$, eigenvalue threshold.*

Output: *Updated $\tilde{U}_1$, $\tilde{\Lambda}_1$, $\tilde{V}_1$, $\tilde{\Gamma}_1$, $\tilde{c}$, $\tilde{D}$, $\tilde{k}$, $\tilde{N}$, $\tilde{n}$ and transformation matrix $\tilde{G}$.*

Step 1. $\tilde{c} = \frac{n}{n+1}c + \frac{1}{n+1}x$, $\tilde{n} = n+1$, $\tilde{k} = k$, $\tilde{n}_i = n_i$ $(i \neq \ell)$ *and* $\tilde{n}_\ell = n_\ell + 1$.

Step 2. $r = c - x$.

Step 3. $r_1 = U_1^T r$, $u = \frac{r - U_1 r_1}{\|r - U_1 r_1\|_2}$, $\alpha = \|r - U_1 r_1\|_2^2$.

Step 4. *Compute the eigenvalue decomposition*

$$\begin{bmatrix} \Lambda_1 & \\ & 0 \end{bmatrix} + \frac{n}{n+1} \begin{bmatrix} r_1 r_1^T & \alpha r_1 \\ \alpha r_1^T & \alpha^2 \end{bmatrix} = \Phi \tilde{\Lambda} \Phi^T,$$

where $\Phi \in \mathbf{R}^{(\tau_1+1)\times(\tau_1+1)}$, $\tilde{\Lambda} \in \mathbf{R}^{(\tau_1+1)\times(\tau_1+1)}$. Denote $\Phi_1 \in \mathbf{R}^{(\tau_1+1)\times\tilde{\tau}_1}$ and $\tilde{\Lambda}_1 \in \mathbf{R}^{\tilde{\tau}_1\times\tilde{\tau}_1}$ as the principal components of Φ and $\tilde{\Lambda}$, respectively, determined by the eigenvalue threshold.

Step 5. *Update* $\tilde{U}_1 = [U_1 \quad u]\,\Phi_1$.

Step 6. $r_2 = V_1^T r$, $v = \frac{r - V_1 r_2}{\|r - V_1 r_2\|_2}$, $\beta = \|r - V_1 r_2\|_2^2$, $r_3 = V_1^T(V_1 c_\ell + r)$, $\delta = v^T(V_1 c_\ell + r)$.

Step 7. *Compute the eigenvalue decomposition*

$$\begin{bmatrix} \Gamma_1 & \\ & 0 \end{bmatrix} + \frac{n}{n+1} \begin{bmatrix} r_2 r_2^T & \beta r_2 \\ \beta r_2^T & \beta^2 \end{bmatrix} - \frac{n_\ell}{n_\ell + 1} \begin{bmatrix} r_3 r_3^T & \delta r_3 \\ \delta r_3^T & \delta^2 \end{bmatrix} = \Psi \tilde{\Gamma} \Psi^T,$$

where $\Psi \in \mathbf{R}^{(\tau_2+1)\times(\tau_2+1)}$, $\tilde{\Gamma} \in \mathbf{R}^{(\tau_2+1)\times(\tau_2+1)}$. Denote $\Psi_1 \in \mathbf{R}^{(\tau_2+1)\times\tilde{\tau}_2}$ and $\tilde{\Gamma}_1 \in \mathbf{R}^{\tilde{\tau}_2\times\tilde{\tau}_2}$ as the principal components of Ψ and $\tilde{\Gamma}$, respectively, determined by the eigenvalue threshold.

Step 8. *Update* $\tilde{V}_1 = [V_1 \quad v]\,\Psi_1$.

Step 9. *Update* $\tilde{D}$ *as*

$$\tilde{d}_i = \begin{cases} \tilde{V}_1^T(V_1 d_i + c - \tilde{c}), & \text{if } i \neq \ell, \\ \tilde{V}_1^T(\frac{n_i(V_1 d_i + c) + x}{n_i + 1} - \tilde{c}), & \text{if } i = \ell. \end{cases}$$

Step 10. *Compute* $\tilde{Z} = \tilde{U}_1 \tilde{\Lambda}_1^{-1/2}$ *and economic QR factorization*

$$\tilde{Z}^T \tilde{V}_1 = \tilde{S}\tilde{T},$$

where $\tilde{S} \in \mathbf{R}^{\tilde{r}_1 \times \tilde{r}_2}$, $\tilde{T} \in \mathbf{R}^{\tilde{r}_2 \times \tilde{r}_2}$.

Step 11. *Compute the eigenvalue decomposition*

$$\tilde{S}^T \tilde{Z}^T \tilde{V}_1 \Gamma_1 \tilde{V}_1^T \tilde{Z} \tilde{S} = \tilde{H}\tilde{\Sigma}\tilde{H}^T,$$

where $\tilde{H} \in \mathbf{R}^{\tilde{r}_2 \times \tilde{r}_2}$ *and* $\tilde{\Sigma} \in \mathbf{R}^{\tilde{r}_2 \times \tilde{r}_2}$. *Denote* $\tilde{H}_1 \in \mathbf{R}^{\tilde{r}_2 \times \tilde{r}}$ *as the principal components of* $\tilde{H}$ *determined by eigenvalue threshold.*

Step 12. $\tilde{G} = \tilde{Z}\tilde{S}\tilde{H}_1$.

Algorithm 2.7. *(ILDA/SSS: Updating New Class)*

Input: *Principal eigenvectors and eigenvalues,* U_1, Λ_1. Γ_1 *and* V_1. *Global centroid* c *and* $D = [d_1 \ \cdots \ d_k]$, *total class number* k, *size of each class* $N = [n_1 \ \cdots \ n_k]$, *number of total samples* n, *new data* x *with class label* $\ell > k$, *eigenvalue threshold.*

Output: *Updated* $\tilde{U}_1$, $\tilde{\Lambda}_1$, $\tilde{V}_1$, $\tilde{\Gamma}_1$, $\tilde{c}$, $\tilde{D}$, $\tilde{k}$, $\tilde{N}$, $\tilde{n}$ *and transformation matrix* $\tilde{G}$.

Step 1. $\tilde{c} = \frac{n}{n+1}c + \frac{1}{n+1}x$, $\tilde{n} = n + 1$, $\tilde{k} = k + 1$, $\tilde{n}_i = n_i$ $(i \leq k)$ *and* $\tilde{n}_{k+1} = 1$.

Steps 2-5. *The same as Steps 2-5 of Algorithm 2.6.*

Step 6. $r_2 = V_1^T r$, $v = \frac{r - V_1(V_1^T r)}{\|r - V_1(V_1^T r)\|_2}$, $\beta = \|r - V_1(V_1^T r)\|_2^2$.

Step 7. *Compute the eigenvalue decomposition*

$$\begin{bmatrix} \Gamma_1 & \\ & 0 \end{bmatrix} + \frac{n}{n+1} \begin{bmatrix} r_2^T r_2 & \beta r_2 \\ \beta r_2^T & \beta^2 \end{bmatrix} = \Psi \tilde{\Gamma} \Psi^T,$$

where $\Psi \in \mathbf{R}^{(\tau_2+1)\times(\tau_2+1)}$, $\tilde{\Gamma} \in \mathbf{R}^{(\tau_2+1)\times(\tau_2+1)}$. *Denote* $\Psi_1 \in \mathbf{R}^{(\tau_2+1)\times\tilde{r}_2}$ *and* $\tilde{\Gamma}_1 \in \mathbf{R}^{\tilde{r}_2 \times \tilde{r}_2}$ *as the principal components of* Ψ *and* $\tilde{\Gamma}$, *respectively, determined by the eigenvalue threshold.*

Step 8. *Update* $\tilde{V}_1 = [V_1 \quad v]\,\Psi_1$.

Step 9. *Update* $\tilde{D}$ *as*

$$\tilde{d}_i = \begin{cases} \tilde{V}_1^T(V_1 d_i + c - \tilde{c}), & \text{if } i \le k, \\ \tilde{V}_1^T(x - \tilde{c}), & \text{if } i = k+1. \end{cases}$$

Steps 10-12. *The same as Steps 10-12 of Algorithm 2.6.*

The incremental ILDA/SSS algorithm for the case that data samples are inserted as a chunk is shown as follows:

Algorithm 2.8. *(Chunk ILDA/SSS)*

Input: *Principal eigenvectors and eigenvalues,* U_1, Λ_1. Γ_1 *and* V_1, *global centroid* c *and* $D = [d_1 \ \cdots \ d_k]$, *total class number* k, *size of each class* $N = [n_1 \ \cdots \ n_k]$, *number of total samples* n, *new data* $\mathcal{X} = [x_1 \ \cdots \ x_s]$ *with class label* ℓ_i $(i = 1, \cdots, s)$, *eigenvalue threshold.*

Output: *Updated* $\tilde{U}_1$, $\tilde{\Lambda}_1$, $\tilde{V}_1$, $\tilde{\Gamma}_1$, $\tilde{c}$, $\tilde{D}$, $\tilde{k}$, $\tilde{N}$, $\tilde{n}$ *and transformation matrix* $\tilde{G}$.

Step 1. *The same as Steps 1-5 of Algorithms 2.5 to construct* $\hat{c}$, $\hat{D} = \begin{bmatrix} \hat{d}_1 & \cdots & \hat{d}_s \end{bmatrix}$, $\hat{U}_1$, $\hat{\Lambda}_1$, $\hat{V}_1$ *and* $\hat{\Gamma}_1$ *for new samples* $\mathcal{X}$.

Step 2. $\tilde{c} = \frac{nc + s\hat{c}}{n+s}$, $\tilde{n} = n + s$.

Step 3. *Compute the economic QR factorization*

$$\begin{bmatrix} \hat{U}_1 - U_1 U_1^T \hat{U}_1 & c - \hat{c} - U_1 U_1^T (c - \hat{c}) \end{bmatrix} = U_2 Y,$$

where $U_2 \in \mathbf{R}^{m \times \hat{\tau}_1}$ *and* $Y \in \mathbf{R}^{\hat{\tau}_1 \times \hat{\tau}_1}$.

Step 4. *Compute the eigenvalue decomposition*

$$\begin{bmatrix} \Lambda_1 & \\ & 0 \end{bmatrix} + \begin{bmatrix} U_1^T \hat{U}_1 \hat{\Lambda}_1 \hat{U}_1^T U_1 & U_1^T \hat{U}_1 \hat{\Lambda}_1^2 U_2 \\ U_2^T \hat{\Lambda}_1^2 \hat{U}_1^T U_1 & U_2^T \hat{\Lambda}^3 U_2 \end{bmatrix}$$
$$+ \frac{ns}{n+s} \begin{bmatrix} U_1^T (c - \hat{c})(c - \hat{c})^T U_1 & U_1^T (c - \hat{c})(c - \hat{c})^T U_2^T \\ U_2^T (c - \hat{c})(c - \hat{c})^T U_1 & U_2^T (c - \hat{c})(c - \hat{c})^T U_2^T \end{bmatrix} = \Phi \tilde{\Lambda} \Phi^T,$$

where $\Phi \in \mathbf{R}^{(\tau_1 + \hat{\tau}_1) \times (\tau_1 + \hat{\tau}_1)}$, $\tilde{\Lambda} \in \mathbf{R}^{(\tau_1 + \hat{\tau}_1) \times (\tau_1 + \hat{\tau}_1)}$. *Denote* $\Phi_1 \in \mathbf{R}^{(\tau_1 + \hat{\tau}_1) \times \tilde{\tau}_1}$ *and* $\tilde{\Lambda}_1 \in \mathbf{R}^{\tilde{\tau}_1 \times \tilde{\tau}_1}$ *as the principal components of* Φ *and* $\tilde{\Lambda}$, *respectively, determined by the eigenvalue threshold.*

Step 5. *Update* $\tilde{U}_1 = [U_1 \ \ U_2] \, \Phi_1$.

Step 6. *Compute the economic QR factorization*

$$\begin{bmatrix} \hat{V}_1 - V_1 V_1^T \hat{V}_1 & c - \hat{c} - V_1 V_1^T (c - \hat{c}) \end{bmatrix} = V_2 Y,$$

where $V_2 \in \mathbf{R}^{m \times \hat{\tau}_2}$ *and* $Y \in \mathbf{R}^{\hat{\tau}_2 \times \hat{\tau}_2}$.

Step 7. *Denote* $r = V_1 d_i + c - \hat{V}_1 \hat{d}_i - \hat{c}$, *compute the eigenvalue decomposition*

$$\begin{bmatrix} \Gamma_1 & \\ & 0 \end{bmatrix} + \begin{bmatrix} V_1^T \hat{V}_1 \hat{\Gamma}_1 \hat{V}_1^T V_1 & V_1^T \hat{V}_1 \hat{\Gamma}_1^2 V_2 \\ V_2^T \hat{\Gamma}_1^2 \hat{V}_1^T V_1 & V_2^T \hat{\Gamma}^3 V_2 \end{bmatrix}$$
$$+ \frac{ns}{n+s} \begin{bmatrix} V_1^T (c-\hat{c})(c-\hat{c})^T V_1 & V_1^T (c-\hat{c})(c-\hat{c})^T V_2^T \\ V_2^T (c-\hat{c})(c-\hat{c})^T V_1 & V_2^T (c-\hat{c})(c-\hat{c})^T V_2^T \end{bmatrix}$$
$$- \sum_{i=1}^{\tilde{k}} \frac{n_i \hat{n}_i}{n_i + \hat{n}_i} \begin{bmatrix} V_1^T r r^T V_1 & V_1^T r r^T V_2 \\ V_2^T r r^T V_1 & V_2^T r r^T V_2 \end{bmatrix} = \Psi \tilde{\Gamma} \Psi^T,$$

where $\Psi \in \mathbf{R}^{(\tau_2+\hat{\tau}_2)\times(\tau_2+\hat{\tau}_2)}$, $\tilde{\Gamma} \in \mathbf{R}^{(\tau_2+\hat{\tau}_2)\times(\tau_2+\hat{\tau}_2)}$. *Denote* $\Psi_1 \in \mathbf{R}^{(\tau_2+\hat{\tau}_2)\times\tilde{\tau}_2}$ *and* $\tilde{\Gamma}_1 \in \mathbf{R}^{\tilde{\tau}_2\times\tilde{\tau}_2}$ *as the principal components of* Ψ *and* $\tilde{\Gamma}$, *respectively, determined by the eigenvalue threshold.*

Step 8. *Update* $\tilde{V}_1 = [V_1 \quad V_2]\,\Psi_1$.

Step 9. *Update* $\tilde{D}$ *as*

$$\tilde{d}_i = \tilde{V}_1^T \left(\frac{n_i (V_1 d_i + c) + \hat{n}_i (\hat{V}_1 \hat{d}_i + \hat{c})}{n_i + \hat{n}_i} - \tilde{c} \right).$$

Step 10. *Compute* $\tilde{Z} = \tilde{U}_1 \tilde{\Lambda}_1^{-1/2}$ *and economic QR factorization*

$$\tilde{Z}^T \tilde{V}_1 = \tilde{S} \tilde{T},$$

where $\tilde{S} \in \mathbf{R}^{\tilde{\tau}_1\times\tilde{\tau}_2}$, $\tilde{T} \in \mathbf{R}^{\tilde{\tau}_2\times\tilde{\tau}_2}$.

Step 11. *Compute the eigenvalue decomposition*

$$\tilde{S}^T \tilde{Z}^T \tilde{V}_1 \Gamma_1 \tilde{V}_1^T \tilde{Z} \tilde{S} = \tilde{H} \tilde{\Sigma} \tilde{H}^T,$$

where $\tilde{H} \in \mathbf{R}^{\tilde{\tau}_2\times\tilde{\tau}_2}$ *and* $\tilde{\Sigma} \in \mathbf{R}^{\tilde{\tau}_2\times\tilde{\tau}_2}$. *Denote* $\tilde{H}_1 \in \mathbf{R}^{\tilde{\tau}_2\times\tilde{\tau}}$ *as the principal components of* $\tilde{H}$ *determined by eigenvalue threshold.*

Step 12. $\tilde{G} = \tilde{Z} \tilde{S} \tilde{H}_1$.

The computational complexity of Algorithms 2.6, 2.7 and 2.8 is shown in Table 2.2, for the details of some matrix computation cost, please refer to Appendix B. When the samples in the dataset are linearly independent and the performance of ILDA/SSS approximate the performance of batch LDA, τ_1 ($\hat{\tau}_1$) would be close to $n-1$ ($s-1$), τ_2 ($\hat{\tau}_2$) and τ ($\hat{\tau}$) would be close to $k-1$ ($\hat{k}-1$). For this case we can observe from Table 2.2 that the main cost of ILDA/SSS for updating of one

Table 2.2: Computational complexity (flops) of algorithm ILDA/SSS

The computational cost of ILDA/SSS (Updating Existing Class)	
Steps 1-3:	$4m\tau_1$
Steps 4-5:	$12(\tau_1+1)^3+2\tau_1^2+2m(\tau_1+1)\tilde{\tau}_1$
Steps 6-8:	$8m\tau_2+12(\tau_2+1)^3+3\tau_2^2+2m(\tau_2+1)\tilde{\tau}_2$
Step 9:	$(2m\tau_2+2m+2m\tilde{\tau}_2)k+2m\tilde{\tau}_2$
Step 10:	$m\tilde{\tau}_1+2m\tilde{\tau}_1\tilde{\tau}_2+14\tilde{\tau}_1\tilde{\tau}_2^2+8\tilde{\tau}_2^3$
Step 11:	$2m\tilde{\tau}_1\tilde{\tau}_2+2\tilde{\tau}_1\tilde{\tau}_2^2+2\tilde{\tau}_2^3+\tilde{\tau}_2^2+12\tilde{\tau}_2^3$
Step 12:	$2m\tilde{\tau}\tilde{\tau}_1+2\tilde{\tau}\tilde{\tau}_1\tilde{\tau}_2$

The computational cost of ILDA/SSS (Updating New Class)	
Steps 1-3:	$4m\tau_1$
Steps 4-5:	$12(\tau_1+1)^3+2\tau_1^2+2m(\tau_1+1)\tilde{\tau}_1$
Steps 6-8:	$4m\tau_2+12(\tau_2+1)^3+2\tau_2^2+2m(\tau_2+1)\tilde{\tau}_2$
Step 9:	$(2m\tau_2+2m+2m\tilde{\tau}_2)k+2m\tilde{\tau}_2$
Step 10:	$m\tilde{\tau}_1+2m\tilde{\tau}_1\tilde{\tau}_2+14\tilde{\tau}_1\tilde{\tau}_2^2+8\tilde{\tau}_2^3$
Step 11:	$2m\tilde{\tau}_1\tilde{\tau}_2+2\tilde{\tau}_1\tilde{\tau}_2^2+2\tilde{\tau}_2^3+\tilde{\tau}_2^2+12\tilde{\tau}_2^3$
Step 12:	$2m\tilde{\tau}\tilde{\tau}_1+2\tilde{\tau}\tilde{\tau}_1\tilde{\tau}_2$

The computational cost of Chunk ILDA/SSS	
Step 1:	$2m(s^2+\hat{k}^2+s\hat{\tau}_1+\hat{k}\hat{\tau}_2)+12s^3+12\hat{k}^3$
Step 3:	$4m\tau_1\hat{\tau}_1+4m\hat{\tau}_1^2-\frac{4}{3}\hat{\tau}_1^3$
Step 4:	$12(\tau_1+\hat{\tau}_1)^3$
Step 5:	$2m(\tau_1+\hat{\tau}_1)^2$
Step 6:	$4m\tau_2\hat{\tau}_2+4m\hat{\tau}_2^2-\frac{4}{3}\hat{\tau}_2^3$
Step 7:	$12(\tau_2+\hat{\tau}_2)^3$
Step 8:	$2m(\tau_2+\hat{\tau}_2)^2$
Step 9:	$(2m\tau_2+2m+2m\tilde{\tau}_2)k+2m\tilde{\tau}_2$
Step 10:	$m\tilde{\tau}_1+2m\tilde{\tau}_1\tilde{\tau}_2+14\tilde{\tau}_1\tilde{\tau}_2^2+8\tilde{\tau}_2^3$
Step 11:	$2m\tilde{\tau}_1\tilde{\tau}_2+2\tilde{\tau}_1\tilde{\tau}_2^2+2\tilde{\tau}_2^3+\tilde{\tau}_2^2+12\tilde{\tau}_2^3$
Step 12:	$2m\tilde{\tau}\tilde{\tau}_1+2\tilde{\tau}\tilde{\tau}_1\tilde{\tau}_2$

single sample is $2mn^2 + 12n^3$ and updating of a chunk of s samples is $2m(n+s)^2 + 12(n+s)^3$, which are relatively high compared with IDR/QR [102]. Like IDR/QR, ILDA/SSS is an approximate scheme that only principal eigenvalues and eigenvectors of the scatter matrices are computed and used in the incremental learning process.

2.3 Least Square Incremental LDA (LS-ILDA)

The batch algorithm of LS-ILDA [69] is based on the idea of multivariate linear regression in [97]. In [97], LDA is put into the framework of multivariate linear regression by adding a constraint to the optimization problem of LDA,

$$\max_{\mathcal{Y}} \operatorname{trace}(G^T S_b G (G^T S_t G)^{(+)}) \tag{2.1}$$
$$\text{s.t. } G = (H_t H_t^T)^{(+)} H_t \mathcal{Y},$$

where $H_t \in \mathbf{R}^{m\times n}$ is the centered data matrix, and $\mathcal{Y}$ is the indicator matrix to be optimized. Optimization problem (2.1) is solved by

$$G = (H_t^{(+)})^T \mathcal{Y}, \quad \mathcal{Y} = [y_{ij}], \quad y_{ij} = \begin{cases} \sqrt{\frac{n}{n_j}} - \sqrt{\frac{n_j}{n}} & \text{if } a_i \text{ belongs to class } j, \\ -\sqrt{\frac{n_j}{n}} & \text{otherwise.} \end{cases}$$

When $\operatorname{rank}(A) = n$, the above optimal solution of (2.1) is proved to be equivalent to the solution of LDA scaled by an orthogonal matrix [97].

In [69], Liu et al. simplify the indicator matrix $\mathcal{Y} = [y_{ij}]$ to be

$$y_{ij} = \begin{cases} \frac{1}{\sqrt{n_j}} & \text{if } a_i \text{ belongs to class } j, \\ 0 & \text{otherwise,} \end{cases} \tag{2.2}$$

for convenience of updating, which leads to the following batch algorithm of LS-ILDA:

Algorithm 2.9. *(Batch LS-ILDA)*

Input: *Data matrix $A \in \mathbf{R}^{m\times n}$ with class label.*

Output: *Transformation matrix* $G \in \mathbf{R}^{m\times k}$.

Step 1. *Construct* $H_t \in \mathbf{R}^{m\times n}$ *and indicator matrix* $\mathcal{Y} \in \mathbf{R}^{n\times k}$ *from (2.2).*

Step 2. *Compute* $H_t^{(+)}$.

Step 3. $G = (H_t^{(+)})^T\mathcal{Y}$.

The incremental algorithm, LS-ILDA [69], of the above batch algorithm consists of three parts: update the indicator matrix $\mathcal{Y}$, update the centered data matrix H_t and its pseudoinverse $H_t^{(+)}$, where the updating of $H_t^{(+)}$ needs the condition that the new inserted data is linearly independent to the original data set. The algorithms for LS_ILDA to update an existing class and a new class when a new data is inserted are presented in Algorithms 2.10 and 2.11, respectively, for the case that $m > n$.

Algorithm 2.10. *(LS-ILDA: Updating Existing Class)*

Input: *Centered data matrix* $H_t \in \mathbf{R}^{m\times n}$, *its pseudo-inverse* $H_t^{(+)}$, *global centroid* c, *indicator matrix* $\mathcal{Y}$, *total class number* k, *size of each class* $N = [n_1 \ \cdots \ n_k]$, *transformation matrix* G, *new data* x *with class label* $1 \le \ell \le k$.

Output: *Updated* $\tilde{H}_t$, $\tilde{H}_t{}^{(+)}$, $\tilde{c}$, $\tilde{\mathcal{Y}}$, $\tilde{k}$, $\tilde{N}$ *and new transformation matrix* $\tilde{G}$.

Step 1. $\tilde{n}_j = n_j$ $(j \neq \ell)$, $\tilde{n}_\ell = n_\ell + 1$, $\tilde{c} = c + \frac{1}{n+1}(x - c)$, $\tilde{k} = k$.

Step 2. *Define* $y \in \mathbf{R}^{\tilde{k}}$ *as*

$$y^T = \begin{bmatrix} 0 & \cdots & \underset{\ell}{\frac{1}{\sqrt{\tilde{n}_\ell}}} & \cdots & 0 \end{bmatrix},$$

and update

$$\tilde{\mathcal{Y}} = \begin{bmatrix} \mathcal{Y} & \begin{bmatrix} I_{\ell-1} & & \\ & \frac{\sqrt{n_\ell}}{\sqrt{n_\ell+1}} & \\ & y^T & \end{bmatrix} & I_{k-\ell} \end{bmatrix}.$$

Step 3. *Update the centered data matrix as*

$$\tilde{H}_t = \begin{bmatrix} H_t - \frac{1}{n+1}(x-c)e^T & \frac{n}{n+1}(x-c) \end{bmatrix}, \quad e = [1 \ \cdots \ 1]^T \in \mathbf{R}^n.$$

Step 4. *Update the pseudo-inverse as*

$$\tilde{H}_t^{(+)} = \begin{bmatrix} H_t^{(+)} - H_t^{(+)}(x-c)h^T - \frac{1}{n}eh^T \\ h^T \end{bmatrix},$$

where

$$h = \frac{x - c - H_t H_t^{(+)}(x-c)}{(x-c)^T(x-c) - (x-c)^T H_t H_t^{(+)}(x-c)}.$$

Step 5. *Compute the transformation matrix*

$$\tilde{G} = (G - h(x-c)^T G - \frac{1}{n}he^T\mathcal{Y}) \begin{bmatrix} I_{\ell-1} & & \\ & \frac{\sqrt{n_\ell}}{\sqrt{n_\ell+1}} & \\ & & I_{k-\ell} \end{bmatrix} + hy^T.$$

Algorithm 2.11. *(LS-ILDA: Updating New Class)*

Input: *Centered data matrix $H_t \in \mathbf{R}^{m\times n}$, its pseudo-inverse $H_t^{(+)}$, global centroid c, indicator matrix $\mathcal{Y}$, total class number k, size of each class $N = [n_1 \ \cdots \ n_k]$, transformation matrix G, new data x with class label $\ell > k$.*

Output: *Updated $\tilde{H}_t$, $\tilde{H}_t^{(+)}$, $\tilde{c}$, $\tilde{\mathcal{Y}}$, $\tilde{k}$, $\tilde{N}$ and new transformation matrix $\tilde{G}$.*

Step 1. $\tilde{n}_j = n_j$ $(j = 1, \cdots, k)$, $\tilde{n}_\ell = 1$, $\tilde{c} = c + \frac{1}{n+1}(x-c)$, $\tilde{k} = k+1$.

Step 2. *Define $y \in \mathbf{R}^{\tilde{k}}$ as*

$$y^T = [0 \ \cdots \ 0 \ \ 1],$$

and update

$$\tilde{\mathcal{Y}} = \begin{bmatrix} \mathcal{Y} & 0 \\ y^T & \end{bmatrix}.$$

Step 3. *Update the centered data matrix as*

$$\tilde{H}_t = \left[H_t - \tfrac{1}{n+1}(x-c)e^T \quad \tfrac{n}{n+1}(x-c)\right], \quad e = [1 \ \cdots \ 1]^T \in \mathbf{R}^n.$$

Step 4. *Update the pseudo-inverse as*

$$\tilde{H}_t^{(+)} = \begin{bmatrix} H_t^{(+)} - H_t^{(+)}(x-c)h^T - \frac{1}{n}eh^T \\ h^T \end{bmatrix},$$

where

$$h = \frac{x - c - H_t H_t^{(+)}(x-c)}{(x-c)^T(x-c) - (x-c)^T H_t H_t^{(+)}(x-c)}.$$

Step 5. *Compute the transformation matrix*

$$\tilde{G} = \begin{bmatrix} G - h(x-c)^T G - \frac{1}{n} h e^T \mathcal{Y} & 0 \end{bmatrix} + h y^T.$$

The incremental LS-ILDA algorithm for the case that data samples are inserted as a chunk is summarized as follows:

Algorithm 2.12. *(Chunk LS-ILDA)*

Input: *Centered data matrix* $H_t \in \mathbf{R}^{m \times n}$, *its pseudo-inverse* $H_t^{(+)}$, *global centroid* c, *indicator matrix* $\mathcal{Y}$, *total class number* k, *size of each class* $N = [n_1 \ \cdots \ n_k]$, *transformation matrix* G, *new data* $\mathcal{X} = [x_1 \ \cdots \ x_s]$ *with class labels* ℓ_i $(i = 1, \cdots, s)$.

Output: *Updated* $\tilde{H}_t$, $\tilde{H}_t^{(+)}$, $\tilde{c}$, $\tilde{\mathcal{Y}}$, $\tilde{k}$, $\tilde{N}$ *and new transformation matrix* $\tilde{G}$.

Step 1. *For each data item* x_i $(i = 1, \cdots, r)$, *determine whether it is from an existing class or a new class.*

Step 2. *If* x_i *is from an existing class, update* $\tilde{H}_t$, $\tilde{H}_t^{(+)}$, $\tilde{c}$, $\tilde{\mathcal{Y}}$, $\tilde{k}$, $\tilde{N}$ *and* $\tilde{G}$ *following Algorithm 2.10.*

Step 3. *If* x_i *is from a new class, update* $\tilde{H}_t$, $\tilde{H}_t^{(+)}$, $\tilde{c}$, $\tilde{\mathcal{Y}}$, $\tilde{k}$, $\tilde{N}$ *and* $\tilde{G}$ *following Algorithm 2.11.*

Table 2.3: Computational complexity (flops) of algorithm LS-ILDA

The computational cost of LS-ILDA (Updating Existing Class)	
Steps 1-3:	mn
Step 4:	$13mn$
Step 5:	$7mk + 2nk$

The computational cost of LS-ILDA (Updating New Class)	
Steps 1-3:	mn
Step 4:	$13mn$
Step 5:	$7mk + 2nk$

The computational complexity of Algorithms 2.10 and 2.11 are shown in Table 2.3. We can summarize from Table 2.3 that the main cost of LS-ILDA for one single

insertion is $14mn+7mk$. Accordingly, the main cost of LS-ILDA for the insertion of a chunk of s samples is $14mns+7mks$, which is very low compared with ILDA/SSS. Besides its computational efficiency, LS-ILDA produces exact least square solution of batch LDA. However, LS-ILDA requires that the new inserted data sample is linearly independent to the training set. This may limit the application of LS-ILDA.

2.4 Incremental Complete LDA (ICLDA)

The batch version of ICLDA [70] is the complete linear discriminant analysis (CLDA) method proposed by Yang et al. [95]. CLDA first divides the whole data space into two complementary subspaces, the range space of S_w and the null space of S_w. Then, CLDA obtains its discriminant vectors in both of these two subspaces. In [70], Lu et al. improved the performance of CLDA by applying QR factorization instead of SVD to obtain the range space of total scatter matrix as well as the range space and null space of the reduced within-class scatter matrix.

For simplicity, we summarize CLDA and ICLDA for the case that $\text{rank}(A) = n$ in this section. Before presenting the new CLDA algorithm in [70], we give some new notations used in this section. Separate data matrix $A \in \mathbf{R}^{m\times n}$ and data matrix of each cluster $\mathcal{A}_i \in \mathbf{R}^{m\times n_i}$ $(i = 1, \cdots, k)$ into two parts, respectively,

$$A = \begin{bmatrix} a_1 & \bar{A} \end{bmatrix}, \quad \mathcal{A}_i = \begin{bmatrix} a_{i1} & \bar{\mathcal{A}}_i \end{bmatrix},$$

where $a_1 \in \mathbf{R}^m$ is the first data point of A and $\bar{A} \in \mathbf{R}^{m\times(n-1)}$ consists of the remaining part, $a_{i1} \in \mathbf{R}^m$ is the first data point of $\mathcal{A}_i$ and $\bar{\mathcal{A}}_i \in \mathbf{R}^{m\times(n_i-1)}$ consists of the remaining part. Define $H_t^d \in \mathbf{R}^{m\times(n-1)}$ and $H_w^d \in \mathbf{R}^{m\times(n-k)}$ as

$$H_t^d = \bar{A} - a_1\bar{e}^T, \quad H_w^d = \begin{bmatrix} \bar{\mathcal{A}}_1 - a_{11}\bar{e}_1^T & \cdots & \bar{\mathcal{A}}_k - a_{k1}\bar{e}_k^T \end{bmatrix},$$

where

$$\bar{e} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathrm{R}^{n-1}, \quad \bar{e}_i = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathrm{R}^{n_i-1}, \quad i = 1, \cdots, k.$$

It is easy to verify that H_t^d has the same range space as H_t and H_w^d has the same range space as H_w. Compute the economic QR factorization $H_t^d = Q^d R_1$, where $Q^d \in \mathbf{R}^{m\times(n-1)}$ is column orthogonal and $R_1 \in \mathbf{R}^{(n-1)\times(n-1)}$ is upper triangular. Project H_w, H_b and H_w^d into the range space of H_t,

$$H_W = (Q^d)^T H_w, \quad H_B = (Q^d)^T H_b, \quad (Q^d)^T H_w^d,$$

then obtain the range space and null space of S_w by computing full QR factorization of $(Q^d)^T H_w^d$ as

$$(Q^d)^T H_w^d = \begin{bmatrix} P_1 & P_2 \end{bmatrix} R_2,$$

where $P_1 \in \mathbf{R}^{(n-1)\times(n-k)}$, whose columns form an orthonormal basis of the range space of $(Q^d)^T H_w^d$, $P_2 \in \mathbf{R}^{(n-1)\times(k-1)}$, whose columns form an orthonormal basis of the null space of $(Q^d)^T H_w^d$, and $R_2 \in \mathbf{R}^{(n-1)\times(n-k)}$ is upper triangular. Compute the SVD of $H_{B2} = P_2^T H_B \in \mathbf{R}^{(k-1)\times k}$ as $H_{B2} = Z\Sigma V^T$, where $Z \in \mathbf{R}^{(k-1)\times(k-1)}$ and $V \in \mathbf{R}^{k\times k}$ are orthogonal and $\Sigma \in \mathbf{R}^{(k-1)\times k}$ is diagonal. Then the optimal discriminant matrix contained in the null space of S_w is

$$G_{nul} = Q^d P_2 Z \in \mathbf{R}^{m\times(k-1)}.$$

To obtain the optimal discriminant matrix in the range space of S_w, compute the generalized eigenvalue decomposition of (S_{B1}, S_{W1}) as

$$S_{B1} X = S_{W1} X \Lambda,$$

where $S_{W1} = P_1^T H_W H_W^T P_1$, and $S_{B1} = P_1^T H_B H_B^T P_1$, $X \in \mathbf{R}^{(n-k)\times(n-k)}$ is orthogonal and $\Lambda \in \mathbf{R}^{(n-k)\times(n-k)}$ is diagonal with the diagonal entries sorted in the nonincreasing order. Then

$$G_{ran} = Q^d P_1 X \in \mathbf{R}^{m\times(n-k)},$$

is the optimal matrix in the range space of S_w, and the final optimal transformation matrix consisting of two discriminant matrices is

$$G = \begin{bmatrix} G_{null} & G_{ran} \end{bmatrix} \in \mathbf{R}^{m\times(n-1)}.$$

The new CLDA developed in [70] is shown as follows:

Algorithm 2.13. *(CLDA)*

Input: *Data matrix* $A \in \mathbf{R}^{m\times n}$ *with cluster label.*

Output: *Transformation matrix* $G \in \mathbf{R}^{m\times(n-1)}$.

Step 1. *Construct* $H_w \in \mathbf{R}^{m\times n}$, $H_b \in \mathbf{R}^{m\times k}$, $H_w^d \in \mathbf{R}^{m\times(n-k)}$ *and* $H_t^d \in \mathbf{R}^{m\times(n-1)}$.

Step 2. *Compute the economic QR factorization of* H_t^d *as*

$$H_t^d = Q^d R_1, \quad Q^d \in \mathbf{R}^{m\times(n-1)}, \quad R_1 \in \mathbf{R}^{(n-1)\times(n-1)}.$$

Step 3. *Compute* $H_W = (Q^d)^T H_w$, $H_B = (Q^d)^T H_b$ *and* $(Q^d)^T H_w^d$.

Step 4. *Compute the full QR factorization of* $(Q^d)^T H_w^d$ *as*

$$(Q^d)^T H_w^d = [P_1 \quad P_2]\, R_2,$$

where $P_1 \in \mathbf{R}^{(n-1)\times(n-k)}$, $P_2 \in \mathbf{R}^{(n-1)\times(k-1)}$, *and* $R_2 \in \mathbf{R}^{(n-1)\times(n-k)}$.

Step 5. $H_{B2} = P_2^T H_B$.

Step 6. *Compute the SVD of* H_{B2} *as*

$$H_{B2} = Z\Sigma V^T, \quad Z \in \mathbf{R}^{(k-1)\times(k-1)}, \quad V \in \mathbf{R}^{k\times k}, \quad \Sigma \in \mathbf{R}^{(k-1)\times k}$$

Step 7. $G_{nul} = Q^d P_2 Z$.

Step 8. $S_{W1} = P_1^T H_W H_W^T P_1$, $S_{B1} = P_1^T H_B H_B^T P_1$.

Step 9. *Compute the generalized eigenvalue decomposition of* (S_{B1}, S_{W1}) *as*

$$S_{B1}X = S_{W1}X\Lambda, \quad \Lambda \in \mathbf{R}^{(n-k)\times(n-k)}, \quad X \in \mathbf{R}^{(n-k)\times(n-k)}.$$

Step 10. $G_{ran} = Q^d P_1 X$.

Step 11. $G = [G_{nul} \quad G_{ran}]$.

Based on this new implementation of CLDA, the incremental algorithm ICLDA includes updating of H_b, H_w, H_t^d, H_w^d, Q^d, H_W, H_B, and the QR-updating of $(Q^d)^T H_w^d$ as well as the updating of local centroids and global centroid. We show two implementations of ICLDA in Algorithms 2.14 and 2.15: one for updating an existing class and the the other one for updating a new class.

Algorithm 2.14. *(ICLDA: Updating Existing Class)*

Input: $H_b \in \mathbf{R}^{m\times k}$, $H_w \in \mathbf{R}^{m\times n}$, $H_w^d \in \mathbf{R}^{m\times(n-k)}$, $H_t^d \in \mathbf{R}^{m\times(n-1)}$, $Q^d \in \mathbf{R}^{m\times(n-1)}$, $H_W \in \mathbf{R}^{(n-1)\times n}$, $H_B \in \mathbf{R}^{(n-1)\times k}$, *QR factorization of* $(Q^d)^T H_w^d = [P_1 \quad P_2]\, R_2$ *with* $P_1 \in \mathbf{R}^{(n-1)\times(n-k)}$, $P_2 \in \mathbf{R}^{(n-1)\times(k-1)}$ *and* $R_2 \in \mathbf{R}^{(n-1)\times(n-k)}$, *global centroid* c, *local centroids* $C = [c_1 \quad \cdots \quad c_k]$, *size of each class* $N = [n_1 \quad \cdots \quad n_k]$, *new inserted sample* x *with class label* $1 \le \ell \le k$.

Output: *Updated* $\tilde{H}_b$, $\tilde{H}_w$, $\tilde{H}_w^d$, $\tilde{H}_t^d$, $\tilde{Q}^d$, $\tilde{H}_W$, $\tilde{H}_B$, *QR factorization of* $(\tilde{Q}^d)^T \tilde{H}_w^d = \begin{bmatrix}\tilde{P}_1 & \tilde{P}_2\end{bmatrix} \tilde{R}_2$, $\tilde{c}$, $\tilde{C}$, $\tilde{N}$ *and transformation matrix* $\tilde{G}$.

Step 1. $\tilde{n}_i = n_i$ $(i \neq \ell)$, $\tilde{n}_\ell = n_\ell + 1$, $\tilde{c}_i = c_i$ $(i \neq \ell)$, $\tilde{c}_\ell = c_\ell + f_\ell$, $\tilde{c} = c + f$, *where*

$$f = \frac{x - c}{n+1}, \quad f_\ell = \frac{x - c_i}{n_i + 1}.$$

Step 2. *Update*

$$\tilde{H}_b = H_b - f\,[\sqrt{n_1} \quad \cdots \quad \sqrt{n_k}] - \sqrt{n_\ell}(c_\ell - \tilde{c})g_1^T + \sqrt{n_\ell + 1}(\tilde{c}_\ell - \tilde{c})g_1^T,$$

$$\tilde{H}_w = [H_w \quad x - c_\ell] - f_\ell g_2^T, \quad \tilde{H}_w^d = \begin{bmatrix}H_w^d & x - a_{\ell 1}\end{bmatrix}, \quad \tilde{H}_t^d = \begin{bmatrix}H_t^d & x - a_1\end{bmatrix},$$

where $g_1 \in \mathbf{R}^k$ *is a unit vector with the* ℓ*-th item equals to* 1 *and* $g_2 \in \mathbf{R}^{n+1}$ *is a column vector with the items equals to* 1 *if the sample belongs to the* ℓ*-th class, otherwise the item is* 0.

Step 3. *Update* $\tilde{Q}^d = \begin{bmatrix}Q^d & q\end{bmatrix}$, *where* $q = \frac{x - a_1 - Q^d((Q^d)^T(x - a_1))}{\left\|x - a_1 - Q^d((Q^d)^T(x - a_1))\right\|_2}$.

Step 4. *Update*

$$\tilde{H}_W = \begin{bmatrix} H_W & (Q^d)^T(x - c_\ell) \\ q^T H_w & q^T(x - c_\ell) \end{bmatrix} - ((\tilde{Q}^d)^T f_\ell)g_2^T,$$

$$\tilde{H}_B = \begin{bmatrix} H_B \\ q^T H_b \end{bmatrix} + (\tilde{Q}^d)^T(-f\,[\sqrt{n_1} \quad \cdots \quad \sqrt{n_k}] - \sqrt{n_\ell}(c_\ell - \tilde{c})g_1^T + \sqrt{n_\ell + 1}(\tilde{c}_\ell - \tilde{c})g_1^T).$$

Step 5. *Do the QR-updating of*

$$\begin{bmatrix} (Q^d)^T H_w^d \\ q^T H_w^d \end{bmatrix},$$

via appending a row $q^T H_w^d$ *to* $(Q^d)^T H_w^d$.

Step 6. *Do the QR-updating of* $(\tilde{Q}^d)^T \tilde{H}_w^d = \begin{bmatrix}\tilde{P}_1 & \tilde{P}_2\end{bmatrix} \tilde{R}_2$ *via appending a column*

$$\begin{bmatrix} (Q^d)^T(x - a_{\ell 1}) \\ q^T(x - a_{\ell 1}) \end{bmatrix} \text{ to } \begin{bmatrix} (Q^d)^T H_w^d \\ q^T H_w^d \end{bmatrix}.$$

Step 7. $\tilde{H}_{B2} = \tilde{P}_2^T \tilde{H}_B$.

Step 8. *Compute the SVD of* $\tilde{H}_{B2}$ *as*

$$\tilde{H}_{B2} = \tilde{Z}\tilde{\Sigma}\tilde{V}^T, \quad \tilde{Z} \in \mathbf{R}^{(k-1)\times(k-1)}, \quad \tilde{V} \in \mathbf{R}^{k\times k}, \quad \tilde{\Sigma} \in \mathbf{R}^{(k-1)\times k}.$$

Step 9. $\tilde{G}_{nul} = \tilde{Q}^d \tilde{P}_2 Z$.

Step 10. $\tilde{S}_{W1} = \tilde{P}_1^T \tilde{H}_W \tilde{H}_W^T \tilde{P}_1$, $\tilde{S}_{B1} = \tilde{P}_1^T \tilde{H}_B \tilde{H}_B^T \tilde{P}_1$.

Step 11. *Compute the generalized eigenvalue decomposition of* $(\tilde{S}_{B1}, \tilde{S}_{W1})$ *as*

$$\tilde{S}_{B1}\tilde{X} = \tilde{S}_{W1}\tilde{X}\tilde{\Lambda},$$

where $\tilde{\Lambda} \in \mathbf{R}^{s\times s}$ *is diagonal with the diagonal entries sorted in the nonincreasing order,* $\tilde{X} \in \mathbf{R}^{s\times s}$.

Step 12. $\tilde{G}_{ran} = \tilde{Q}^d \tilde{P}_1 \tilde{Y}$.

Step 13. $\tilde{G} = \begin{bmatrix} \tilde{G}_{nul} & \tilde{G}_{ran} \end{bmatrix}$.

Algorithm 2.15. *(ICLDA: Updating New Class)*

Input: $H_b \in \mathbf{R}^{m\times k}$, $H_w \in \mathbf{R}^{m\times n}$, $H_w^d \in \mathbf{R}^{m\times(n-k)}$, $H_t^d \in \mathbf{R}^{m\times(n-1)}$, $Q^d \in \mathbf{R}^{m\times(n-1)}$, $H_W \in \mathbf{R}^{(n-1)\times n}$, $H_B \in \mathbf{R}^{(n-1)\times k}$, *QR factorization of* $(Q^d)^T H_w^d = \begin{bmatrix} P_1 & P_2 \end{bmatrix} R_2$ *with* $P_1 \in \mathbf{R}^{(n-1)\times(n-k)}$, $P_2 \in \mathbf{R}^{(n-1)\times(k-1)}$ *and* $R_2 \in \mathbf{R}^{(n-1)\times(n-k)}$, *global centroid* c, *local centroids* $C = [c_1 \ \cdots \ c_k]$, *size of each class* $N = [n_1 \ \cdots \ n_k]$, *new inserted sample* x *with class label* $\ell > k$.

Output: *Updated* $\tilde{H}_b$, $\tilde{H}_w$, $\tilde{H}_w^d$, $\tilde{H}_t^d$, $\tilde{Q}^d$, $\tilde{H}_W$, $\tilde{H}_B$, *QR factorization of* $(\tilde{Q}^d)^T \tilde{H}_w^d = \begin{bmatrix} \tilde{P}_1 & \tilde{P}_2 \end{bmatrix} \tilde{R}_2$, $\tilde{c}$, $\tilde{C}$, $\tilde{N}$ *and transformation matrix* $\tilde{G}$.

Step 1. $\tilde{n}_i = n_i$ $(i \le k)$, $\tilde{n}_{k+1} = 1$, $\tilde{c}_i = c_i$ $(i \le k)$, $\tilde{c}_{k+1} = x$, $\tilde{c} = c + f$, *where*

$$f = \frac{x - c}{n + 1}.$$

Step 2. *Update*

$$\tilde{H}_b = [H_b \quad x - c] - f\,[\sqrt{n_1} \ \cdots \ \sqrt{n_k} \quad 1],$$
$$\tilde{H}_w = [H_w \quad 0], \quad \tilde{H}_w^d = H_w^d, \quad \tilde{H}_t^d = \begin{bmatrix} H_t^d & x - a_1 \end{bmatrix}.$$

Step 3. *Update* $\tilde{Q}^d = \begin{bmatrix} Q^d & q \end{bmatrix}$, *where* $q = \frac{x - a_1 - Q^d((Q^d)^T(x - a_1))}{\left\| x - a_1 - Q^d((Q^d)^T(x - a_1)) \right\|_2}$.

Step 4. *Update*

$$\tilde{H}_W = \begin{bmatrix} H_W & 0 \\ q^T H_w & 0 \end{bmatrix},$$

and

$$\tilde{H}_B = \begin{bmatrix} H_B & (Q^d)^T(x-c) \\ q^T H_b & q^T(x-c) \end{bmatrix} - (\tilde{Q}^d)^T f \begin{bmatrix} \sqrt{n_1} & \cdots & \sqrt{n_k} & 1 \end{bmatrix}.$$

Step 5. *Do the QR-updating of*

$$\begin{bmatrix} (Q^d)^T H_w^d \\ q^T H_w^d \end{bmatrix} = \begin{bmatrix} \tilde{P}_1 & \tilde{P}_2 \end{bmatrix} \tilde{R}_2,$$

via appending a row $q^T H_w^d$ *to* $(Q^d)^T H_w^d$.

Steps 6-12. *The same as the Steps 7-13 of Algorithm 2.14.*

The incremental ICLDA algorithm for updating a chunk of samples is shown as follows:

Algorithm 2.16. *(Chunk ICLDA)*

Input: $H_b \in \mathbf{R}^{m\times k}$, $H_w \in \mathbf{R}^{m\times n}$, $H_w^d \in \mathbf{R}^{m\times(n-k)}$, $H_t^d \in \mathbf{R}^{m\times(n-1)}$, $Q^d \in \mathbf{R}^{m\times(n-1)}$, $H_W \in \mathbf{R}^{(n-1)\times n}$, $H_B \in \mathbf{R}^{(n-1)\times k}$, *QR factorization of* $(Q^d)^T H_w^d = \begin{bmatrix} P_1 & P_2 \end{bmatrix} R_2$ *with* $P_1 \in \mathbf{R}^{(n-1)\times(n-k)}$, $P_2 \in \mathbf{R}^{(n-1)\times(k-1)}$ *and* $R_2 \in \mathbf{R}^{(n-1)\times(n-k)}$, *global centroid* c, *local centroids* $C = \begin{bmatrix} c_1 & \cdots & c_k \end{bmatrix}$, *size of each class* $N = \begin{bmatrix} n_1 & \cdots & n_k \end{bmatrix}$, *new inserted sample* x *with class label* $\ell > k$.

Output: *Updated* $\tilde{H}_b$, $\tilde{H}_w$, $\tilde{H}_w^d$, $\tilde{H}_t^d$, $\tilde{Q}^d$, $\tilde{H}_W$, $\tilde{H}_B$, *QR factorization of* $(\tilde{Q}^d)^T \tilde{H}_w^d = \begin{bmatrix} \tilde{P}_1 & \tilde{P}_2 \end{bmatrix} \tilde{R}_2$, $\tilde{c}$, $\tilde{C}$, $\tilde{N}$ *and transformation matrix* $\tilde{G}$.

Step 1. *For each data item* x_i $(i = 1, \cdots, r)$, *determine whether it is from an existing class or a new class.*

Step 2. *If* x_i *is from an existing class, update* $\tilde{H}_b$, $\tilde{H}_w$, $\tilde{H}_w^d$, $\tilde{H}_t^d$, $\tilde{Q}^d$, $\tilde{H}_W$, $\tilde{H}_B$, *QR factorization of* $(\tilde{Q}^d)^T \tilde{H}_w^d = \begin{bmatrix} \tilde{P}_1 & \tilde{P}_2 \end{bmatrix} \tilde{R}_2$, $\tilde{c}$, $\tilde{C}$, $\tilde{N}$ *and* $\tilde{G}$ *following Algorithm 2.14.*

Step 3. *If* x_i *is from a new class, update* $\tilde{H}_b$, $\tilde{H}_w$, $\tilde{H}_w^d$, $\tilde{H}_t^d$, $\tilde{Q}^d$, $\tilde{H}_W$, $\tilde{H}_B$, *QR factorization of* $(\tilde{Q}^d)^T \tilde{H}_w^d = \begin{bmatrix} \tilde{P}_1 & \tilde{P}_2 \end{bmatrix} \tilde{R}_2$, $\tilde{c}$, $\tilde{C}$, $\tilde{N}$ *and* $\tilde{G}$ *following Algorithm 2.15.*

Table 2.4 shows the computational cost of ICLDA for Algorithms 2.14 and 2.15, where the details of some matrix computation cost are given in Appendix B. In short, the main cost of ICLDA for one insertion is about $2mn^2 + 20n^3$ and for the insertion of a chunk of s samples is $2mn^2s + 20n^3s$, thus, it is time consuming for ICLDA to incorporate a sample especially when sample size n is large. ICLDA is an exact scheme, however, the dimension of the reduced space which equals to the rank of total scatter matrix S_t is high, it is actually $n-1$ when the training samples are linearly independent.

Table 2.4: Computational complexity (flops) of algorithm ICLDA

The computational cost of ICLDA (Updating Existing Class)	
Steps 1-6:	$O(mn)$
Step 7:	$2nk^2$
Step 8:	$12k^3$
Step 9:	$2mnk + 2nk^2$
Step 10:	$2n^2(n-k) + 2n(n-k)^2 + 2n(n-k)k + 2(n-k)^2k$
Step 11:	$14(n-k)^3$
Step 12:	$2mn(n-k) + 2n(n-k)^2$

The computational cost of ICLDA (Updating New Class)	
Steps 1-5:	$O(mn)$
Step 6:	$2nk^2$
Step 7:	$12k^3$
Step 8:	$2mnk + 2nk^2$
Step 9:	$2n^2(n-k) + 2n(n-k)^2 + 2n(n-k)k + 2(n-k)^2k$
Step 10:	$14(n-k)^3$
Step 11:	$2mn(n-k) + 2n(n-k)^2$

Chapter 3

New Incremental LDA

Incremental dimensionality reduction methods have been proven to perform efficiently if large amounts of training data have to be processed or if not all data is available in advance. Several incremental LDA algorithms have been developed and achieve success, however, as we have mentioned previously, existing LDA-based incremental algorithms either provide approximate solutions or suffer from high computational cost.

So in this chapter, to overcome the limitations of existing methods, we develop a novel incremental LDA method called ILDA/QR, under a mild condition which has been shown to hold for most high-dimensional datasets. In order to develop ILDA/QR we first propose a new, simple and efficient implementation of batch LDA, called LDA/QR, which applies QR factorization to data matrix rather than scatter matrices. The distinct advantage of this strategy is that it enables the efficient incremental learning of ILDA/QR when new samples are inserted. Our theoretical analysis shows that ILDA/QR produces the exact transformation of LDA/QR, more importantly, it can easily handle not only the case that only one new sample is inserted but also the case that a chunk of new samples are added. Extensive experiments confirm the claimed theoretical analysis of discrimination and theoretical estimate of computational efficiency.

The rest of this chapter is organized as follows. Some foundamental knowledge

including new representations of scatter matrices, is given beforehand in Section 3.1. Our new proposed LDA/QR and its incremental version ILDA/QR are shown in Section 3.2 and Section 3.3, respectively. The experimental results are reported in Section 3.4 and Section 3.5 draws the conclusions.

3.1 Preliminaries

Lemma 3.1. *Let* $X,\ Y,\ Z \in \mathbf{R}^{\mu\times\mu}$ *be symmetric positive semi-definite, and*

$$\mathrm{rank}\begin{bmatrix} X & Z \end{bmatrix} = \mathrm{rank}(X) = \mathrm{rank}(Z).$$

Then

$$\mathrm{trace}((X+Y)^{(+)}Z) \leq \mathrm{trace}(X^{(+)}Z),$$

and the equality holds if and only if

$$\mathrm{rank}\begin{bmatrix} X & Y \end{bmatrix} = \mathrm{rank}(X) + \mathrm{rank}(Y),$$

which is equivalent to

$$Y = 0$$

if X *is positive definite.*

Proof. We can assume without loss of generality that

$$X = \begin{bmatrix} I_{\mu_1} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},\quad Y = \begin{bmatrix} Y_{1,1} & Y_{1,2} & 0 \\ Y_{1,2}^T & Y_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix},\quad Z = \begin{bmatrix} Z_{1,1} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

where $Y_{1,1},\ Z_{1,1} \in \mathbf{R}^{\mu_1\times\mu_1}$, $Y_{2,2} \in \mathbf{R}^{\mu_2\times\mu_2}$, $\mathrm{rank}(Y_{2,2}) = \mu_2$, $\mathrm{rank}(Z_{1,1}) = \mu_1$. Then,

$$\mathrm{trace}(X^{(+)}Z) = \mathrm{trace}(Z_{1,1}),$$

and

$$
\begin{aligned}
&\text{trace}((X+Y)^{(+)}Z)\\
=&\text{trace}\left(\begin{bmatrix} I+Y_{1,1} & Y_{1,2} \\ Y_{1,2}^T & Y_{2,2}\end{bmatrix}^{-1}\begin{bmatrix} Z_{1,1} & 0 \\ 0 & 0\end{bmatrix}\right)\\
=&\text{trace}\left(\begin{bmatrix} I+Y_{1,1} & Y_{1,2} \\ Y_{1,2}^T & Y_{2,2}\end{bmatrix}^{-1}\left(\begin{bmatrix} Z_{1,1}+Y_{1,1}Z_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}-\begin{bmatrix} Y_{1,1}Z_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}\right)\right)\\
=&\text{trace}\left(\begin{bmatrix} I+Y_{1,1} & Y_{1,2} \\ Y_{1,2}^T & Y_{2,2}\end{bmatrix}^{-1}\left(\begin{bmatrix} I+Y_{1,1} & Y_{1,2} \\ Y_{1,2}^T & Y_{2,2}\end{bmatrix}\begin{bmatrix} Z_{1,1} & 0 \\ 0 & 0\end{bmatrix}-\begin{bmatrix} Y_{1,1}Z_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}\right)\right)\\
=&\text{trace}\left(\begin{bmatrix} Z_{1,1} & 0 \\ 0 & 0\end{bmatrix}-\begin{bmatrix} I+Y_{1,1} & Y_{1,2} \\ Y_{1,2}^T & Y_{2,2}\end{bmatrix}^{-1}\begin{bmatrix} Y_{1,1}Z_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}\right)\\
=&\text{trace}(Z_{1,1})-\text{trace}((I+Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)^{-1}(Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)Z_{1,1}),
\end{aligned}
$$

where the last equality is due to the fact that

$$
\begin{aligned}
&\text{trace}\left(\begin{bmatrix} I+Y_{1,1} & Y_{1,2} \\ Y_{1,2}^T & Y_{2,2}\end{bmatrix}^{-1}\begin{bmatrix} Y_{1,1}Z_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}\right)\\
=&\text{trace}\left\{\left(\begin{bmatrix} I & Y_{1,2}Y_{2,2}^{-1} \\ & I\end{bmatrix}\begin{bmatrix} I+Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T & \\ & Y_{2,2}\end{bmatrix}\begin{bmatrix} I & \\ Y_{2,2}^{-1}Y_{1,2}^T & I\end{bmatrix}\right)^{-1}\right.\\
&\qquad\left.\times\begin{bmatrix} Y_{1,1}Z_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}\right\}\\
=&\text{trace}\left(\begin{bmatrix} I+Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T & \\ & Y_{2,2}\end{bmatrix}^{-1}\begin{bmatrix} I & -Y_{1,2}Y_{2,2}^{-1} \\ & I\end{bmatrix}\begin{bmatrix} Y_{1,1}Z_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}\right.\\
&\qquad\left.\times\begin{bmatrix} I & \\ -Y_{2,2}^{-1}Y_{1,2}^T & I\end{bmatrix}\right)\\
=&\text{trace}\left(\begin{bmatrix} (I+Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)^{-1} & \\ & Y_{2,2}^{-1}\end{bmatrix}\begin{bmatrix} Y_{1,1}Z_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^TZ_{1,1} & 0 \\ Y_{1,2}^TZ_{1,1} & 0\end{bmatrix}\right)\\
=&\text{trace}((I+Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)^{-1}(Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)Z_{1,1}). \qquad (3.1)
\end{aligned}
$$

Note that Y is positive semi-definite, so $Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T$, the schur complement of $Y_{2,2}$, is positive semi-definite. In addition, $(I+Y_{1,1}-Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)^{-1}$ and $Z_{1,1}$ are

positive definite, thus,

$$\text{trace}((I + Y_{1,1} - Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)^{-1}(Y_{1,1} - Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)Z_{1,1}) \geq 0.$$

Hence, we have

$$\text{trace}((X+Y)^{(+)}Z) \leq \text{trace}(X^{(+)}Z),$$

and

$$\begin{aligned}
& \text{trace}((X+Y)^{(+)}Z) = \text{trace}(X^{(+)}Z) \\
\Leftrightarrow\ & \text{trace}((I + Y_{1,1} - Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)^{-1}(Y_{1,1} - Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)Z_{1,1}) = 0 \\
\Leftrightarrow\ & (I + Y_{1,1} - Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)^{-1}(Y_{1,1} - Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T)Z_{1,1} = 0 \\
\Leftrightarrow\ & Y_{1,1} - Y_{1,2}Y_{2,2}^{-1}Y_{1,2}^T = 0 \\
\Leftrightarrow\ & \text{rank}(Y) = \text{rank}(Y_{2,2}) = \mu_2 \\
\Leftrightarrow\ & \text{rank}\begin{bmatrix} X & Y \end{bmatrix} = \text{rank}(X) + \text{rank}(Y).
\end{aligned}$$

□

Remark 3.1. *For the general X, Y and Z, they can be reduced to the special form in Lemma 3.1 as follows: Let the eigen-decomposition of X be*

$$U^T X U = \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix},$$

where $\Lambda \in \mathbf{R}^{\mu_1 \times \mu_1}$, $\mu_1 = \text{rank}(X)$. Since

$$\text{rank}\begin{bmatrix} X & Z \end{bmatrix} = \text{rank}(X) = \text{rank}(Z),$$

it is easy to show that

$$U^T Z U = \begin{bmatrix} \hat{Z} & 0 \\ 0 & 0 \end{bmatrix},$$

where $\hat{Z} \in \mathbf{R}^{\mu_1 \times \mu_1}$. Denote

$$\hat{Y} = \begin{bmatrix} \hat{Y}_{1,1} & \hat{Y}_{1,2} \\ \hat{Y}_{2,1} & \hat{Y}_{2,2} \end{bmatrix} =: U^T Y U,$$

where $\hat{Y}_{1,1} \in \mathbf{R}^{\mu_1 \times \mu_1}$. *Let the eigen-decomposition of* $\hat{Y}_{2,2}$ *be*

$$\hat{Y}_{2,2} = V \begin{bmatrix} Y_{2,2} & \\ & 0 \end{bmatrix} V^T,$$

where $Y_{2,2} \in \mathbf{R}^{\mu_2 \times \mu_2}$ *is nonsingular. Thus,*

$$\begin{bmatrix} I & \\ & V \end{bmatrix}^T \hat{Y} \begin{bmatrix} I & \\ & V \end{bmatrix} = \begin{bmatrix} \hat{Y}_{1,1} & \tilde{Y}_{1,2} & 0 \\ \tilde{Y}_{2,1} & Y_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Let

$$\Phi = U \begin{bmatrix} \Lambda^{-\frac{1}{2}} & \\ & V \end{bmatrix},$$

by calculating, we have

$$\Phi^T X \Phi = \begin{bmatrix} I_{\mu_1} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \Phi^T Y \Phi = \begin{bmatrix} Y_{1,1} & Y_{1,2} & 0 \\ Y_{1,2}^T & Y_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \Phi^T Z \Phi = \begin{bmatrix} Z_{1,1} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Lemma 3.2. *When* $\operatorname{rank}(A) = n$, *the following condition holds:*

$$\operatorname{rank}(S_t) = \operatorname{rank}(S_b) + \operatorname{rank}(S_w).$$

Proof. According to (1.4) and the definitions of H_w, H_b and H_t in (1.3), when $\operatorname{rank}(A) = n$,

$$\operatorname{rank}(S_t) = \operatorname{rank}(H_t) = n - 1,$$
$$\operatorname{rank}(S_w) = \operatorname{rank}(H_w) = n - k,$$
$$\operatorname{rank}(S_b) = \operatorname{rank}(H_b) = k - 1,$$

thus, it holds that

$$\operatorname{rank}(S_t) = \operatorname{rank}(S_b) + \operatorname{rank}(S_w).$$

□

Denote

$$\mathcal{E} = \frac{1}{n} e e^T, \quad \mathcal{E}_i = \frac{1}{n_i} e_i e_i^T, \quad i = 1, \cdots, k,$$

where

$$e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^n, \quad e_i = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^{n_i}, \quad i = 1, \cdots, k.$$

Then we have the following new expression of scatter matrices.

Lemma 3.3.

$$S_t = A(I - \mathcal{E})A^T,$$

$$S_b = A\left(\begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} - \mathcal{E}\right)A^T, \quad S_w = A\left(I - \begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix}\right)A^T. \tag{3.2}$$

Proof. By the definition of scatter matrices in (1.1), we have

$$\begin{aligned} S_t &= \sum_{i=1}^{n}(a_i - c)(a_i - c)^T \\ &= \sum_{i=1}^{n}(a_i - \frac{1}{n}Ae)(a_i - \frac{1}{n}Ae)^T \\ &= \sum_{i=1}^{n} a_i a_i^T - \frac{1}{n}(\sum_{i=1}^{n} a_i)e^T A^T - \frac{1}{n}Ae\sum_{i=1}^{n} a_i^T + \frac{1}{n}Aee^T A^T \\ &= AA^T - \frac{1}{n}Aee^T A^T \\ &= A(I - \mathcal{E})A^T, \end{aligned}$$

$$\begin{aligned} S_b &= \sum_{i=1}^{k} n_i(c_i - c)(c_i - c)^T \\ &= \sum_{i=1}^{k} n_i(\frac{1}{n_i}\mathcal{A}_i e_i - \frac{1}{n}Ae)(\frac{1}{n_i}\mathcal{A}_i e_i - \frac{1}{n}Ae)^T \\ &= \sum_{i=1}^{k}(\frac{1}{n_i}\mathcal{A}_i e_i e_i^T \mathcal{A}_i^T - \frac{1}{n}\mathcal{A}_i e_i e^T A^T - \frac{1}{n}Aee_i^T \mathcal{A}_i^T + \frac{n_i}{n^2}Aee^T A^T) \\ &= \sum_{i=1}^{k} \mathcal{A}_i \mathcal{E}_i \mathcal{A}_i^T - \frac{1}{n}Aee^T A^T \\ &= A\left(\begin{bmatrix} \mathcal{E}_1 & & \\ & \cdots & \\ & & \mathcal{E}_k \end{bmatrix} - \mathcal{E}\right)A^T, \end{aligned}$$

and

$$S_w = S_t - S_b = A\left(I - \begin{bmatrix} \mathcal{E}_1 & & \\ & \cdots & \\ & & \mathcal{E}_k \end{bmatrix}\right) A^T$$

directly follows from (1.2). □

Note that

$$I - \mathcal{E}, \quad \begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} - \mathcal{E}, \quad I - \begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix},$$

are orthogonal projections in $\mathbf{R}^n$. Let $\mathcal{R}_t$, $\mathcal{R}_b$ and $\mathcal{R}_w$ be the range spaces of the above orthogonal projections, respectively. It can be shown that $\mathcal{R}_t = \mathcal{R}_b \oplus \mathcal{R}_w$ with

$$\dim(\mathcal{R}_t) = n - 1, \quad \dim(\mathcal{R}_b) = k - 1, \quad \dim(\mathcal{R}_w) = n - k.$$

We now devise an orthogonal basis in $\mathbf{R}^n$ containing partitions that span the subspaces $\mathcal{R}_b$ and $\mathcal{R}_w$. Define Householder transformations

$$\mathcal{H}_i = I - \left(\begin{bmatrix} 1 - \sqrt{n_i} \\ 1 \\ \vdots \\ 1 \end{bmatrix} / \sqrt{n_i - \sqrt{n_i}}\right)\left(\begin{bmatrix} 1 - \sqrt{n_i} \\ 1 \\ \vdots \\ 1 \end{bmatrix} / \sqrt{n_i - \sqrt{n_i}}\right)^T, \quad i = 1, \cdots, k,$$

and

$$\mathcal{H} = I - \left(\begin{bmatrix} \sqrt{n_1} - \sqrt{n} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} / \sqrt{n - \sqrt{n n_1}}\right)\left(\begin{bmatrix} \sqrt{n_1} - \sqrt{n} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} / \sqrt{n - \sqrt{n n_1}}\right)^T.$$

Notice that matrices $\mathcal{H}_i$ $(i = 1, \cdots, k)$ and $\mathcal{H}$ are orthogonal that satisfy

$$\mathcal{H}_i = \mathcal{H}_i^T \ (i = 1, \cdots, k), \quad \mathcal{H} = \mathcal{H}^T,$$

$$
\begin{aligned}
&\mathcal{H}_i^T(e_i/\sqrt{n_i}) \\
&= \left(\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} /\sqrt{n_i} \right) - \left(\begin{bmatrix} 1-\sqrt{n_i} \\ 1 \\ \vdots \\ 1 \end{bmatrix} /\sqrt{n_i - \sqrt{n_i}} \right) \left(\frac{n_i - \sqrt{n_i}}{\sqrt{n_i}\sqrt{n_i - \sqrt{n_i}}} \right) = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}
\end{aligned}
$$

for $i = 1, \cdots, k$, and

$$
\begin{aligned}
&\mathcal{H}^T \left(\begin{bmatrix} \sqrt{n_1} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} /\sqrt{n} \right) \\
&= \left(\begin{bmatrix} \sqrt{n_1} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} /\sqrt{n} \right) - \left(\begin{bmatrix} \sqrt{n_1}-\sqrt{n} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} /\sqrt{n - \sqrt{nn_1}} \right) \left(\frac{n - \sqrt{nn_1}}{\sqrt{n}\sqrt{n - \sqrt{nn_1}}} \right) \\
&= \left(\begin{bmatrix} \sqrt{n_1} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} /\sqrt{n} \right) - \left(\begin{bmatrix} \sqrt{n_1}-\sqrt{n} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} /\sqrt{n} \right) = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.
\end{aligned}
$$

Let $\mathcal{P}$ be the permutation matrix obtained by exchanging the $(\sum_{j=1}^{i-1} n_j + 1)$-th column and the i-th column (for $i = 2, \cdots, k$) of the $n \times n$ identity matrix, but otherwise leaving the order of the remaining columns unchanged. Consequently,

$$
\begin{aligned}
&\left(\begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix} \mathcal{P} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix} \right)^T \begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} \left(\begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix} \mathcal{P} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix} \right) \\
&= \begin{bmatrix} \mathcal{H}^T & \\ & I \end{bmatrix} \mathcal{P}^T \begin{bmatrix} \mathcal{H}_1^T \mathcal{E}_1 \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k^T \mathcal{E}_k \mathcal{H}_k \end{bmatrix} \mathcal{P} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix} \\
&= \begin{bmatrix} \mathcal{H}^T & \\ & I \end{bmatrix} \begin{bmatrix} I_k & \\ & 0 \end{bmatrix} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix} = \begin{bmatrix} I_k & \\ & 0 \end{bmatrix},
\end{aligned}
$$

and

$$\left(\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}\right)^T\mathcal{E}\left(\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}\right)$$

$$=\frac{1}{n}\begin{bmatrix}\mathcal{H}^T & \\ & I\end{bmatrix}\mathcal{P}^T\begin{bmatrix}\mathcal{H}_1^Te_1\\ \vdots\\ \mathcal{H}_k^Te_k\end{bmatrix}\begin{bmatrix}e_1^T\mathcal{H}_1 & \cdots & e_k^T\mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}$$

$$=\frac{1}{n}\begin{bmatrix}\mathcal{H}^T & \\ & I\end{bmatrix}\begin{bmatrix}\sqrt{n_1}\\ \vdots\\ \sqrt{n_k}\\ 0\end{bmatrix}\begin{bmatrix}\sqrt{n_1} & \cdots & \sqrt{n_k} & 0\end{bmatrix}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}$$

$$=\begin{bmatrix}1 & \\ & 0_{n-1}\end{bmatrix},$$

which give that

$$\left(\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}\right)^T(I-\mathcal{E})\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}=\begin{bmatrix}0_1 & \\ & I_{n-1}\end{bmatrix}, \tag{3.3}$$

$$\left(\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}\right)^T\left(I-\begin{bmatrix}\mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k\end{bmatrix}\right)\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}$$

$$=\begin{bmatrix}0_k & \\ & I_{n-k}\end{bmatrix}, \tag{3.4}$$

and

$$\left(\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}\right)^T\left(\begin{bmatrix}\mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k\end{bmatrix}-\mathcal{E}\right)\begin{bmatrix}\mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k\end{bmatrix}\mathcal{P}\begin{bmatrix}\mathcal{H} & \\ & I\end{bmatrix}$$

$$=\begin{bmatrix}0_1 & & \\ & I_{k-1} & \\ & & 0_{n-k}\end{bmatrix}. \tag{3.5}$$

Now, we have another formation of scatter matrices S_t, S_b and S_w.

Corollary 3.4. *Denote*

$$\begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix} := A \begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix} \mathcal{P} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix}, \tag{3.6}$$

where $A_1 \in \mathbf{R}^{m\times 1}$, $A_2 \in \mathbf{R}^{m\times(k-1)}$, *and* $A_3 \in \mathbf{R}^{m\times(n-k)}$. *Then*

$$S_b = A_2 A_2^T, \quad S_w = A_3 A_3^T, \quad S_t = \begin{bmatrix} A_2 & A_3 \end{bmatrix} \begin{bmatrix} A_2 & A_3 \end{bmatrix}^T. \tag{3.7}$$

Proof. Follows from (3.3)-(3.5), Lemma 3.3 and the definition of A_1, A_2 and A_3 in (3.6), we have

$$\begin{aligned} S_b &= A \left(\begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} - \mathcal{E} \right) A^T \\ &= \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix} \begin{bmatrix} 0_1 & & \\ & I_{k-1} & \\ & & 0_{n-k} \end{bmatrix} \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix}^T \\ &= A_2 A_2^T, \end{aligned}$$

$$\begin{aligned} S_w &= A \left(I - \begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} \right) A^T \\ &= \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix} \begin{bmatrix} 0 & \\ & I_{n-k} \end{bmatrix} \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix}^T = A_3 A_3^T, \end{aligned}$$

and

$$\begin{aligned} S_t &= A(I - \mathcal{E})A^T \\ &= \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix} \begin{bmatrix} 0 & \\ & I_{n-1} \end{bmatrix} \begin{bmatrix} A_1 & A_2 & A_3 \end{bmatrix}^T \\ &= \begin{bmatrix} A_2 & A_3 \end{bmatrix} \begin{bmatrix} A_2 & A_3 \end{bmatrix}^T. \end{aligned}$$

□

Next, we present a useful lemma of generalized singular value decomposition (GSVD) developed by Paige and Sauders [72], which was originally defined by Van Loan [88].

Lemma 3.5 (GSVD). *[72] Suppose two matrices $X \in \mathbf{R}^{\mu\times\nu_1}$ and $Y \in \mathbf{R}^{\mu\times\nu_2}$ are given, then there exist orthogonal matrices $U \in \mathbf{R}^{\nu_1\times\nu_1}$ and $V \in \mathbf{R}^{\nu_2\times\nu_2}$ and a nonsingular matrix $\Phi \in \mathbf{R}^{\mu\times\mu}$, such that*

$$U^T X^T \Phi = \begin{bmatrix} \Sigma & 0 \end{bmatrix}, \quad V^T Y^T \Phi = \begin{bmatrix} \Omega & 0 \end{bmatrix},$$

where

$$\Sigma = \begin{bmatrix} I_p & & \\ & \Theta & \\ & & 0_{(\nu_1-p-s)\times(\gamma-p-s)} \end{bmatrix}, \quad \Omega = \begin{bmatrix} 0_{(\nu_2-\gamma+p)\times p} & & \\ & \Xi & \\ & & I_{\gamma-p-s} \end{bmatrix},$$

and $\Sigma^T\Sigma + \Omega^T\Omega = I_\gamma$, where $\gamma = \operatorname{rank}\begin{bmatrix} X & Y \end{bmatrix}$, $p = \gamma - \operatorname{rank}(Y)$, and $s = \operatorname{rank}(X) + \operatorname{rank}(Y) - \gamma$. Diagonal matrices

$$\Theta = \begin{bmatrix} \theta_1 & & \\ & \ddots & \\ & & \theta_s \end{bmatrix}, \quad \Xi = \begin{bmatrix} \xi_1 & & \\ & \ddots & \\ & & \xi_s \end{bmatrix},$$

satisfy

$$1 > \theta_1 \geq \cdots \geq \theta_s > 0, \quad 0 < \xi_1 \leq \cdots \leq \xi_s < 1,$$

and

$$\theta_i^2 + \xi_i^2 = 1,$$

for $i = 1, \cdots, s$.

Lemma 3.6. *When $\operatorname{rank}(A) = n$, for $l \geq k-1$,*

$$\operatorname{trace}(S_t^{(+)} S_b) = \max_{G\in\mathbf{R}^{m\times l}} \operatorname{trace}((G^T S_t G)^{(+)} G^T S_b G) = k-1.$$

Proof. By adopting GSVD in Lemma 3.5 to H_b and H_w, we have

$$\Phi^T S_b \Phi = \Phi^T H_b U U^T H_b^T \Phi = \begin{bmatrix} \Sigma^T \Sigma & \\ & 0_{m-\gamma} \end{bmatrix}$$
$$= \begin{bmatrix} I_p & & & \\ & \Theta^2 & & \\ & & 0_{\gamma-p-s} & \\ & & & 0_{m-\gamma} \end{bmatrix},$$

and

$$\Phi^T S_w \Phi = \Phi^T H_w V V^T H_w^T \Phi = \begin{bmatrix} \Omega^T \Omega & \\ & 0_{m-\gamma} \end{bmatrix}$$
$$= \begin{bmatrix} 0_p & & & \\ & \Xi^2 & & \\ & & I_{\gamma-p-s} & \\ & & & 0_{m-\gamma} \end{bmatrix},$$

where, as defined in Lemma 3.5, $\Phi \in \mathbf{R}^{m \times m}$ is nonsingular, $U \in \mathbf{R}^{k \times k}$ and $V \in \mathbf{R}^{n \times n}$ are orthogonal, $\Sigma \in \mathbf{R}^{\gamma \times \gamma}$, $\Omega \in \mathbf{R}^{\gamma \times \gamma}$, $\Theta \in \mathbf{R}^{s \times s}$ and $\Xi \in \mathbf{R}^{s \times s}$ are diagonal, with $\gamma = \text{rank} \begin{bmatrix} H_b & H_w \end{bmatrix} = \text{rank}(S_t)$, $p = \gamma - \text{rank}(H_w)$, $s = \text{rank}(H_b) + \text{rank}(H_w) - \gamma$. In addition,

$$\Phi^T S_t \Phi = \Phi^T S_b \Phi + \Phi^T S_w \Phi = \begin{bmatrix} I_\gamma & \\ & 0_{m-\gamma} \end{bmatrix}.$$

Furthermore, when $\text{rank}(A) = n$, by Lemma 3.2,

$$\text{rank}(\Phi^T S_t \Phi) = \text{rank}(\Phi^T S_b \Phi) + \text{rank}(\Phi^T S_w \Phi)$$

holds, which results in $s = 0$ and $p = \text{rank}(H_b) = \text{rank}(S_b) = k - 1$. Thus,

$$\Phi^T S_b \Phi = \begin{bmatrix} I_{k-1} & \\ & 0_{m-k+1} \end{bmatrix},$$

and

$$\text{trace}(S_t^{(+)} S_b) = \text{trace}((\Phi^T S_t \Phi)^{(+)} \Phi^T S_b \Phi) = k - 1.$$

For any $G \in \mathbf{R}^{m\times l}$, we have

$$G^T S_t G = G^T \Phi^{-T} \Phi^T S_t \Phi \Phi^{-1} G = \mathcal{G}^T \begin{bmatrix} I_\gamma & \\ & 0_{m-\gamma} \end{bmatrix} \mathcal{G},$$

$$G^T S_b G = G^T \Phi^{-T} \Phi^T S_b \Phi \Phi^{-1} G = \mathcal{G}^T \begin{bmatrix} I_{k-1} & \\ & 0_{m-k+1} \end{bmatrix} \mathcal{G},$$

where $\mathcal{G} = \Phi^{-1} G$. Let

$$\mathcal{G} = \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \\ \mathcal{G}_3 \end{bmatrix} \in \mathbf{R}^{m\times l}$$

be the partition of $\mathcal{G}$ so that $\mathcal{G}_1 \in \mathbf{R}^{(k-1)\times l}$, $\mathcal{G}_2 \in \mathbf{R}^{(\gamma-k+1)\times l}$ and $\mathcal{G}_3 \in \mathbf{R}^{(m-\gamma)\times l}$. It follows that

$$G^T S_t G = \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}^T \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix} = \mathcal{G}_1^T \mathcal{G}_1 + \mathcal{G}_2^T \mathcal{G}_2, \quad G^T S_b G = \mathcal{G}_1^T \mathcal{G}_1.$$

Hence, by Lemma 3.1, we have

$$\begin{aligned} \operatorname{trace}((G^T S_t G)^{(+)} G^T S_b G) &= \operatorname{trace}((\mathcal{G}_1^T \mathcal{G}_1 + \mathcal{G}_2^T \mathcal{G}_2)^{(+)} \mathcal{G}_1^T \mathcal{G}_1) \\ &\le \operatorname{trace}((\mathcal{G}_1^T \mathcal{G}_1)^{(+)} \mathcal{G}_1^T \mathcal{G}_1) \\ &\le k-1, \end{aligned}$$

when $l \ge k-1$. Let $\mathcal{G}_2 = 0$, $\mathcal{G}_3 = 0$ and $\mathcal{G}_1$ be any full row rank matrix, then $G = \Phi\mathcal{G}$ is the one such that

$$\operatorname{trace}((G^T S_t G)^{(+)} G^T S_b G) = \operatorname{trace}(S_t^{(+)} S_b) = k-1.$$

By now, we have proved that when $\operatorname{rank}(A) = n$ there is a matrix G such that $\operatorname{trace}((G^T S_t G)^{(+)} G^T S_b G)$ achieves its maximum $\operatorname{trace}(S_t^{(+)} S_b) = k-1$. □

3.2 A New, Efficient and Simple LDA (LDA/QR)

In this section, a new, efficient and simple algorithm to solve the linear discriminant analysis problem

$$G = \arg \max_{G\in\mathbf{R}^{m\times l}} \operatorname{trace}((G^T S_t G)^{(+)} G^T S_b G) \tag{3.8}$$

is proposed, under a mild condition that $\text{rank}(A) = n$. Relationship of this new implementation of LDA, called LDA/QR, to the eigen-decomposition of $S_t^{(+)}S_b$ is derived, and a comparison of LDA/QR with ULDA/QA [22] is studied.

3.2.1 New Implementation of LDA

Lemma 3.7. *Denote*

$$E = \begin{bmatrix} e_1 & & \\ & \ddots & \\ & & e_k \end{bmatrix} \in \mathbf{R}^{n\times k}$$

as the indicator matrix, then any solution $G \in \mathbf{R}^{m\times k}$ of linear system

$$A^T G = E, \tag{3.9}$$

satisfies $\text{rank}(G) = k$, $G^T S_w G = 0$, *and* $G^T S_t G = G^T S_b G$.

Proof. For any $G \in \mathbf{R}^{m\times k}$ satisfies (3.9), we have

$$\text{rank}(G) \geq \text{rank}(A^T G) = \text{rank}(E) = k,$$

thus, $\text{rank}(G) = k$. By Lemma 3.3, the new form of scatter matrices, it holds that

$$\begin{aligned}
G^T S_w G &= G^T A \left(I - \begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} \right) A^T G \\
&= E^T \left(I - \begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} \right) E \\
&= E^T E - E^T \begin{bmatrix} \frac{1}{\sqrt{n_1}} e_1 & & \\ & \ddots & \\ & & \frac{1}{\sqrt{n_k}} e_k \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{n_1}} e_1 & & \\ & \ddots & \\ & & \frac{1}{\sqrt{n_k}} e_k \end{bmatrix}^T E \\
&= \begin{bmatrix} n_1 & & \\ & \ddots & \\ & & n_k \end{bmatrix} - \begin{bmatrix} \sqrt{n_1} & & \\ & \ddots & \\ & & \sqrt{n_k} \end{bmatrix} \begin{bmatrix} \sqrt{n_1} & & \\ & \ddots & \\ & & \sqrt{n_k} \end{bmatrix}^T \\
&= 0.
\end{aligned}$$

Therefore, scatter matrices equality (1.4) implies that

$$G^T S_t G = G^T S_b G.$$

□

Theorem 3.8. *Assume* $\mathrm{rank}(A) = n$, *let* $G \in \mathbf{R}^{m\times k}$ *satisfy (3.9), then*

$$\mathrm{trace}((G^T S_t G)^{(+)} G^T S_b G) = k - 1,$$

i.e., G is an optimal solution of LDA (3.8). Furthermore, let $E \in \mathbf{R}^{n\times k}$ *be defined in Lemma 3.7 and the economic QR factorization of A be*

$$A = QR, \tag{3.10}$$

where $Q \in \mathbf{R}^{m\times n}$ *is column orthogonal and* $R \in \mathbf{R}^{n\times n}$ *is upper triangular and nonsingular, then*

$$G = QR^{-T}E, \tag{3.11}$$

is the minimum 2-norm solution of LDA (3.8).

Proof. Let $G \in \mathbf{R}^{m\times k}$ satisfy (3.9) then by Lemma 3.3, we have

$$\begin{aligned} G^T S_b G &= G^T A \left(\begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} - \mathcal{E} \right) A^T G \\ &= E^T \left(\begin{bmatrix} \mathcal{E}_1 & & \\ & \ddots & \\ & & \mathcal{E}_k \end{bmatrix} - \mathcal{E} \right) E \\ &= \begin{bmatrix} n_1 & & \\ & \ddots & \\ & & n_k \end{bmatrix} - \frac{1}{n} \begin{bmatrix} n_1 \\ \vdots \\ n_k \end{bmatrix} \begin{bmatrix} n_1 & \cdots & n_k \end{bmatrix}. \end{aligned}$$

It is easy to verify that $\mathrm{rank}(G^T S_b G) = k - 1$. Together with the results from Lemma 3.7, we yield

$$\mathrm{trace}((G^T S_t G)^{(+)} G^T S_b G) = k - 1.$$

Thus follows from Lemma 3.6, any G as a solution of linear system (3.9), is an optimal solution of LDA (3.8).

Under the condition $\text{rank}(A) = n$, R from QR factorization (3.10) is nonsingular, G defined in (3.11) is the minimum 2-norm solution of linear system (3.9), thereby it is an optimal solution of LDA (3.8). □

Theorem 3.8 leads to the following new implementation of LDA.

Algorithm 3.1. *(LDA/QR)*

Input: *Data matrix $A \in \mathbf{R}^{m\times n}$ of full column rank with cluster label, size of each class n_i $(i = 1, \cdots, k)$, cluster number k.*

Output: *Transformation matrix $G \in \mathbf{R}^{m\times k}$.*

Step 1. *Construct*

$$E = \begin{bmatrix} e_1 & & \\ & \ddots & \\ & & e_k \end{bmatrix} \in \mathbf{R}^{n\times k}, \quad e_i = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbf{R}^{n_i}, \quad i = 1, \cdots, k.$$

Step 2. *Compute the economic QR factorization of A as*

$$A = QR, \quad Q \in \mathbf{R}^{m\times n}, \quad R \in \mathbf{R}^{n\times n}.$$

Step 3. $G = Q(R^{-T}E)$.

We now examine the relationship between LDA/QR and the eigen-decomposition of $S_t^{(+)}S_b$.

Theorem 3.9. *Assume* $\text{rank}(A) = n$. *Then eigenspace of $S_t^{(+)}S_b$ corresponding to all nonzero eigenvalues is contained in the range space of the minimal* 2*-norm solution G of linear system (3.9) .*

Proof. Denote

$$\begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix} = R \begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix} \mathcal{P} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix}, \tag{3.12}$$

where $R_1 \in \mathbf{R}^{m\times 1},\ R_2 \in \mathbf{R}^{m\times(k-1)}$, and $R_3 \in \mathbf{R}^{m\times(n-k)}$. Let the QR factorization of $\begin{bmatrix} R_2 & R_3 \end{bmatrix}$ be

$$\begin{bmatrix} R_1 & R_2 \end{bmatrix} = \mathcal{Q} \begin{bmatrix} R_{12} & R_{13} \\ 0 & R_{23} \\ 0 & 0 \end{bmatrix},$$

where $\mathcal{Q} \in \mathbf{R}^{n\times n}$ is orthogonal, $R_{12} \in \mathbf{R}^{(k-1)\times(k-1)}$, and $R_{23} \in \mathbf{R}^{(n-k)\times(n-k)}$. Since $\text{rank}(A) = n$, we have

$$\text{rank}(R_{12}) = k-1,\ \text{rank}(R_{23}) = n-k.$$

Set

$$\mathcal{Q}^T R_1 = \begin{bmatrix} R_{11} \\ R_{21} \\ R_{31} \end{bmatrix},$$

where $R_{11} \in \mathbf{R}^{(k-1)\times 1},\ R_{21} \in \mathbf{R}^{(n-k)\times 1}$. $R_{31} \in \mathbf{R}^{1\times 1}$. Then

$$\begin{cases} S_b = (Q\mathcal{Q} \begin{bmatrix} R_{12} \\ 0 \\ 0 \end{bmatrix})(Q\mathcal{Q} \begin{bmatrix} R_{12} \\ 0 \\ 0 \end{bmatrix})^T, \\ S_t = (Q\mathcal{Q} \begin{bmatrix} R_{12} & R_{13} \\ 0 & R_{23} \\ 0 & 0 \end{bmatrix})(Q\mathcal{Q} \begin{bmatrix} R_{12} & R_{13} \\ 0 & R_{23} \\ 0 & 0 \end{bmatrix})^T, \end{cases}$$

so,

$$S_t^{(+)} S_b = Q\mathcal{Q} \begin{bmatrix} I & 0 & 0 \\ -R_{23}^{-T} R_{13}^T & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} (Q\mathcal{Q})^T,$$

i.e.,

$$S_t^{(+)} S_b Q\mathcal{Q} \begin{bmatrix} I \\ -R_{23}^{-T} R_{13}^T \\ 0 \end{bmatrix} = Q\mathcal{Q} \begin{bmatrix} I \\ -R_{23}^{-T} R_{13}^T \\ 0 \end{bmatrix}. \tag{3.13}$$

Hence, the columns of $Q\mathcal{Q}\begin{bmatrix} I \\ -R_{23}^{-T}R_{13}^T \\ 0 \end{bmatrix}$ span the eigen-space of $S^{(+)}S_b$ corresponding its all nonzero eigenvalues.

On the other hand,

$$R = \begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix}^T \mathcal{P}^T \begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix}^T$$
$$= \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & 0 & R_{23} \\ R_{31} & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix}^T \mathcal{P}^T \begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix}^T,$$

and so, the minimal 2-norm solution G of linear system (3.9), which is given in (3.11), satisfies

$$\begin{aligned} G =& QR^{-T}E \\ =& Q\mathcal{Q} \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & 0 & R_{23} \\ R_{31} & 0 & 0 \end{bmatrix}^{-T} \begin{bmatrix} \mathcal{H}^T \begin{bmatrix} \sqrt{n_1} & & \\ & \ddots & \\ & & \sqrt{n_k} \end{bmatrix} \\ & 0 \end{bmatrix} \\ =& Q\mathcal{Q} \begin{bmatrix} 0 & I \\ 0 & -R_{23}^{-T}R_{13}^T \\ R_{31}^{-T} & 0 \end{bmatrix} \begin{bmatrix} I & (R_{21}^T R_{23}^{-T} R_{13}^T - R_{11}^T)R_{12}^{-T} \\ 0 & R_{12}^{-T} \end{bmatrix} \mathcal{H}^T \begin{bmatrix} \sqrt{n_1} & & \\ & \ddots & \\ & & \sqrt{n_k} \end{bmatrix}. \end{aligned} \tag{3.14}$$

Therefore, Theorem 3.9 follows directly from (3.13) and (3.14). □

3.2.2 Relationship to ULDA

Uncorrelated LDA (ULDA) was originally proposed in [55] for extracting feature vectors with uncorrelated attributes. Later on, the ULDA in [55] was generalized by Ye et. al. in [96, 98] for undersampled problems based on simultaneous

diagonalization of scatter matrices. The new criterion in [96, 98] is

$$G = \arg \max_{G \in \mathbf{R}^{m \times l},\ G^T S_t G = I} \operatorname{trace}((G^T S_t G)^{(+)}(G^T S_b G)). \tag{3.15}$$

In [98], two SVDs are involved in the new proposed ULDA algorithm, which is expensive for large and high-dimensional data sets. To further improve the performance of ULDA, Chu et al. [22] characterized all solutions of the optimization problem (3.15) and solve it by applying QR factorizations only. We denote this ULDA algorithm as ULDA/QR, and show it in Algorithm 3.2, for details, please refer to [22].

Note that the optimal solution obtained in Algorithm 3.2 is a specific solution of the optimization problem (3.15) with minimum Frobenius norm.

Algorithm 3.2. *(ULDA/QR)*

Input: *Data matrix* $A \in \mathbf{R}^{m \times n}$ *with cluster label.*

Output: *Transformation matrix* $G \in \mathbf{R}^{m \times l}$.

Step 1. *Compute economic QR factorization of data matrix* A *as*

$$A = UR, \quad U \in \mathbf{R}^{m \times n}, \quad R \in \mathbf{R}^{n \times n}.$$

Step 2. *Denote*

$$[R_1 \quad R_2 \quad R_3] := R \begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix} \mathcal{P} \begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix},$$

where $R_1 \in \mathbf{R}^{n \times 1}$, $R_2 \in \mathbf{R}^{n \times (k-1)}$, $R_3 \in \mathbf{R}^{n \times (n-k)}$, $\mathcal{P}$, $\mathcal{H}_i$ $(i = 1, \cdots, k)$ *and* $\mathcal{H}$ *are defined in section 3.1.*

Step 3. *Compute the economic QR factorization of* $[R_2 \quad R_3]$ *with column pivoting*

$$[R_2 \quad R_3] = \mathcal{Q}\mathcal{R},$$

where $\mathcal{Q} \in \mathbf{R}^{n \times \gamma}$, $\mathcal{R} \in \mathbf{R}^{\gamma \times (n-1)}$, $\gamma = \operatorname{rank}[R_2 \quad R_3]$.

Step 4. *Compute the economic QR factorization of* $\mathcal{R}^T$ *be*

$$\mathcal{R}^T = P^T \Delta^T, \quad P \in \mathbf{R}^{\gamma \times (n-1)}, \quad \Delta \in \mathbf{R}^{\gamma \times \gamma}.$$

Step 5. *Denote* $P_1 := P(:, 1:k-1)$ *and compute the economic QR factorization of* P_1 *with column pivoting be*

$$P_1 = V\Pi,$$

where $V \in \mathbf{R}^{\gamma\times q}$, $\Pi \in \mathbf{R}^{q\times(k-1)}$, $q = \mathrm{rank}(P_1)$.

Step 6. *Solve the upper triangular linear system of equations*

$$\Delta^T Y = V.$$

Step 7. $G = U(\mathcal{Q}Y)$.

The relationship of LDA/QR to ULDA/QR is summarized in the following theorem.

Theorem 3.10. *Assume* $\mathrm{rank}(A) = n$, *let* $G \in \mathbf{R}^{m\times k}$ *be a solution of linear system (3.9), there exists a full column rank matrix*

$$\Psi = \begin{bmatrix} \sqrt{n_1} & & \\ & \ddots & \\ & & \sqrt{n_k} \end{bmatrix}^{-1} \mathcal{H}^T \begin{bmatrix} 0 \\ I_{k-1} \end{bmatrix} \in \mathbf{R}^{k\times(k-1)},$$

such that $G\Psi$ *is a solution of ULDA (3.15), where* $\mathcal{H} \in \mathbf{R}^{k\times k}$ *is a Householder transformation defined in section 3.1.*

Proof. If G is a solution of linear system (3.9), by Lemma 3.7 and using the proof in Theorem 3.8, we have

$$\begin{aligned} G^T S_t G =& G^T S_b G \\ =& \begin{bmatrix} n_1 & & \\ & \ddots & \\ & & n_k \end{bmatrix} - \frac{1}{n} \begin{bmatrix} n_1 \\ \vdots \\ n_k \end{bmatrix} \begin{bmatrix} n_1 & \cdots & n_k \end{bmatrix} \\ =& \begin{bmatrix} \sqrt{n_1} & & \\ & \ddots & \\ & & \sqrt{n_k} \end{bmatrix} \left(I - \frac{1}{n} \begin{bmatrix} \sqrt{n_1} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} \begin{bmatrix} \sqrt{n_1} & \cdots & \sqrt{n_k} \end{bmatrix} \right) \begin{bmatrix} \sqrt{n_1} & & \\ & \ddots & \\ & & \sqrt{n_k} \end{bmatrix}, \end{aligned}$$

furthermore,

$$\begin{aligned}
\Psi^T G^T S_t G \Psi &= \Psi^T G^T S_b G \Psi \\
&= \begin{bmatrix} 0 & I_{k-1} \end{bmatrix} \mathcal{H} \left(I - \frac{1}{n} \begin{bmatrix} \sqrt{n_1} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} \begin{bmatrix} \sqrt{n_1} \\ \vdots \\ \sqrt{n_k} \end{bmatrix}^T \right) \mathcal{H}^T \begin{bmatrix} 0 \\ I_{k-1} \end{bmatrix} \\
&= \begin{bmatrix} 0 & I_{k-1} \end{bmatrix} \left(I - \left(\mathcal{H} \begin{bmatrix} \sqrt{n_1} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} / \sqrt{n} \right) \left(\mathcal{H} \begin{bmatrix} \sqrt{n_1} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} / \sqrt{n} \right)^T \right) \begin{bmatrix} 0 \\ I_{k-1} \end{bmatrix} \\
&= \begin{bmatrix} 0 & I_{k-1} \end{bmatrix} \left(I - \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}^T \right) \begin{bmatrix} 0 \\ I_{k-1} \end{bmatrix} = I_{k-1}.
\end{aligned}$$

Therefore, $G\Psi$ is an optimal solution of ULDA (3.15). □

Theorem 3.10 implies that our new proposed solution G of LDA (3.8) is equivalent to the solution of ULDA (3.15) scaled by a full column rank matrix.

3.2.3 Complexity Analysis

We close this section by analyzing the time complexity of the batch algorithm LDA/QR as well as ULDA/QR [22].

Both of these two algorithms apply economic QR factorization rather than SVD or GSVD. As compared in [22], ULDA/QR is less expensive than another famous ULDA algorithm in [96, 98]. We summarize in Table 3.1, the computational cost of LDA/QR shown in Algorithm 3.1 and ULDA/QR [22] shown in Algorithm 3.2, where there are n samples of k classes in m dimension, γ is the rank of S_t and q is the rank of S_b. The cost for some matrix computation such as economic QR factorization and solving an upper triangular linear system are given in Appendix B. Note that when the training set is of full column rank which is satiesfied in many applications, $\gamma = \text{rank}(S_t) = n - 1$ and $q = \text{rank}(S_b) = k - 1$. For this case,

we can observe from Table 3.1 that the main cost of ULDA/QR and LDA/QR are $4mn^2+4n^3$ and $4mn^2-\frac{4}{3}n^3$, respectively, which implies that LDA/QR outperforms ULDA/QR and the gap is obvious when the sample size n is large.

Table 3.1: Computational complexity (flops) of algorithms ULDA/QR and LDA/QR

The computational cost of Algorithm ULDA/QR:	
Step 1:	$4mn^2 - \frac{4}{3}n^3$
Step 2:	$O(n^2)$
Step 3:	$2n(n-1)^2 - \frac{2}{3}(n-1)^3 + 4n(n-1)\gamma - 2\gamma^2(2n-1) + \frac{4}{3}\gamma^3$
Step 4:	$4(n-1)\gamma^2 - \frac{4}{3}\gamma^3$
Step 5:	$2\gamma(k-1)^2 - \frac{2}{3}(k-1)^3 + 4\gamma(k-1)q - 2q^2(\gamma+k-1) + \frac{4}{3}q^3$
Step 6:	$2mnq + 2n\gamma q + \gamma^2 q$

The computational cost of Algorithm LDA/QR:	
Step 1:	n
Step 2:	$4mn^2 - \frac{4}{3}n^3$
Step 3:	$2mnk + n^2k$

3.3 Incremental Implementation (ILDA/QR)

In this section, we study the incremental learning of LDA/QR proposed in section 3.2. We will adopt the following convention in rest of this section. For any variable X, its updated version after the insertion of new samples is denoted by $\tilde{X}$. For example, the data matrix A is changed to $\tilde{A}$, and the number, n_i, of elements in the i-th class is changed to $\tilde{n}_i$.

With the insertion of new data, the indicator matrix E, data matrix A and optimal transformation G will change accordingly. From the batch method LDA/QR presented in Algorithm 3.1, the incremental updating of LDA/QR proceeds in three steps:

- Updating of indicator matrix E;
- QR-updating of data matrix A;
- Updating of optimal transformation G.

3.3.1 Sequential Incremental Implementation

Let x be the new inserted sample, which belongs to the ℓ-th class. Without loss of generality, let us assume that we have data $a_1, \cdots, a_n$ from the 1st to the kth class, just before the new data is inserted. This can be done by switching class labels between different classes. In terms of the learning of batch algorithm in section 3.2, matrix E is closely related to the class of new inserted sample, so is G, therefore, the updating of matrix E and optimal transformation G are demonstrated in two distinct cases:

- x belongs to an existing class, i.e., $1 \leq \ell \leq k$;
- x belongs to a new class, i.e., $\ell > k$.

As will be seen later, the updated E or G are different for these two cases. While the updating of data matrix A is just to accommodate the new inserted sample, there is no class label involved in the QR-updating step. So the updating of the QR factorization of data matrix A is unified into one form.

Updating of E

This section will focus on the updating of the indicator matrix E defined in Lemma 3.7 for two different cases: the new inserted sample is from an existing class; the new inserted sample is from a new class.

I. Insertion of a new sample from an existing class

Before we update the indicator matrix E, the updating of three scatter matrices with the new expression in (3.2) is shown as follows:

Lemma 3.11. *When the new data sample x belongs to an existing class, i.e., $1 \le \ell \le k$, the updated scatter matrices are*

$$\tilde{S}_t = \tilde{A}(I - \tilde{\mathcal{E}})\tilde{A}^T,$$

$$\tilde{S}_b = \tilde{A}\left(\begin{bmatrix} \mathcal{E}_1 & & & & \\ & \ddots & & & \\ & & \tilde{\mathcal{E}}_\ell & & \\ & & & \ddots & \\ & & & & \mathcal{E}_k \end{bmatrix} - \tilde{\mathcal{E}}\right)\tilde{A}^T,\ \tilde{S}_w = \tilde{A}\left(I - \begin{bmatrix} \mathcal{E}_1 & & & & \\ & \ddots & & & \\ & & \tilde{\mathcal{E}}_\ell & & \\ & & & \ddots & \\ & & & & \mathcal{E}_k \end{bmatrix}\right)\tilde{A}^T$$

where $\tilde{A}$ is the updated data matrix, i.e.,

$$\tilde{A} = \begin{bmatrix} \mathcal{A}_1 & \cdots & \mathcal{A}_\ell & x & \mathcal{A}_{\ell+1} & \cdots & \mathcal{A}_k \end{bmatrix},$$

$\tilde{\mathcal{E}} = \frac{1}{\tilde{n}}\tilde{e}\tilde{e}^T$ *with* $\tilde{e} = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^T \in \mathbf{R}^{\tilde{n}}$, $\tilde{n} = n + 1$, *and* $\tilde{\mathcal{E}}_\ell = \frac{1}{\tilde{n}_\ell}\tilde{e}_\ell\tilde{e}_\ell^T$ *with* $\tilde{e}_\ell = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix}^T \in \mathbf{R}^{\tilde{n}_\ell}$, $\tilde{n}_\ell = n_\ell + 1$.

Proof. The proof is similar to the one in Lemma 3.3. □

Let P be the permutation matrix which is obtained by exchanging the i-th column and the $(i-1)$-th column of the $(n+1) \times (n+1)$ identity matrix, i is chosen from $n+1$ to $\sum_{j=1}^{\ell} n_j + 1$, then we have the following results.

Theorem 3.12. *When the new inserted sample x belongs to an existing class, i.e., $1 \le \ell \le k$, let*

$$\hat{A} = \begin{bmatrix} A & x \end{bmatrix}, \quad \hat{E} = \begin{bmatrix} E \\ z^T \end{bmatrix}, \tag{3.16}$$

where $z \in \mathbf{R}^k$ is a column vector with the ℓ-th element 1 and the others zero, then the updated data matrix $\tilde{A}$ and indicator matrx

$$\tilde{E} = \begin{bmatrix} e_1 & & & & \\ & \ddots & & & \\ & & \tilde{e}_\ell & & \\ & & & \ddots & \\ & & & & e_k \end{bmatrix}$$

satisfy

$$\tilde{A} = \hat{A}P, \quad \tilde{E} = P^T\hat{E}. \tag{3.17}$$

Assume $\tilde{A}$ or equivalently $\hat{A}$ is of full column rank, then

$$\tilde{A}^T\tilde{G} = \tilde{E} \Leftrightarrow \hat{A}^T\tilde{G} = \hat{E}, \tag{3.18}$$

and any $\tilde{G} \in \mathbf{R}^{n\times k}$ satisfying (3.18) is an optimal solution of LDA (3.8) with S_t and S_b being changed to $\tilde{S}_t$ and $\tilde{S}_b$, respectively.

Proof. It is easy to check that (3.17) holds according to the definition of P and (3.16). The equivalence of two linear systems in (3.18) directly follows from (3.17). For the last part, by the proof of Theorem 3.8, it is sufficient to show that

$$\operatorname{rank}(\tilde{G}^T\tilde{S}_b\tilde{G}) = k-1 \quad \text{and} \quad \tilde{G}^T\tilde{S}_w\tilde{G} = 0.$$

Let $\tilde{G}$ satisfy (3.18), by Lemma 3.11, we have the rank of matrix

$$\begin{aligned}\tilde{G}^T\tilde{S}_b\tilde{G} &= \tilde{E}^T\Big(\begin{bmatrix}\mathcal{E}_1 & & & & \\ & \ddots & & & \\ & & \tilde{\mathcal{E}}_\ell & & \\ & & & \ddots & \\ & & & & \mathcal{E}_k\end{bmatrix} - \tilde{\mathcal{E}}\Big)\tilde{E} \\ &= \begin{bmatrix} n_1 & & & & \\ & \ddots & & & \\ & & \tilde{n}_\ell & & \\ & & & \ddots & \\ & & & & n_k\end{bmatrix} - \frac{1}{\tilde{n}}\begin{bmatrix} n_1 \\ \vdots \\ \tilde{n}_\ell \\ \vdots \\ n_k\end{bmatrix}\begin{bmatrix} n_1 & \cdots & \tilde{n}_\ell & \cdots & n_k\end{bmatrix},\end{aligned}$$

is $k-1$, and

$$\tilde{G}^T\tilde{S}_w\tilde{G} = \tilde{E}^T\Big(I - \begin{bmatrix}\mathcal{E}_1 & & & & \\ & \ddots & & & \\ & & \tilde{\mathcal{E}}_\ell & & \\ & & & \ddots & \\ & & & & \mathcal{E}_k\end{bmatrix}\Big)\tilde{E} = \tilde{E}^T\tilde{E} - \tilde{E}^T\tilde{E} = 0.$$

From here we have proved that any solution of linear system (3.18) is an optimal transformation matrix of LDA (3.8). □

Remark 3.2. *In our experimental implementation for the incremental algorithm as will be seen later, the new presented data is inserted into the last column of the data matrix, which is the same as $\hat{A}$ exhibit. In the rest of this thesis, for the case that the new inserted sample is from an existing class, we use $\hat{A}$ and $\hat{E}$ given in (3.16) to represent the updated data matrix and the updated indicator matrix, respectively. For convenience, we still denote them as $\tilde{A}$ and $\tilde{E}$.*

II. Insertion of a new sample from a new class

For the case that the new sample is from a new class, the updating of scatter matrices: S_t, S_b and S_w is given in the following lemma, and the updating of indicator matrix E is shown in Theorem 3.14.

Lemma 3.13. *When the new data sample x belongs to a new class, i.e., $\ell > k$, the updated scatter matrices are*

$$\tilde{S}_t = \tilde{A}(I - \tilde{\mathcal{E}})\tilde{A}^T,$$

$$\tilde{S}_b = \tilde{A}(\begin{bmatrix} \mathcal{E}_1 & & & \\ & \ddots & & \\ & & \mathcal{E}_k & \\ & & & 1 \end{bmatrix} - \tilde{\mathcal{E}})\tilde{A}^T, \quad \tilde{S}_w = \tilde{A}(I - \begin{bmatrix} \mathcal{E}_1 & & & \\ & \ddots & & \\ & & \mathcal{E}_k & \\ & & & 1 \end{bmatrix})\tilde{A}^T,$$

where $\tilde{A} = \begin{bmatrix} A & x \end{bmatrix}$, *and* $\tilde{\mathcal{E}} = \frac{1}{\tilde{n}}\tilde{e}\tilde{e}^T$ *with* $\tilde{e} = \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \in \mathbf{R}^{\tilde{n}}$, $\tilde{n} = n + 1$.

Proof. The proof is similar to the one in Lemma 3.3. □

Theorem 3.14. *When the new inserted sample x belongs to a new class, i.e., $\ell > k$, $\tilde{k} = k + 1$, the updated E is*

$$\tilde{E} = \begin{bmatrix} E & \\ & 1 \end{bmatrix}. \tag{3.19}$$

Assume the new data matrix $\tilde{A}$ is of full column rank, then any $\tilde{G} \in \mathbf{R}^{n\times\tilde{k}}$ satisfying linear system

$$\tilde{A}^T\tilde{G} = \tilde{E}, \tag{3.20}$$

is an optimal solution of (3.8) with S_t and S_b being changed to $\tilde{S}_t$ and $\tilde{S}_b$, respectively.

Proof. The proof is similar to the proof of Theorem 3.8. □

QR-updating of A

In this section, we will analyze the updating of QR factorization of data matrix A under the condition that the data set containing the new inserted sample x is of full column rank. As shown in the above section for the updating of indicator matrix, the updated data matrix has the form

$$\tilde{A} = \begin{bmatrix} A & x \end{bmatrix}$$

either x belongs to an existing class or x is from a new class. Therefore, the QR-updating of A is integrated into one case.

Given the economic QR factorization $A = QR$, where $Q \in \mathbf{R}^{m\times n}$ and $R \in \mathbf{R}^{n\times n}$,

$$\tilde{A} = \begin{bmatrix} A & x \end{bmatrix} = \begin{bmatrix} QR & x \end{bmatrix} = \begin{bmatrix} Q & x \end{bmatrix} \begin{bmatrix} R & \\ & 1 \end{bmatrix}.$$

We seek vectors $q \in \mathbf{R}^m$, $r \in \mathbf{R}^n$ and a scalar α so that

$$\begin{bmatrix} Q & x \end{bmatrix} = \begin{bmatrix} Q & q \end{bmatrix} \begin{bmatrix} I & r \\ & \alpha \end{bmatrix},$$

with $Q^Tq = 0$ and $\|q\|_2 = 1$. By directly calculating, the last column is

$$x = Qr + q\alpha,$$

multiplying both sides by Q^T gives $r = Q^Tx$ and $q\alpha = x - Qr$.

If $x - Qr \neq 0$, $\alpha = \|x - Qr\|_2 = \sqrt{x^T x - r^T r}$ and $q = \frac{x-Qr}{\alpha}$, which result in

$$\begin{aligned}
\tilde{A} &= \begin{bmatrix} Q & x \end{bmatrix} \begin{bmatrix} R & \\ & 1 \end{bmatrix} \\
&= \begin{bmatrix} Q & q \end{bmatrix} \begin{bmatrix} I & r \\ & \alpha \end{bmatrix} \begin{bmatrix} R & \\ & 1 \end{bmatrix} \\
&= \begin{bmatrix} Q & q \end{bmatrix} \begin{bmatrix} R & r \\ & \alpha \end{bmatrix} \\
&= \tilde{Q}\tilde{R},
\end{aligned}$$

the updated QR factorization of $\tilde{A}$, where

$$\tilde{Q} = \begin{bmatrix} Q & q \end{bmatrix} \in \mathbf{R}^{m\times(n+1)}, \quad \tilde{R} = \begin{bmatrix} R & r \\ & \alpha \end{bmatrix} \in \mathbf{R}^{(n+1)\times(n+1)}. \tag{3.21}$$

If $x - Qr = 0$, then $\alpha = 0$,

$$\tilde{A} = \begin{bmatrix} Q & q \end{bmatrix} \begin{bmatrix} R & r \\ 0 & 0 \end{bmatrix} = \tilde{Q}\tilde{R}$$

is the updated QR factorization of $\tilde{A}$, where

$$\tilde{Q} = \begin{bmatrix} Q & q \end{bmatrix} \in \mathbf{R}^{m\times(n+1)}, \quad \tilde{R} = \begin{bmatrix} R & r \\ 0 & 0 \end{bmatrix} \in \mathbf{R}^{(n+1)\times(n+1)}.$$

Theoretically, any unit vector orthogonal to the range of Q could be used for q.

When $x - Qr = 0$, x is in the range space of Q, i.e., in the range space of A. Therefore, for the case that all data samples are linearly independent, this situation will not occur, that is, when $\text{rank}(\tilde{A}) = n + 1$, $\alpha \neq 0$.

Updating of G

When the new inserted data sample x belongs to an existing class ℓ, $1 \leq \ell \leq k$, from the definition of $\tilde{E}$ in (3.16), $\tilde{Q}$ and $\tilde{R}$ in (3.21), we have

$$\begin{aligned}
\tilde{G} &= \tilde{Q}\tilde{R}^{-T}\tilde{E} \\
&= \tilde{Q}\begin{bmatrix} R & r \\ & \alpha \end{bmatrix}^{-T}\tilde{E} \\
&= \tilde{Q}\begin{bmatrix} R^{-T} & \\ -\frac{1}{\alpha}r^T R^{-T} & \frac{1}{\alpha} \end{bmatrix}\begin{bmatrix} E \\ z^T \end{bmatrix} \\
&= \begin{bmatrix} Q & q \end{bmatrix}\begin{bmatrix} R^{-T}E \\ \frac{1}{\alpha}(z^T - r^T R^{-T}E) \end{bmatrix} \\
&= QR^{-T}E + \frac{1}{\alpha}(qz^T - qr^T R^{-T}E) \\
&= QR^{-T}E + \frac{1}{\alpha}(qz^T - qx^T QR^{-T}E) \\
&= G + \frac{1}{\alpha}(qz^T - qx^T G) \\
&= G + \frac{1}{\alpha}q(z - G^T x)^T.
\end{aligned}$$

When the new inserted data sample x belongs to a new class ℓ, $\ell > k$, use the expression of $\tilde{R}^{-T}$ given in the above equation, the definition of $\tilde{E}$ in (3.19) and $\tilde{Q}$ in (3.21), we have

$$\begin{aligned}
\tilde{G} &= \tilde{Q}\tilde{R}^{-T}\tilde{E} \\
&= \tilde{Q}\begin{bmatrix} R^{-T} & \\ -\frac{1}{\alpha}r^T R^{-T} & \frac{1}{\alpha} \end{bmatrix}\begin{bmatrix} E & \\ & 1 \end{bmatrix} \\
&= \begin{bmatrix} Q & q \end{bmatrix}\begin{bmatrix} R^{-T}E & \\ -\frac{1}{\alpha}r^T R^{-T}E & \frac{1}{\alpha} \end{bmatrix} \\
&= \begin{bmatrix} QR^{-T}E - \frac{1}{\alpha}qr^T R^{-T}E & \frac{1}{\alpha}q \end{bmatrix} \\
&= \begin{bmatrix} QR^{-T}E - \frac{1}{\alpha}qx^T QR^{-T}E & \frac{1}{\alpha}q \end{bmatrix} \\
&= \begin{bmatrix} G - \frac{1}{\alpha}qx^T G & \frac{1}{\alpha}q \end{bmatrix}.
\end{aligned}$$

Sequential Incremental Algorithm

With the above analysis, the incremental implementation of LDA/QR finally involves in two steps: The updating of orthonormal matrix Q from QR factorization of data matrix and the updating of optimal transformation G. We call this incremental LDA method as ILDA/QR. The algorithms of ILDA/QR for updating existing class and new class are shown in the following two separated algorithms, respectively.

Algorithm 3.3. *(ILDA/QR: Updating Existing Class)*

Input: *Orthonormal matrix $Q \in \mathbf{R}^{m\times n}$ from QR factorization of data matrix, optimal transformation $G \in \mathbf{R}^{m\times k}$, and new sample x from the ℓ-th class, $1 \le \ell \le k$.*

Output: *Updated $\tilde{Q}$ and $\tilde{G}$.*

Step 1. *Compute $r = Q^T x$, $\alpha = \sqrt{x^T x - r^T r}$, update*

$$\tilde{Q} = [Q \quad (x - Qr)/\alpha] .$$

Step 2. *Compute $r = -G^T x$, $r(\ell) = r(\ell) + 1$ and $r = r/\alpha$, update*

$$\tilde{G} = G + \tilde{Q}(:, n+1)r^T .$$

Algorithm 3.4. *(ILDA/QR: Updating New Class)*

Input: *Orthonormal matrix $Q \in \mathbf{R}^{m\times n}$ from QR factorization of data matrix, optimal transformation $G \in \mathbf{R}^{m\times k}$, and new sample x from the ℓ-th class, $\ell > k$.*

Output: *Updated $\tilde{Q}$ and $\tilde{G}$.*

Step 1. *Compute $r = Q^T x$ and $\alpha = \sqrt{x^T x - r^T r}$, update*

$$\tilde{Q} = [Q \quad (x - Qr)/\alpha] .$$

Step 2. *Let $r = \tilde{Q}(:, n+1)/\alpha$, update*

$$\tilde{G} = \begin{bmatrix} G - r(x^T G) & r \end{bmatrix} .$$

Complexity Analysis

In this section, we analyze the time and memory complexity of the proposed ILDA/QR algorithm.

In both our updating algorithms of ILDA/QR, the update of Q and G involves in several simple operations including multiplication between a matrix and a vector, and addition between matrices. Note that, there are no multiplication between matrices or matrix decomposition. Therefore, the complexity of these operations is at most $O(mn)$ for each update, where m is the dimension of the data and n is the sample size. We summarize the precise computational cost of ILDA/QR for one single insertion in Table 3.2, where k is the number of classes.

Table 3.2: Computational complexity (flops) of algorithm ILDA/QR

The computational cost of ILDA/QR (Updating existng class):
Step 1: $4mn + 4m + 2n$
Step 2: $5mk + k$

The computational cost of ILDA/QR (Updating new class):
Step 1: $4mn + 4m + 2n$
Step 2: $5mk + m$

To compare the efficiency of ILDA/QR with four existing incremental LDA algorithms: IDR/QR [102], ILDA/SSS [62, 63], LS-ILDA [69] and ICLDA [70] shown in Chapter 2, we present the main cost of these five algorithms in Table 3.3. We can observe from Table 3.3 that when the dataset contains limited classes, IDR/QR is very fast; the time complexity of LS-ILDA and ILDA/QR is linear in the number of points and linear in the dimension of dataset; the computational cost of ILDA/SSS and ICLDA are relatively high, especially when n is large. In addition, using the notations given in Chapter 2, those information that need to be kept in main memory for incremental learning is summarized in Table 3.4. In both Table 3.3 and Table 3.4, we assume for ILDA/SSS that the total scatter

matrix is nearly full column rank and the performance of ILDA/SSS approximate the performance of batch LDA.

Table 3.3: Main computational cost (flops) of algorithms IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR for a single insertion

Method	Time Complexity
IDR/QR	$2mk^2 + 91/3k^3$
ILDA/SSS	$2mn^2 + 12n^3$
LS-ILDA	$14mn + 7mk$
ICLDA	$2mn^2 + 20n^3$
ILDA/QR	$4mn + 5mk$

Table 3.4: Memory cost of algorithms IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR

Method	Data	Space Complexity
IDR/QR	$C,\ \mathcal{Q},\ \mathcal{R},\ W,\ N$	$mk + mk + k^2 + k^2 + k$
ILDA/SSS	$U_1,\ \Lambda_1,\ V_1,\ \Gamma_1$ $c,\ C,\ N$	$m(n-1) + (n-1) + m(k-1) + (k-1)$ $+m + mk + k$
LS-ILDA	$H_t,\ H_t^{(+)},\ \mathcal{Y},\ G,\ c,\ N$	$mn + mn + n + mk + m + k$
ICLDA	$H_b,\ H_w,\ H_w^d,\ H_t^d$ $Q^d,\ H_W, H_B$ $P_1,\ P_2$ $R_2,\ c,\ C,\ N$	$mk + mn + m(n-k)$ $+m(n-1) + m(n-1) + (n-1)n$ $+(n-1)k + (n-1)(n-k)$ $+(n-1)(k-1) + m + mk + k$
ILDA/QR	$Q\ , G$	$mn + mk$

3.3.2 Chunk Incremental Implementation

In this section, we study another case of the incremental learning of LDA/QR that the new samples are acquired in a chunk way. The same as we assumed in sequential ILDA/QR, we have data $A = \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix}$ from the 1st to the kth classes, just before the new data sets are inserted. Let $\mathcal{X} = \begin{bmatrix} x_1 & \cdots & x_s \end{bmatrix}$ be the

new inserted samples and ℓ_i be the class label of x_i, $i = 1, \cdots, s$, and assume that after the insertion of data $\mathcal{X}$ the new data matrix is of full column rank.

After incorporating $\mathcal{X}$, the new data matrix $\tilde{A} = \begin{bmatrix} A & \mathcal{X} \end{bmatrix}$ contains totally $\tilde{k}$ ($\tilde{k} \geq k$) classes of samples and the new indicator matrix has the form

$$\tilde{E} = \begin{bmatrix} E & 0 \\ & Z \end{bmatrix} \in \mathbf{R}^{(n+s)\times\tilde{k}}, \tag{3.22}$$

where $Z = \begin{bmatrix} z_1 & \cdots & z_s \end{bmatrix}^T \in \mathbf{R}^{s\times\tilde{k}}$ and $z_i \in \mathbf{R}^{\tilde{k}}$ ($i = 1, \cdots, s$) is a unit vector with the ℓ_i-th element 1.

Given the economic QR factorization of $A = QR$, where $Q \in \mathbf{R}^{m\times n}$ and $R \in \mathbf{R}^{n\times n}$, let the economic QR factorization of $\mathcal{X} - Q(Q^T\mathcal{X})$ be

$$\mathcal{X} - Q(Q^T\mathcal{X}) = \hat{Q}\hat{R},$$

where $\hat{Q} \in \mathbf{R}^{m\times s}$ is orthonormal and column orthogonal to Q and $\hat{R} \in \mathbf{R}^{s\times s}$ is upper triangular. It is easy to show that when $\tilde{A}$ is of full column rank, $\hat{R}$ is nonsingular. Therefore, the updated QR factorization of $\tilde{A}$ is

$$\begin{aligned}
\tilde{A} &= \begin{bmatrix} A & \mathcal{X} \end{bmatrix} \\
&= \begin{bmatrix} QR & \mathcal{X} \end{bmatrix} \\
&= \begin{bmatrix} Q & (I - QQ^T)\mathcal{X} \end{bmatrix} \begin{bmatrix} R & Q^T\mathcal{X} \\ & I \end{bmatrix} \\
&= \begin{bmatrix} Q & \hat{Q} \end{bmatrix} \begin{bmatrix} R & Q^T\mathcal{X} \\ & \hat{R} \end{bmatrix} \\
&= \tilde{Q}\tilde{R},
\end{aligned}$$

where

$$\tilde{Q} = \begin{bmatrix} Q & \hat{Q} \end{bmatrix} \in \mathbf{R}^{m\times(n+s)}, \quad \tilde{R} = \begin{bmatrix} R & Q^T\mathcal{X} \\ & \hat{R} \end{bmatrix} \in \mathbf{R}^{(n+s)\times(n+s)},$$

and $\tilde{R}$ is nonsingular.

Furthermore, the updated transformation matrix is

$$\begin{aligned}
\tilde{G} &= \tilde{Q}\tilde{R}^{-T}\tilde{E} \\
&= \tilde{Q}\begin{bmatrix} R & Q^T\mathcal{X} \\ & \hat{R} \end{bmatrix}^{-T}\tilde{E} \\
&= \tilde{Q}\begin{bmatrix} R^{-T} & \\ -\hat{R}^{-T}\mathcal{X}^T Q R^{-T} & \hat{R}^{-T} \end{bmatrix}\begin{bmatrix} E & 0 \\ & Z \end{bmatrix} \\
&= \begin{bmatrix} Q & \hat{Q} \end{bmatrix}\begin{bmatrix} R^{-T}\begin{bmatrix} E & 0 \end{bmatrix} \\ \hat{R}^{-T}Z^T - \hat{R}^{-T}\mathcal{X}^T Q R^{-T}\begin{bmatrix} E & 0 \end{bmatrix} \end{bmatrix} \\
&= QR^{-T}\begin{bmatrix} E & 0 \end{bmatrix} + \hat{Q}\hat{R}^{-T}Z - \hat{Q}\hat{R}^{-T}\mathcal{X}^T Q R^{-T}\begin{bmatrix} E & 0 \end{bmatrix} \\
&= \begin{bmatrix} G & 0 \end{bmatrix} + \hat{Q}\hat{R}^{-T}Z - \hat{Q}\hat{R}^{-T}\mathcal{X}^T\begin{bmatrix} G & 0 \end{bmatrix} \\
&= \begin{bmatrix} G - \hat{Q}(\hat{R}^{-T}(\mathcal{X}^T G)) & 0 \end{bmatrix} + \hat{Q}(\hat{R}^{-T}Z).
\end{aligned}$$

Based on the above analysis on the updating of $\tilde{E}$, QR factorization of $\tilde{A}$ and $\tilde{G}$, the algorithm for chunk ILDA/QR is summarized in the following.

Algorithm 3.5. *(Chunk ILDA/QR)*

Input: *Orthonormal matrix $Q \in \mathbf{R}^{m\times n}$ from QR factorization of data matrix, optimal transformation $G \in \mathbf{R}^{m\times k}$, and new samples $\mathcal{X} = [x_1 \ \cdots \ x_s]$ with class labels ℓ_i, $i = 1, \cdots, s$.*

Output: *Updated $\tilde{Q}$ and $\tilde{G}$.*

Step 1. *Construct $Z \in \mathbf{R}^{s\times\tilde{k}}$ by*

$$Z = [z_1 \ \cdots \ z_s]^T,$$

where

$$z_i^T = [0 \ \cdots \ \underset{\ell_i}{1} \ \cdots \ 0] \in \mathbf{R}^{\tilde{k}}.$$

Step 2. *Compute economic QR factorization*

$$\mathcal{X} - Q(Q^T\mathcal{X}) = \hat{Q}\hat{R},$$

where $\hat{Q} \in \mathbf{R}^{m\times s}$ and $\hat{R} \in \mathbf{R}^{s\times s}$. Update

$$\tilde{Q} = \begin{bmatrix} Q & \hat{Q} \end{bmatrix}.$$

Step 3. *Update*

$$\tilde{G} = \begin{bmatrix} G - \hat{Q}(\hat{R}^{-T}(\mathcal{X}^T G)) & 0 \end{bmatrix} + \hat{Q}(\hat{R}^{-T} Z).$$

Remark 3.3. *With the above chunk incremental updating algorithm and two sequential incremental updating schemes presented in Section 3.3.1, the incremental method ILDA/QR works as follows:*

- *For a given initial training dataset, use batch algorithm LDA/QR in Algorithm 3.1 to compute and save the orthonormal matrix Q from the economic QR factorization of the data matrix A and the transformation matrix G.*
- *When a new sample x is inserted, determine whether it is from an existing or a new class. If it is from an existing class, update the orthonormal matrix Q and the transformation G by applying Algorithm 3.3; otherwise update the orthonormal matrix Q and the transformation G by applying Algorithm 3.4.*
- *When a chunk of new samples $x_1,\ x_2, \cdots, x_s$ are inserted, update the column orthogonal matrix Q and the optimal transformation G by applying Algorithm 3.5.*
- *The above procedure is repeated until all points are considered. Then the optimal transformation is the final updated $\tilde{G}$.*

The ILDA/QR algorithm follows the general criteria of the incremental learning algorithm [76]:

- It is able to learn additional information from new samples;
- It does not need to process the original data;
- It preserves the previously acquired knowledge;
- It can accommodate new classes that may be introduced with new data.

We summarize the computational cost of chunk ILDA/QR for the insertion of a chunk of s samples in Table 3.5, where $\tilde{k}$ is the number of classes of the new data matrix.

Table 3.5: Computational complexity (flops) of algorithm Chunk ILDA/QR

The computational cost of Chunk ILDA/QR:	
Step 2:	$4mns + ms + 4ms^2 - \frac{4}{3}s^3$
Step 3:	$4msk + mk + s^2k + 2ms\tilde{k} + s^2\tilde{k}$

To compare the efficiency of chunk ILDA/QR with IDR/QR [102], ILDA/SSS [62, 63], LS-ILDA [69] and ICLDA [70] for the case that new data samples are inserted as a chunk, we present the main cost of these five algorithms in Table 3.6. Those information that need to be kept in main memory for incremental learning is the same as we shown in Table 3.4. In Table 3.6, we assume for ILDA/SSS that the total scatter matrix is nearly full column rank and the performance of ILDA/SSS approximate the performance of batch LDA.

Table 3.6: Main computational cost (flops) of algorithms IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR for a chunk insertion (s samples)

Method	Time Complexity
Chunk IDR/QR	$2mk^2 + 26mks + 91/3k^3$
Chunk ILDA/SSS	$2m(n+s)^2 + 12(n+s)^3$
Chunk LS-ILDA	$14mns + 7mks$
Chunk ICLDA	$2mn^2s + 20n^3s$
Chunk ILDA/QR	$2ms(2n + 2s + 2k + \tilde{k})$

3.4 Numerical Experiments

In this section, we evaluate the efficiency of our new proposed incremental algorithm ILDA/QR by comparing with batch LDA algorithms: LDA/QR and ULDA/QR[22], and four existing incremental LDA algorithms: IDR/QR [102],

ILDA/SSS [62, 63], LS-ILDA [69] and ICLDA [70]. The performance is mainly measured by the computational cost (seconds) and the classification accuracy (as a percentage). Before reporting the experimental results, we discuss the testing databases and the experimental setting.

Experimental Platforms: All experiments were conducted by using a Sun v40z with 2.4GHz Opteron 850 CPUs and 32 GB RAM computer in Center for Computational Science and Engineering, National University of Singapore.

Experimental Data Sets: Our experiments were performed on the following 20 real-world data sets from three different sources, including text document, face image and gene expression. The structures of these datasets are summarized in Table 3.7, where m is the dimension of the dataset, n is the total sample size and k is the number of classes, a more detailed description of these data is presented in Appendix C.

For all data sets used in this section, we perform our study by repeated random splitting into two groups using the following algorithm: within each class, we randomly reorder the data and then for each class with size n_i, the first $\lceil 0.5n_i \rceil$ data are sorted into Group I and the others are sorted into Group II, whereby $\lceil \cdot \rceil$ is the ceiling function. Initially, we select the first $\lceil 0.5k \rceil$ classes of samples from Group I for training, while the others are inserted into the training set one by one incrementally for sequential incremental implementation and chunk by chunk for chunk incremental implementation. Incremental learning is completed until all samples in Group I are added into the training set. The classification accuracy of the final updated transformation matrix is then computed using Group II as the test data. The computational cost is the CPU time of updating the transformation matrix for one single insertion (sequential incremental experiment) or a chunk insertion (chunk incremental experiment). For each algorithm, to reduce the variability, this process is repeated for 10 times, and the average results are recorded.

K-Nearest Neighbor method (K-NN) [30] is a popular method for classification that gives the maximum likelihood estimation of the class posterior probabilities

by finding the closest K training points according to some metric, e.g., Euclidean, Manhattan, etc. In our experiments, K-NN with $K = 1$, which predicts the same class as the nearest instance in the training set, measured by Euclidean distance is used as classification algorithm. CPU time in this experiment is recorded by MATLAB commands *tic* and *toc*, which provide the time elapsed between their points of usage.

Table 3.7: Data Structures

Type	Data	m	n	k
Text Document	$K1a$	21839	2340	20
	$K1b$	21839	2340	6
	$Tr12$	5804	313	8
	$Tr23$	5832	204	6
	Wap	8460	1560	20
Face Image	$AR_{50\times40}$	2000	1680	120
	$AR_{50\times45}$	2250	1680	120
	$Feret$	6400	1000	200
	$ORL_{32\times32}$	1024	400	40
	$ORL_{64\times64}$	4096	400	40
	$Palmprint$	4096	600	100
	$Yale_{32\times32}$	1024	165	15
	$Yale_{64\times64}$	4096	165	15
	$YaleB$	32256	2424	38
Gene Expression	$Brain$	5597	42	5
	$Colon$	2000	62	2
	$Leukemia$	3571	72	2
	$Lymphoma$	4026	62	3
	$Prostate$	6033	102	2
	$SRBCT$	2308	63	4

3.4.1 Experiments for Sequential ILDA/QR

In the experiment for sequential ILDA/QR, new data samples are inserted into the training set one by one.

Comparison with Batch LDA

In this experiment, we compare the performance of ILDA/QR with its batch version LDA/QR and another LDA-based batch dimensionality reduction algorithm: ULDA/QR [22] which is summarized in Algorithm 3.2.

For batch algorithms: LDA/QR and ULDA/QR, when a new data sample is inserted, the construction of optimal transformation G is achieved from scratch. That it, either LDA/QR or ULDA/QR will repeat the learning from the beginning whenever one additional sample is presented, and the knowledge acquired in the past is discarded. The CPU time for LDA/QR and ULDA/QR is the total time of computing the optimal transformation. While for incremental algorithm, ILDA/QR, the optimal transformation is updated from the previously obtained information when a new sample is added. Thus, the CPU time for ILDA/QR to construct the optimal transformation is the updating time.

The results of mean classification accuracies of the final optimal transformation matrix and 10 times' standard deviation are shown in Table 3.8. To give a concrete idea of the benefit of using incremental method from the perspective of the computational efficiency, we show a comparison on the execution time of ILDA/QR with LDA/QR and ULDA/QR for each single updating in Figures 3.1-3.4. In the Figures, the horizontal axis shows the number of new inserted data items, and the vertical axis indicates the CPU time (seconds in logarithmic scale) of computing the transformation matrix.

Main observations are as follows:

- ILDA/QR achieves the same accuracies as that of LDA/QR, which coincides with our theoretical analysis that our new proposed incremental algorithm

ILDA/QR is an exact scheme of LDA/QR.

- ILDA/QR and LDA/QR are comparative with ULDA/QR in terms of classification accuracy. It is interesting to note that ILDA/QR or LDA/QR achieves higher accuracies than ULDA/QR for the text document datasets.

- Considering the execution time, ILDA/QR is much faster than LDA/QR and ULDA/QR. Indeed, for a single updating, the computational complexity of LDA/QR is $4mn^2 - \frac{4}{3}n^3$ and ULDA/QR is $4mn^2 + 4n^3$, while ILDA/QR only takes $4mn$ flops. As more new samples are inserted, that is, the sample size n increases, the speed-up of incremental algorithm ILDA/QR over batch algorithms LDA/QR and ULDA/QR keeps increasing. Generally speaking, LDA/QR outperforms ULDA/QR, especially when n is large, which can be explained by their computational complexity.

Table 3.8: Comparison of ULDA/QR, LDA/QR and ILDA/QR

Data	Method	Accuracy	Standard Deviation
$K1a$	ULDA/QR	80.34	0.63
	LDA/QR	82.37	0.84
	ILDA/QR	82.37	0.84
$K1b$	ULDA/QR	95.87	0.50
	LDA/QR	96.28	0.53
	ILDA/QR	96.29	0.52
$Tr12$	ULDA/QR	76.43	3.31
	LDA/QR	83.05	3.43
	ILDA/QR	83.05	3.43
$Tr23$	ULDA/QR	74.40	3.04
	LDA/QR	80.90	2.34

Continued on next page

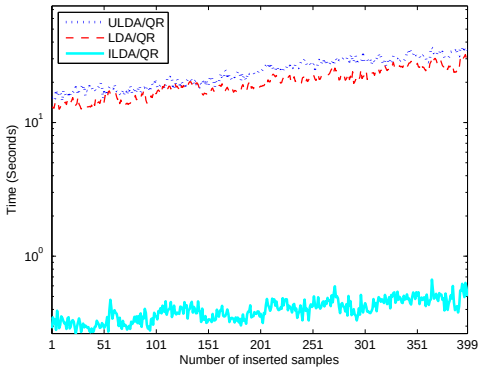
Table 3.8 – continued from previous page

Data	Method	Accuracy	Standard Deviation
	ILDA/QR	80.90	2.34
	ULDA/QR	76.40	1.03
Wap	LDA/QR	80.05	1.45
	ILDA/QR	80.05	1.45
	ULDA/QR	95.87	0.77
$AR_{50\times40}$	LDA/QR	95.98	0.81
	ILDA/QR	95.98	0.81
	ULDA/QR	79.15	0.78
$AR_{50\times45}$	LDA/QR	79.36	0.70
	ILDA/QR	79.36	0.70
	ULDA/QR	69.88	1.63
$Feret$	LDA/QR	70.28	1.46
	ILDA/QR	70.28	1.46
	ULDA/QR	91.25	1.63
$ORL_{32\times32}$	LDA/QR	91.35	1.55
	ILDA/QR	91.35	1.55
	ULDA/QR	94.65	1.52
$ORL_{64\times64}$	LDA/QR	94.00	1.38
	ILDA/QR	94.00	1.38
	ULDA/QR	99.27	0.33
$Palmprint$	LDA/QR	99.27	0.33
	ILDA/QR	99.27	0.33
	ULDA/QR	78.53	2.70
$Yale_{32\times32}$	LDA/QR	78.53	2.56
	ILDA/QR	78.53	2.56
	ULDA/QR	90.13	2.54
$Yale_{64\times64}$	LDA/QR	90.93	1.67
	ILDA/QR	90.80	1.63
	ULDA/QR	93.29	1.28

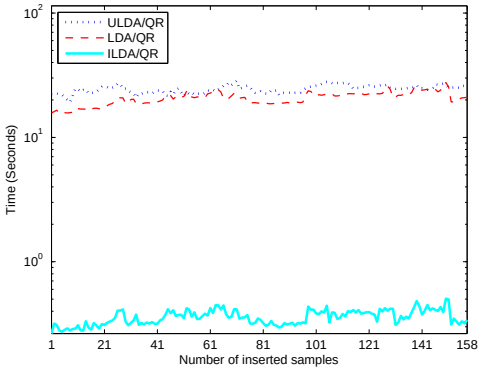
Continued on next page

Table 3.8 – continued from previous page

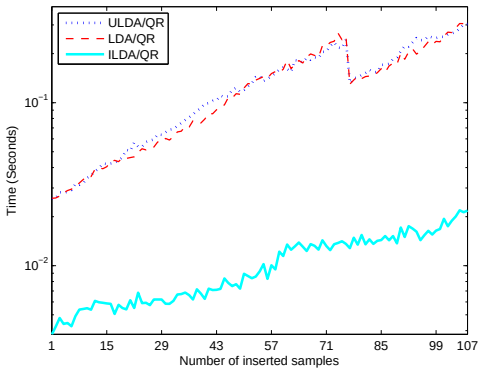
Data	Method	Accuracy	Standard Deviation
YaleB	LDA/QR	94.23	1.31
	ILDA/QR	94.23	1.31
	ULDA/QR	80.00	5.55
Brain	LDA/QR	81.90	5.13
	ILDA/QR	81.90	5.13
	ULDA/QR	84.84	3.24
Colon	LDA/QR	83.87	4.08
	ILDA/QR	83.87	4.08
	ULDA/QR	97.14	1.81
Leukemia	LDA/QR	97.43	2.00
	ILDA/QR	97.43	2.00
	ULDA/QR	100.00	0.00
Lymphoma	LDA/QR	97.00	3.14
	ILDA/QR	97.00	3.14
	ULDA/QR	91.57	2.78
Prostate	LDA/QR	91.57	2.78
	ILDA/QR	91.57	2.78
	ULDA/QR	97.74	2.07
SRBCT	LDA/QR	98.06	2.14
	ILDA/QR	98.06	2.14



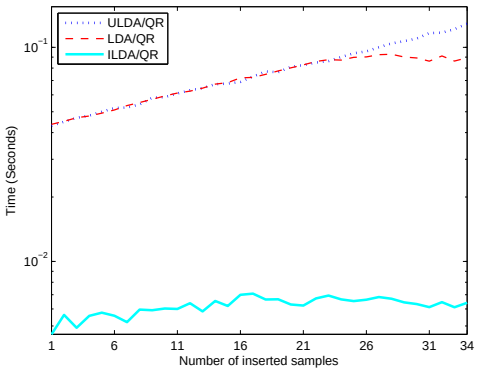
(a) CPU time for $K1a$



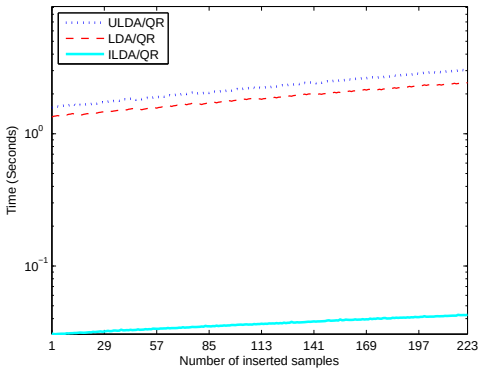
(b) CPU time for $K1b$



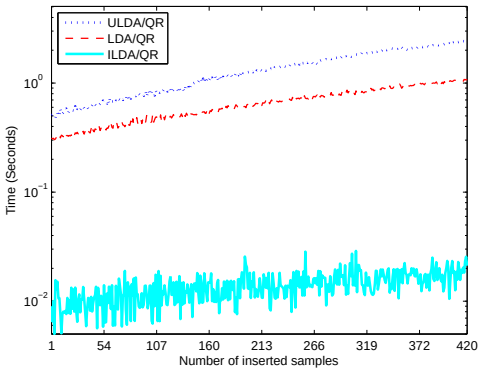
(c) CPU time for $Tr12$



(d) CPU time for $Tr23$

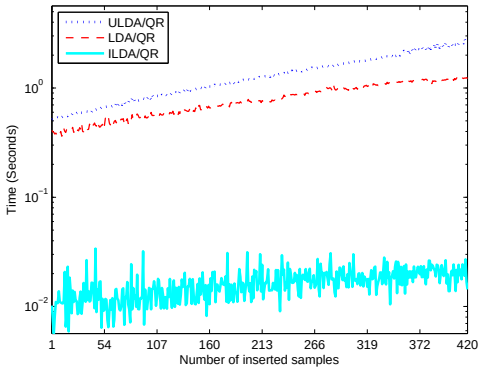


(e) CPU time for Wap

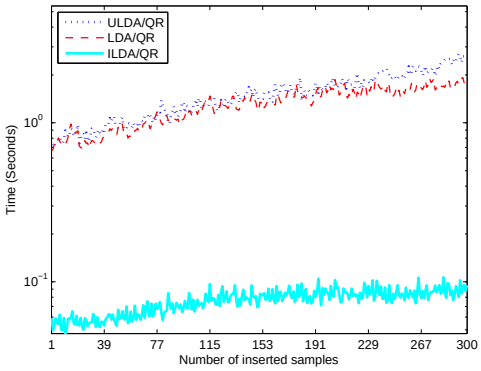


(f) CPU time for $AR_{50\times 40}$

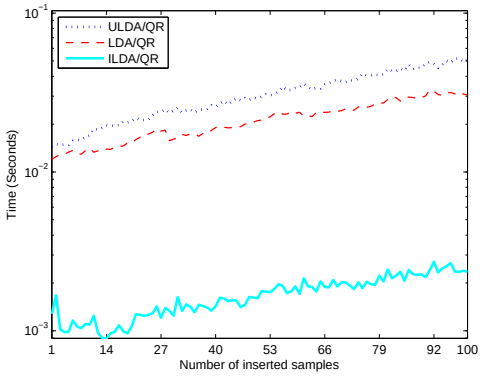
Figure 3.1: Comparing the CPU time of ULDA/QR, LDA/QR and ILDA/QR (measured in log scale)



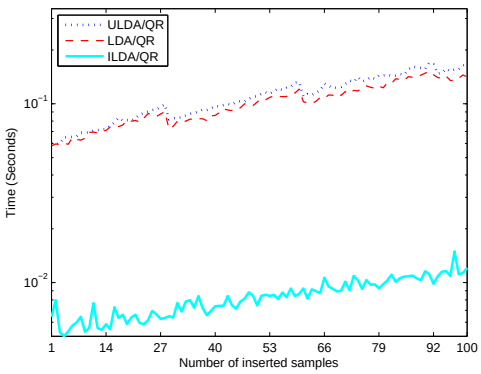
(a) CPU time for $AR_{50\times45}$



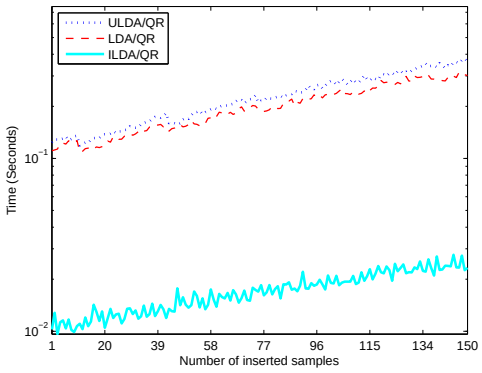
(b) CPU time for $Feret$



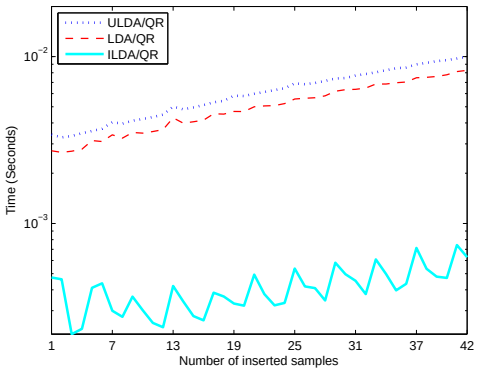
(c) CPU time for $ORL_{32\times32}$



(d) CPU time for $ORL_{64\times64}$

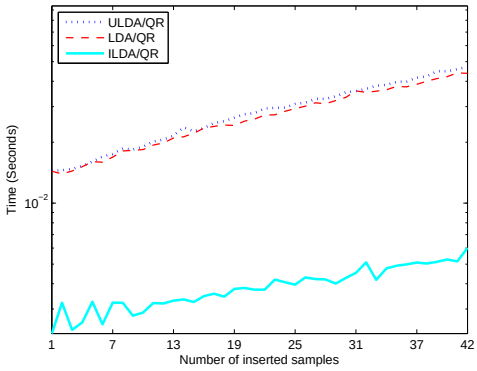


(e) CPU time for $Palmprint$

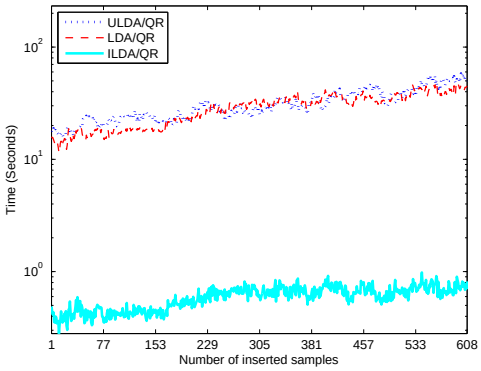


(f) CPU time for $Yale_{32\times32}$

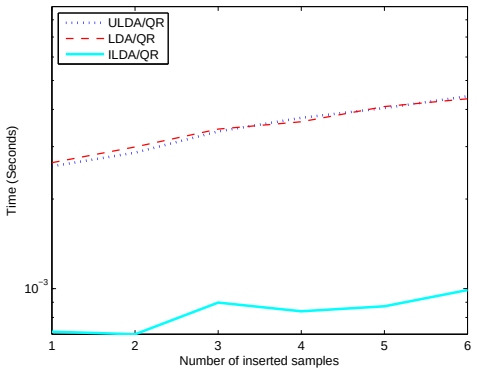
Figure 3.2: Comparing the CPU time of ULDA/QR, LDA/QR and ILDA/QR (measured in log scale)



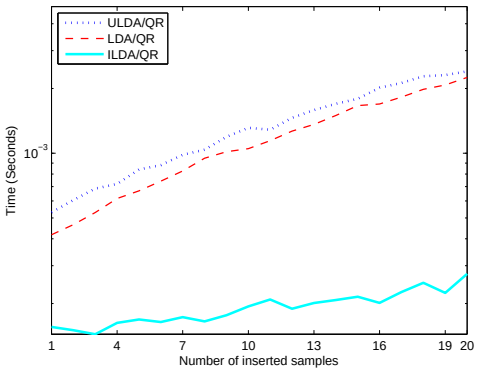
(a) CPU time for $Yale_{64\times64}$



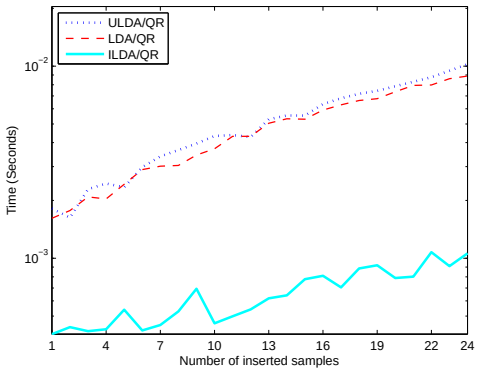
(b) CPU time for $YaleB$



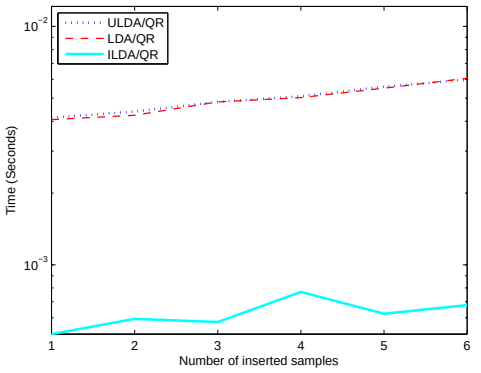
(c) CPU time for *Brain*



(d) CPU time for *Colon*

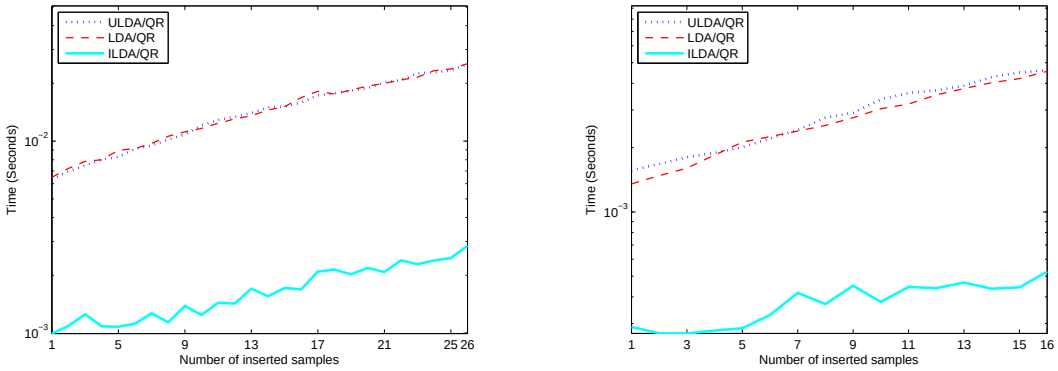


(e) CPU time for *Leukemia*



(f) CPU time for *Lymphoma*

Figure 3.3: Comparing the CPU time of ULDA/QR, LDA/QR and ILDA/QR (measured in log scale)



(a) CPU time for *Prostate* (b) CPU time for *SRBCT*

Figure 3.4: Comparing the CPU time of ULDA/QR, LDA/QR and ILDA/QR (measured in log scale)

Comparison with Some Existing Incremental LDA

In this experiment, we compare the performance of our new proposed incremental algorithm ILDA/QR with that of four existing incremental LDA: IDR/QR [102], ILDA/SSS [62, 63], LS-ILDA [69] and ICLDA [70].

For ILDA/SSS [62], we used the MATLAB code written by one of the authors on the website [61]. While for IDR/QR [102], LS-ILDA [69] and ICLDA [70], we wrote the MATLAB codes that follow their algorithms presented in Chapter 2. As we mentioned in Chapter 2, ILDA/SSS has three parameters: the threshold for significant components of the total scatter matrix, the threshold for significant components of the between-class scatter matrix and the threshold for the discriminative components. The authors of ILDA/SSS did not give the best parameter setting either in the paper or on the website, we used the same threshold 0.1 for these three parameters as Liu et. al. selected in [69], which enables the algorithm to achieve its best performance. The regularization parameter μ in IDR/QR was set to be 0.5, which produced good overall results in [102].

The execution time for incremental algorithms in this experiment is the total CPU time of updating the optimal transformation matrix in a single insertion,

and the classification accuracy is of the final transformation when incremental updating is completed. The results of mean classification accuracies of 10 times' and corresponding standard deviation are shown in Table 3.10, the mean execution time of each upating is shown in Figures 3.5-3.8. In the Figures, the horizontal axis shows the number of inserted samples while the vertical axis indicates the execution time (in log-scale) of different tested methods.

The following observations can be made from Table 3.10 and Figures 3.5-3.8:

- Overall, ILDA/QR always achieve comparative classification accuracy among the five incremental algorithms.
  - It is interesting to note that ILDA/QR achieves higher accuracies than IDR/QR, ILDA/SSS, LS-ILDA and ICLDA on text document datasets. While ICLDA achieves relatively high accuracies on some face image datasets. As illustrated in Chapter 2, the dimension of the reduced space of ICLDA is $n-1$ when the training samples are linearly independent which is just the case in our experiment. Thus, it is reasonable that ICLDA has good classification performance. However, when sample size n is large, the reduced representation may not be suitable for efficient indexing and retrieval.
  - For some datasets, IDR/QR produces relatively low accuracies compared with the other four algorithms, such as, $Tr12$, $Tr23$, $Feret$, $Yale_{32\times32}$, $Yale_{64\times64}$ and $Prostate$. This is mainly caused by projecting the scatter matrices into the range space of the between-class scatter matrix, in which some useful information are discarded, as well as the approximation for the updating of reduced scatter matrix W.
  - It is interesting to note that LS-ILDA achieves almost the same accuracies as ILDA/SSS on some datasets. However, the performance of ILDA/SSS is largely determined by the threshold. As shown in Table 3.11, the accuracy deviation between ILDA/SSS with threshold 0.1 and

threshold 1 exceeds 10%.

Table 3.9: Comparison of classification accuracies of ILDA/SSS with different threholds: 0.1 and 1

Accuracy	$Tr12$	$ORL_{32\times32}$	$Brain$
threshold 0.1	76.43	91.25	80.00
threshold 1	60.97	80.10	57.61

- The execution time for computing the transformation in a single updating by IDR/QR or ILDA/QR is significantly smaller than by ILDA/SSS, LS-ILDA and ICLDA.
  - ILDA/QR and IDR/QR are very fast, much faster than ILDA/SSS, LS-ILDA, and ICLDA. ILDA/QR is faster than IDR/QR on face imagine datasets, while IDR/QR is faster than ILDA/QR on text document datasets. Indeed, for a single updating, IDR/QR takes $O(mk^2 + k^3)$, while ILDA/QR takes $O(mn)$, where k is the number of classes in the current training set and n is the size of the current training set. When $n \gg k$, IDR/QR costs less than ILDA/QR, otherwise, ILDA/QR performs faster.
  - ILDA/SSS and ICLDA are the two slowest incremental algorithms, as the computational costs of them are about $O(mn^2 + n^3)$ for each updating. It is interesting to see that ICLDA outperforms ILDA/SSS on text documents and face images, while ILDA/SSS outperforms ICLDA on gene expression datasets.
  - LS-ILDA is faster than ICLDA and ILDA/SSS but slower than IDR/QR and ILDA/QR.
- Considering both classification accuracy and computational cost, ILDA/QR is the best choice among the five compared algorithms. It provides an efficient and effective incremental dimensionality reduction for large-scale streaming datasets.

Table 3.10: Comparison of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR

Data	Method	Accuracy	Standard Deviation
	IDR/QR	80.71	0.37
	ILDA/SSS	80.34	0.63
$K1a$	LS-ILDA	80.34	0.63
	ICLDA	76.62	9.35
	ILDA/QR	82.37	0.84
	IDR/QR	95.97	0.35
	ILDA/SSS	95.87	0.50
$K1b$	LS-ILDA	95.87	0.50
	ICLDA	91.44	11.98
	ILDA/QR	96.29	0.52
	IDR/QR	69.61	4.59
	ILDA/SSS	76.43	3.31
$Tr12$	LS-ILDA	76.43	3.31
	ICLDA	74.81	9.93
	ILDA/QR	83.05	3.43
	IDR/QR	72.80	5.31
	ILDA/SSS	74.40	3.04
$Tr23$	LS-ILDA	74.40	3.04
	ICLDA	74.10	2.77
	ILDA/QR	80.90	2.34
	IDR/QR	79.06	1.37
	ILDA/SSS	76.40	1.03
Wap	LS-ILDA	76.40	1.03
	ICLDA	75.57	2.60
	ILDA/QR	80.05	1.45
	IDR/QR	96.40	0.61
	ILDA/SSS	95.87	0.77
$AR_{50\times40}$	LS-ILDA	95.87	0.77
	ICLDA	98.20	0.62

Continued on next page

Table 3.10 – continued from previous page

Data	Method	Accuracy	Standard Deviation
	ILDA/QR	95.98	0.81
$AR_{50\times45}$	IDR/QR	71.33	1.09
	ILDA/SSS	79.15	0.78
	LS-ILDA	79.15	0.78
	ICLDA	87.71	1.11
	ILDA/QR	79.36	0.70
$Feret$	IDR/QR	51.08	2.51
	ILDA/SSS	69.88	1.63
	LS-ILDA	69.88	1.63
	ICLDA	85.60	1.30
	ILDA/QR	70.28	1.46
$ORL_{32\times32}$	IDR/QR	92.80	1.12
	ILDA/SSS	91.25	1.63
	LS-ILDA	91.25	1.63
	ICLDA	97.00	1.02
	ILDA/QR	91.35	1.55
$ORL_{64\times64}$	IDR/QR	90.60	1.89
	ILDA/SSS	93.95	1.68
	LS-ILDA	94.65	1.52
	ICLDA	97.10	1.02
	ILDA/QR	94.00	1.38
$Palmprint$	IDR/QR	98.37	0.59
	ILDA/SSS	99.27	0.33
	LS-ILDA	99.27	0.33
	ICLDA	98.50	0.75
	ILDA/QR	99.27	0.33
$Yale_{32\times32}$	IDR/QR	64.13	4.75
	ILDA/SSS	78.53	2.70
	LS-ILDA	78.53	2.70

Continued on next page

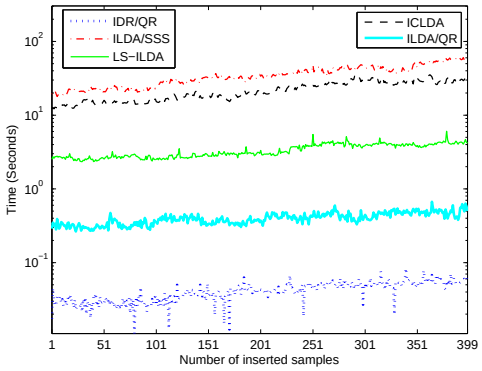
Table 3.10 – continued from previous page

Data	Method	Accuracy	Standard Deviation
	ICLDA	80.53	3.69
	ILDA/QR	78.53	2.56
	IDR/QR	72.13	4.36
	ILDA/SSS	87.73	1.67
$Yale_{64\times64}$	LS-ILDA	90.13	2.54
	ICLDA	87.47	3.44
	ILDA/QR	90.80	1.63
	IDR/QR	75.40	1.52
	ILDA/SSS	84.40	1.68
$YaleB$	LS-ILDA	93.29	1.28
	ICLDA	92.72	1.15
	ILDA/QR	94.23	1.31
	IDR/QR	81.43	6.19
	ILDA/SSS	80.00	5.55
$Brain$	LS-ILDA	80.00	5.55
	ICLDA	79.52	5.65
	ILDA/QR	81.90	5.13
	IDR/QR	79.68	7.50
	ILDA/SSS	84.84	3.24
$Colon$	LS-ILDA	84.84	3.24
	ICLDA	84.84	3.24
	ILDA/QR	83.87	4.08
	IDR/QR	95.14	2.57
	ILDA/SSS	97.14	1.81
$Leukemia$	LS-ILDA	97.14	1.81
	ICLDA	97.14	1.81
	ILDA/QR	97.43	2.00
	IDR/QR	98.33	2.24
	ILDA/SSS	100.00	0.00

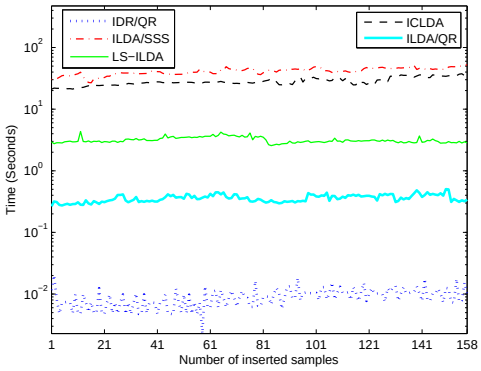
Continued on next page

Table 3.10 – continued from previous page

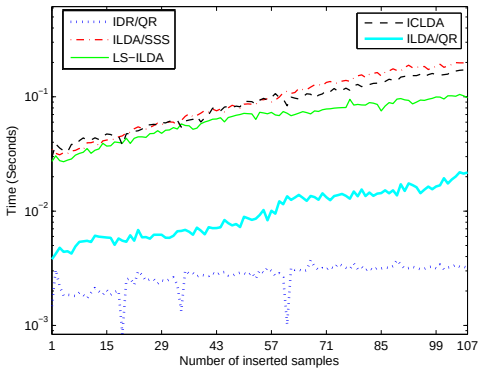
Data	Method	Accuracy	Standard Deviation
Lymphoma	LS-ILDA	100.00	0.00
	ICLDA	100.00	0.00
	ILDA/QR	97.00	3.14
Prostate	IDR/QR	74.12	7.00
	ILDA/SSS	91.57	2.78
	LS-ILDA	91.57	2.78
	ICLDA	91.57	2.78
	ILDA/QR	91.57	2.78
SRBCT	IDR/QR	94.19	3.76
	ILDA/SSS	97.74	2.07
	LS-ILDA	97.74	2.07
	ICLDA	97.74	2.07
	ILDA/QR	98.06	2.14



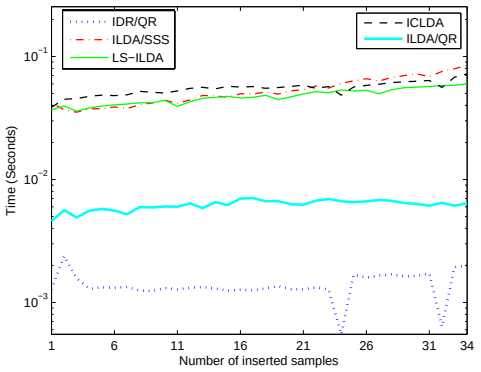
(a) CPU time for $K1a$



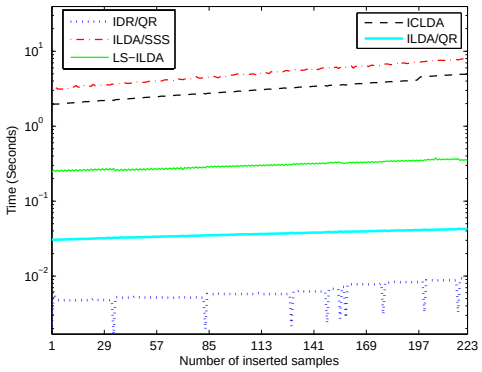
(b) CPU time for $K1b$



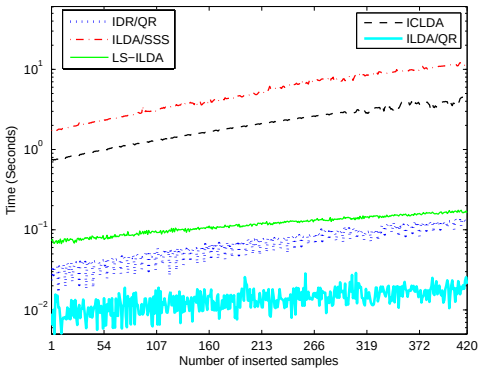
(c) CPU time for $Tr12$



(d) CPU time for $Tr23$

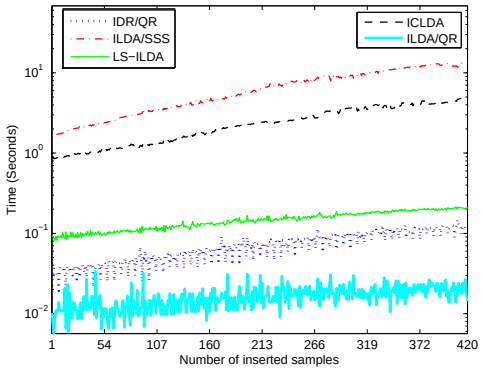


(e) CPU time for Wap

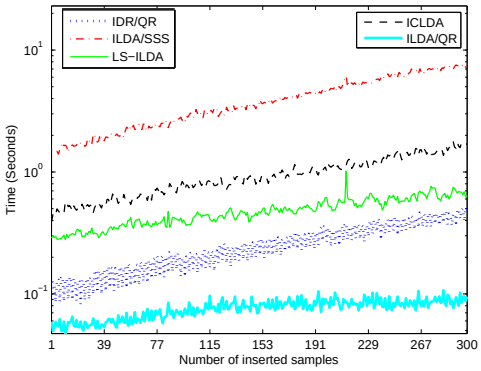


(f) CPU time for $AR_{50\times40}$

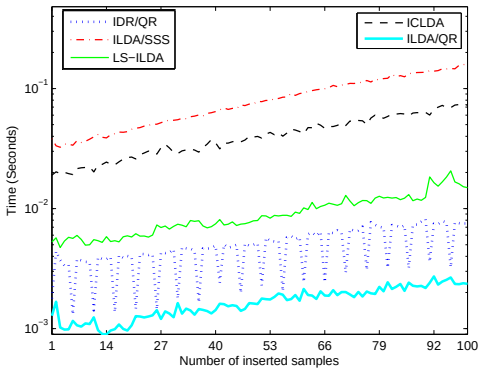
Figure 3.5: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR (measured in seconds in log-scale)



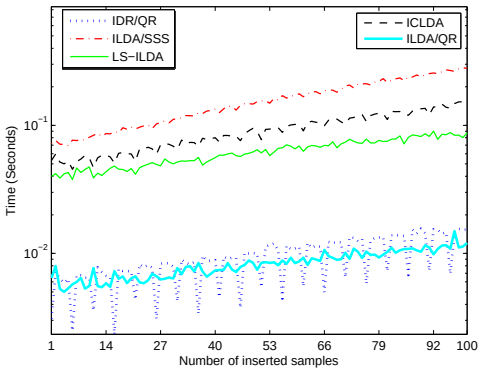
(a) CPU time for $AR_{50\times45}$



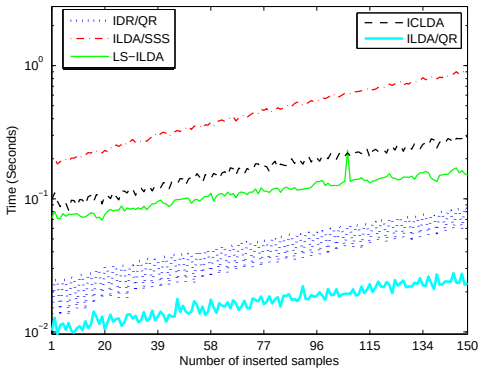
(b) CPU time for $Feret$



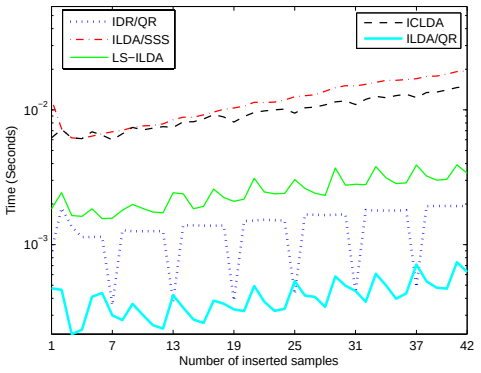
(c) CPU time for $ORL_{32\times32}$



(d) CPU time for $ORL_{64\times64}$

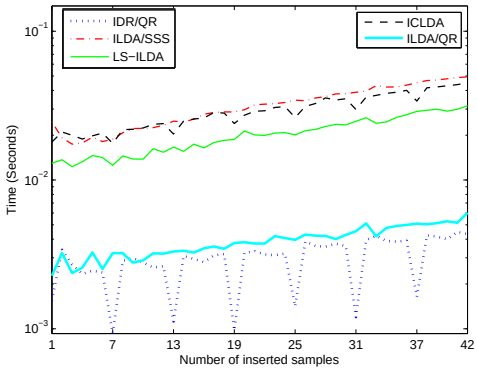


(e) CPU time for $Palmprint$

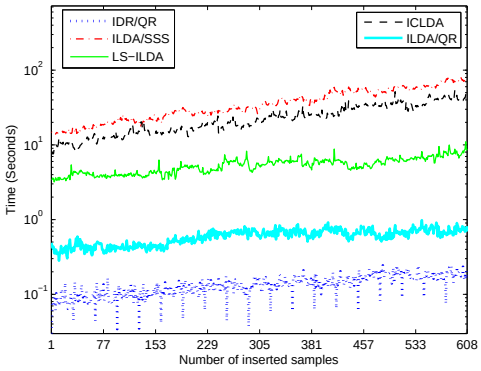


(f) CPU time for $Yale_{32\times32}$

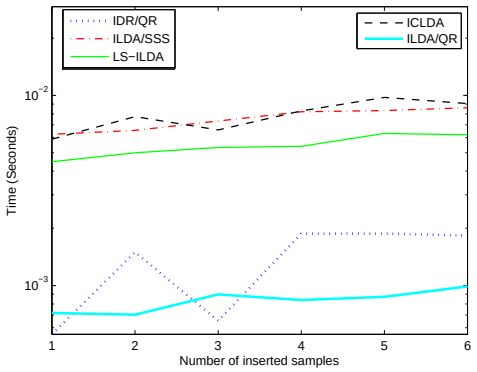
Figure 3.6: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR (measured in seconds in log-scale)



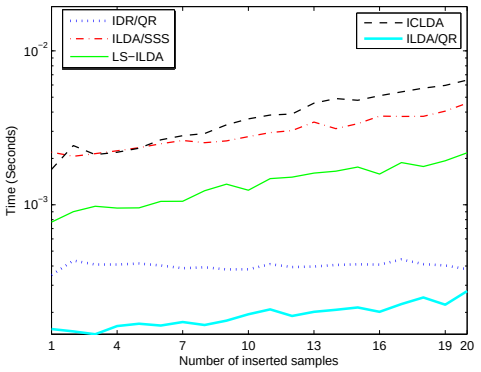
(a) CPU time for $Yale_{64\times64}$



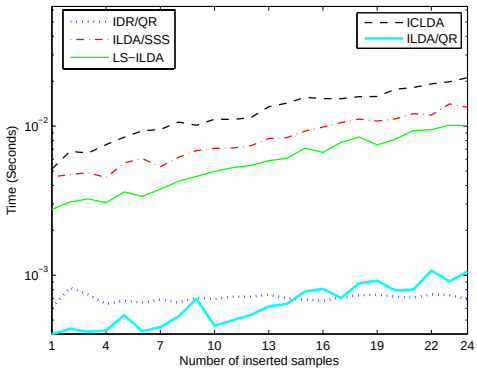
(b) CPU time for $YaleB$



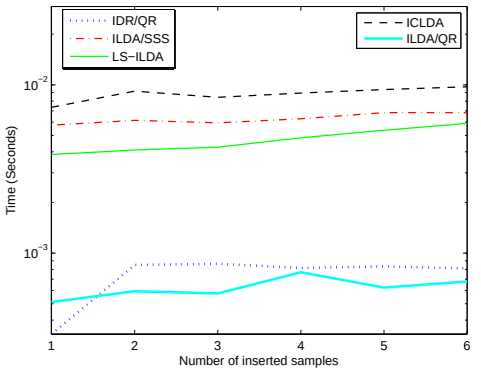
(c) CPU time for *Brain*



(d) CPU time for *Colon*

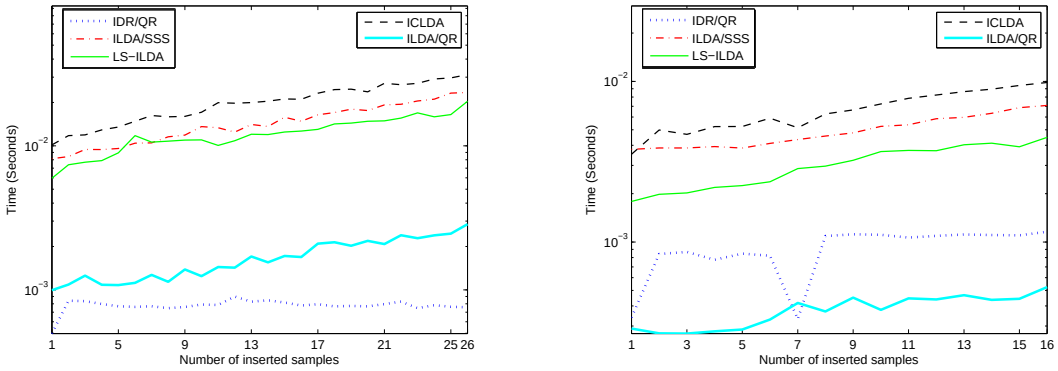


(e) CPU time for *Leukemia*



(f) CPU time for *Lymphoma*

Figure 3.7: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR (measured in seconds in log-scale)



(a) CPU time for *Prostate* (b) CPU time for *SRBCT*

Figure 3.8: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA and ILDA/QR (measured in seconds in log-scale)

3.4.2 Experiments for Chunk ILDA/QR

In the experiment for chunk ILDA/QR, new data samples are inserted into the training set chunk by chunk (The size of each chunk is shown in the CPU-time figures). The recorded CPU-time is the execution time for a chunk insertion.

Comparison with LDA/QR and Sequential ILDA/QR

In this experiment, we compare the performance of chunk ILDA/QR with its batch algorithm LDA/QR and its sequential version. The results of mean classification accuracies and 10 times' standard deviation are presented in Table 3.11. And the CPU-time of these three algorithms is given in Figures 3.9-3.12.

Main observations are as follows:

- ILDA/QR and LDA/QR yield the same accuracies for all tested data sets except for data $Tr12$ and $Yale_{64\times64}$, which coincides with our theoretical analysis that both of our sequential and chunk incremental algorithms are exact schemes of their batch version. The difference between ILDA/QR and LDA/QR on data $Tr12$ and $Yale_{64\times64}$ is subtle which is less than 2%.

- Chunk ILDA/QR is obviously faster than LDA/QR and sequential ILDA/QR.
- When chunk size s is relatively large, LDA/QR is faster than sequential ILDA/QR, otherwise, sequential ILDA/QR is faster than LDA/QR. This is because, for sequential ILDA/QR, one chunk insertion process consists of s times' single updating. Thus, in practical application, preferably chunk ILDA/QR is utilized when the size of the new presented data samples is large.

Table 3.11: Comparison of LDA/QR, ILDA/QR and ILDA/QR(Chunk)

Data	Method	Accuracy	Standard Deviation
$K1a$	LDA/QR	82.37	0.84
	ILDA/QR	82.37	0.84
	ILDA/QR(Chunk)	82.37	0.84
$K1b$	LDA/QR	96.28	0.53
	ILDA/QR	96.29	0.52
	ILDA/QR(Chunk)	96.29	0.52
$Tr12$	LDA/QR	83.05	3.43
	ILDA/QR	83.05	3.43
	ILDA/QR(Chunk)	81.10	3.90
$Tr23$	LDA/QR	80.90	2.34
	ILDA/QR	80.90	2.34
	ILDA/QR(Chunk)	80.90	2.34
Wap	LDA/QR	80.05	1.45
	ILDA/QR	80.05	1.45
	ILDA/QR(Chunk)	80.05	1.45
$AR_{50\times40}$	LDA/QR	95.98	0.81
	ILDA/QR	95.98	0.81
	ILDA/QR(Chunk)	95.98	0.81
$AR_{50\times45}$	LDA/QR	79.36	0.70
	ILDA/QR	79.36	0.70

Continued on next page

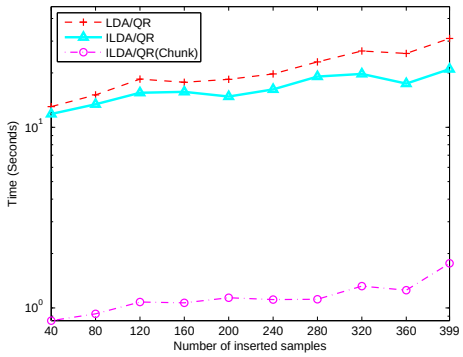
Table 3.11 – continued from previous page

Data	Method	Accuracy	Standard Deviation
	ILDA/QR(Chunk)	79.36	0.70
	LDA/QR	70.28	1.46
$Feret$	ILDA/QR	70.28	1.46
	ILDA/QR(Chunk)	70.28	1.46
	LDA/QR	91.35	1.55
$ORL_{32\times32}$	ILDA/QR	91.35	1.55
	ILDA/QR(Chunk)	91.35	1.55
	LDA/QR	94.00	1.38
$ORL_{64\times64}$	ILDA/QR	94.00	1.38
	ILDA/QR(Chunk)	94.00	1.38
	LDA/QR	99.27	0.33
$Palmprint$	ILDA/QR	99.27	0.33
	ILDA/QR(Chunk)	99.27	0.33
	LDA/QR	78.53	2.56
$Yale_{32\times32}$	ILDA/QR	78.53	2.56
	ILDA/QR(Chunk)	78.53	2.56
	LDA/QR	90.93	1.67
$Yale_{64\times64}$	ILDA/QR	90.80	1.63
	ILDA/QR(Chunk)	90.27	2.53
	LDA/QR	94.23	1.31
$YaleB$	ILDA/QR	94.23	1.31
	ILDA/QR(Chunk)	94.23	1.31
	LDA/QR	81.90	5.13
$Brain$	ILDA/QR	81.90	5.13
	ILDA/QR(Chunk)	81.90	5.13
	LDA/QR	83.87	4.08
$Colon$	ILDA/QR	83.87	4.08
	ILDA/QR(Chunk)	83.87	4.08
	LDA/QR	97.43	2.00

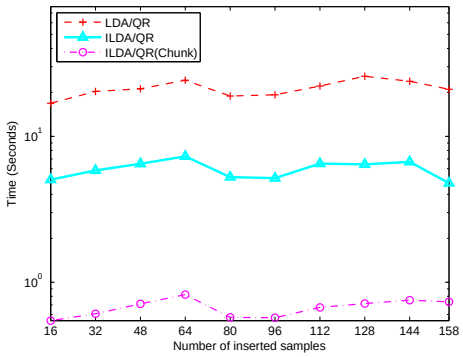
Continued on next page

Table 3.11 – continued from previous page

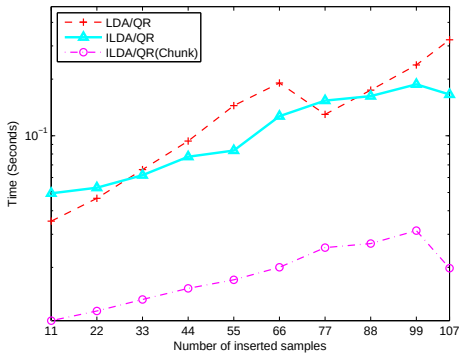
Data	Method	Accuracy	Standard Deviation
Leukemia	ILDA/QR	97.43	2.00
	ILDA/QR(Chunk)	97.43	2.00
	LDA/QR	97.00	3.14
Lymphoma	ILDA/QR	97.00	3.14
	ILDA/QR(Chunk)	97.00	3.14
	LDA/QR	91.57	2.78
Prostate	ILDA/QR	91.57	2.78
	ILDA/QR(Chunk)	91.57	2.78
	LDA/QR	98.06	2.14
SRBCT	ILDA/QR	98.06	2.14
	ILDA/QR(Chunk)	98.06	2.14



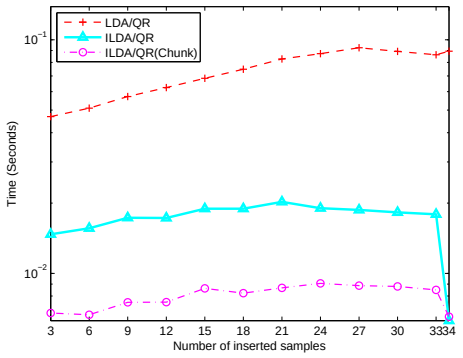
(a) CPU time for $K1a$



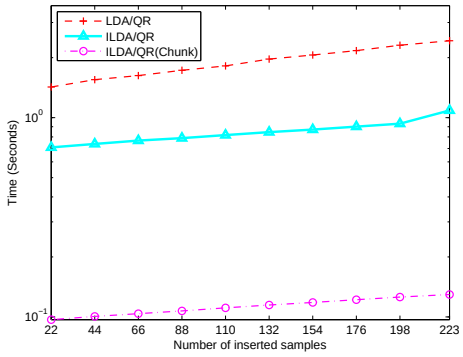
(b) CPU time for $K1b$



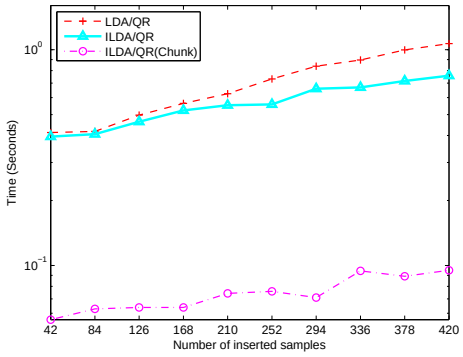
(c) CPU time for $Tr12$



(d) CPU time for $Tr23$

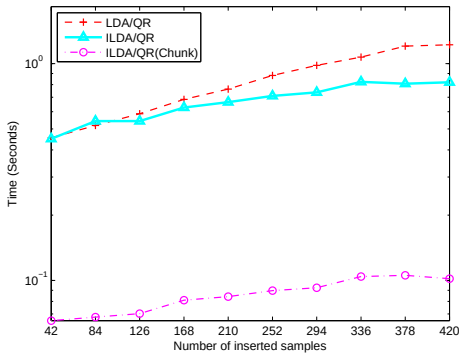


(e) CPU time for Wap

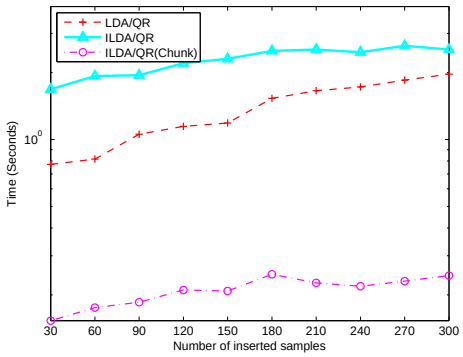


(f) CPU time for $AR_{50\times 40}$

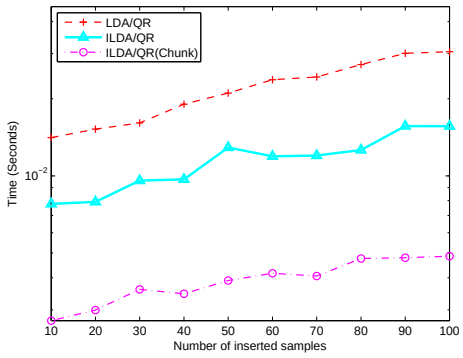
Figure 3.9: Comparing the CPU time of LDA/QR, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)



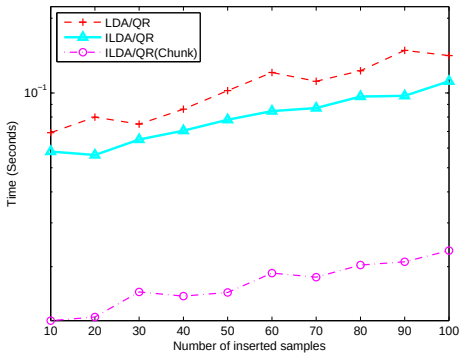
(a) CPU time for $AR_{50\times45}$



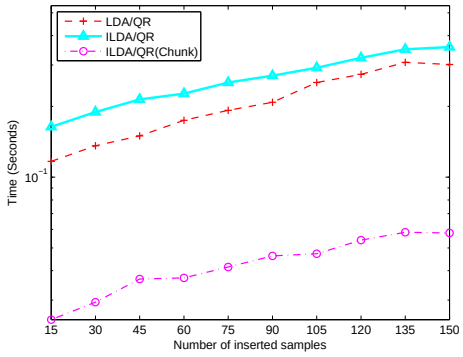
(b) CPU time for $Feret$



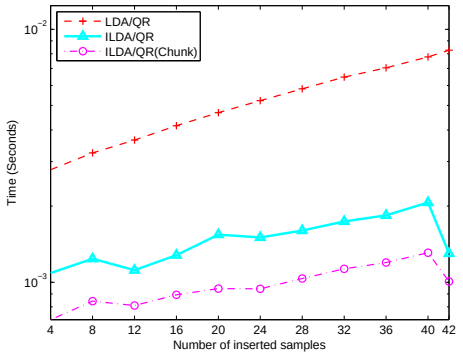
(c) CPU time for $ORL_{32\times32}$



(d) CPU time for $ORL_{64\times64}$

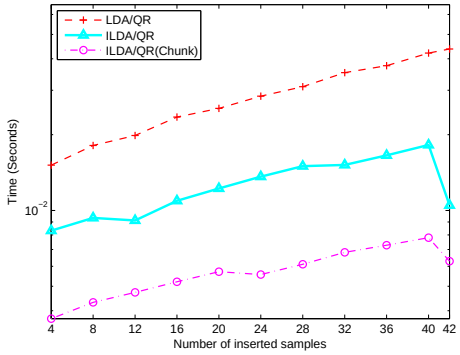


(e) CPU time for $Palmprint$

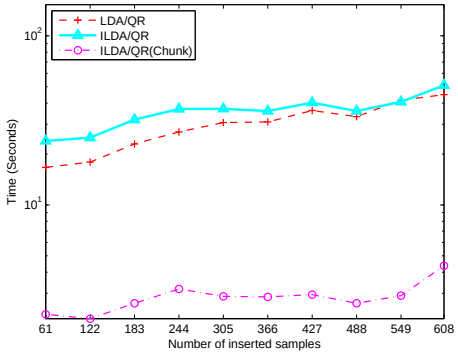


(f) CPU time for $Yale_{32\times32}$

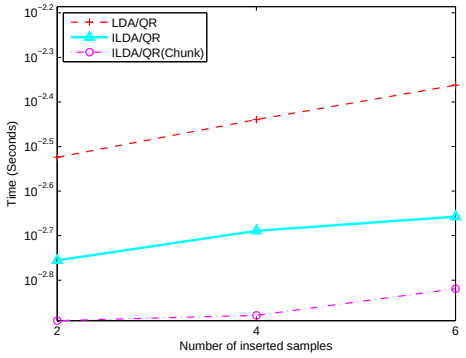
Figure 3.10: Comparing the CPU time of LDA/QR, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)



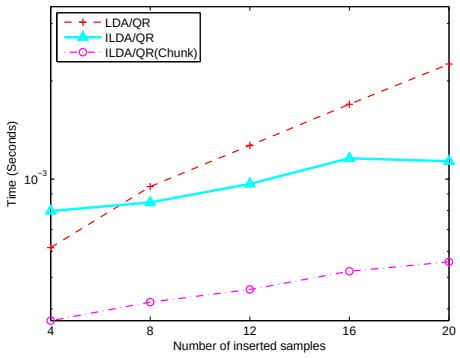
(a) CPU time for $Yale_{64\times64}$



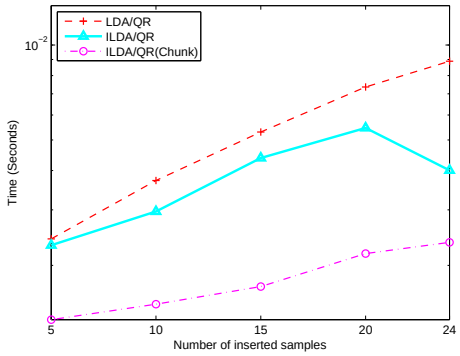
(b) CPU time for $YaleB$



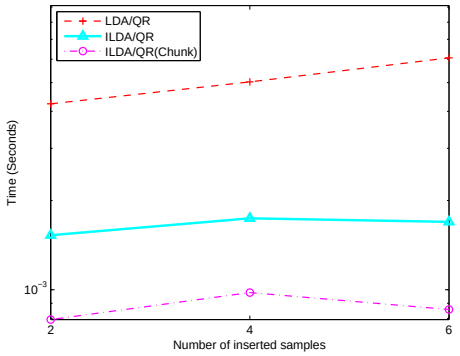
(c) CPU time for *Brain*



(d) CPU time for *Colon*

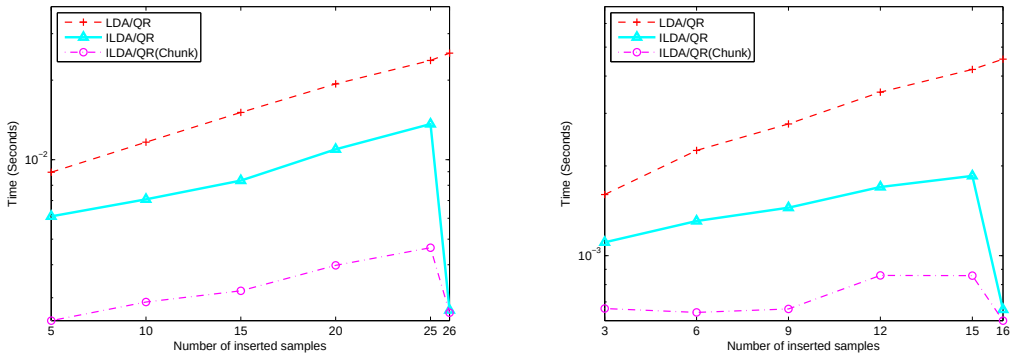


(e) CPU time for *Leukemia*



(f) CPU time for *Lymphoma*

Figure 3.11: Comparing the CPU time of LDA/QR, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)



(a) CPU time for *Prostate* (b) CPU time for *SRBCT*

Figure 3.12: Comparing the CPU time of LDA/QR, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)

Comparison with Existing Incremental LDA

In this experiment, we compare the performance of chunk ILDA/QR with four existing incremental LDA: IDR/QR [102], ILDA/SSS [62, 63], LS-ILDA [69] and ICLDA [70]. The results for classification accuracies and execution time are presented in Table 3.12 and Figures 3.13-3.16, respectively.

Main observations are shown in the following:

- Similarly to the performance of sequential ILDA/QR, our new chunk ILDA/QR always produces reasonable classification accuracies. For more details, please refer to the experimental results shown in Section 3.4.1.
- Chunk ILDA/QR is faster than IDR/QR for all face image data sets and all gene expression data sets except for *prostate*; IDR/QR is faster than chunk ILDA/QR for text document data sets $Tr12$, $Tr23$ and $K1b$; chunk ILDA/QR and IDR/QR are comparative in terms of execution time for data sets Wap, $K1a$ and *prostate*. Compared with the numerical experiment of sequential ILDA/QR, chunk ILDA/QR improves its computational performance.

Table 3.12: Comparison of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA, ILDA/QR and ILDA/QR(Chunk)

Data	Method	Accuracy	Standard Deviation
	IDR/QR	80.71	0.37
	ILDA/SSS	80.34	0.63
$K1a$	LS-ILDA	80.34	0.63
	ICLDA	76.62	9.35
	ILDA/QR(Chunk)	82.37	0.84
	IDR/QR	95.97	0.35
	ILDA/SSS	95.87	0.50
$K1b$	LS-ILDA	95.87	0.50
	ICLDA	91.44	11.98
	ILDA/QR(Chunk)	96.29	0.52
	IDR/QR	69.61	4.59
	ILDA/SSS	76.43	3.31
$Tr12$	LS-ILDA	76.43	3.31
	ICLDA	74.81	9.93
	ILDA/QR(Chunk)	81.10	3.90
	IDR/QR	72.80	5.31
	ILDA/SSS	74.40	3.04
$Tr23$	LS-ILDA	74.40	3.04
	ICLDA	74.10	2.77
	ILDA/QR(Chunk)	80.90	2.34
	IDR/QR	79.06	1.37
	ILDA/SSS	76.40	1.03
Wap	LS-ILDA	76.40	1.03
	ICLDA	75.17	3.74
	ILDA/QR(Chunk)	80.05	1.45
	IDR/QR	96.40	0.61
	ILDA/SSS	95.87	0.77
$AR_{50\times40}$	LS-ILDA	95.87	0.77

Continued on next page

Table 3.12 – continued from previous page

Data	Method	Accuracy	Standard Deviation
	ICLDA	98.20	0.62
	ILDA/QR(Chunk)	95.98	0.81
	IDR/QR	71.33	1.09
	ILDA/SSS	79.15	0.78
$AR_{50\times45}$	LS-ILDA	79.15	0.78
	ICLDA	87.71	1.11
	ILDA/QR(Chunk)	79.36	0.70
	IDR/QR	51.08	2.51
	ILDA/SSS	69.88	1.63
$Feret$	LS-ILDA	69.88	1.63
	ICLDA	85.60	1.30
	ILDA/QR(Chunk)	70.28	1.46
	IDR/QR	92.80	1.12
	ILDA/SSS	91.25	1.63
$ORL_{32\times32}$	LS-ILDA	91.25	1.63
	ICLDA	97.00	1.02
	ILDA/QR(Chunk)	91.35	1.55
	IDR/QR	90.60	1.89
	ILDA/SSS	93.95	1.68
$ORL_{64\times64}$	LS-ILDA	94.65	1.52
	ICLDA	97.10	1.02
	ILDA/QR(Chunk)	94.00	1.38
	IDR/QR	98.37	0.59
	ILDA/SSS	99.27	0.33
$Palmprint$	LS-ILDA	99.27	0.33
	ICLDA	98.50	0.75
	ILDA/QR(Chunk)	99.27	0.33
	IDR/QR	64.13	4.75
	ILDA/SSS	78.53	2.70

Continued on next page

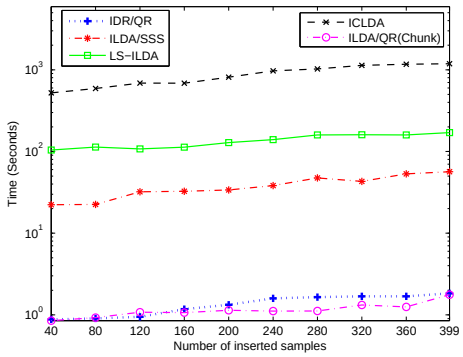
Table 3.12 – continued from previous page

Data	Method	Accuracy	Standard Deviation
$Yale_{32\times32}$	LS-ILDA	78.53	2.70
	ICLDA	80.53	3.69
	ILDA/QR(Chunk)	78.53	2.56
$Yale_{64\times64}$	IDR/QR	72.13	4.36
	ILDA/SSS	87.73	1.67
	LS-ILDA	90.13	2.54
	ICLDA	87.47	3.44
	ILDA/QR(Chunk)	90.27	2.53
$YaleB$	IDR/QR	75.40	1.52
	ILDA/SSS	84.40	1.68
	LS-ILDA	93.29	1.28
	ICLDA	92.72	1.15
	ILDA/QR(Chunk)	94.23	1.31
$Brain$	IDR/QR	81.43	6.19
	ILDA/SSS	80.00	5.55
	LS-ILDA	80.00	5.55
	ICLDA	79.52	5.65
	ILDA/QR(Chunk)	81.90	5.13
$Colon$	IDR/QR	79.68	7.50
	ILDA/SSS	84.84	3.24
	LS-ILDA	84.84	3.24
	ICLDA	84.84	3.24
	ILDA/QR(Chunk)	83.87	4.08
$Leukemia$	IDR/QR	95.14	2.57
	ILDA/SSS	97.14	1.81
	LS-ILDA	97.14	1.81
	ICLDA	97.14	1.81
	ILDA/QR(Chunk)	97.43	2.00
	IDR/QR	98.33	2.24

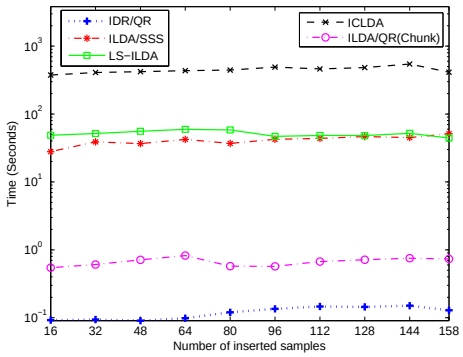
Continued on next page

Table 3.12 – continued from previous page

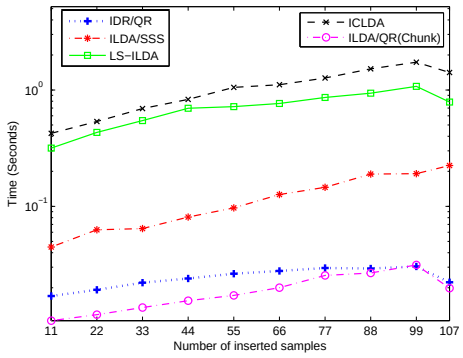
Data	Method	Accuracy	Standard Deviation
Lymphoma	ILDA/SSS	100.00	0.00
	LS-ILDA	100.00	0.00
	ICLDA	100.00	0.00
	Chunk ILDA/QR	97.00	3.14
Prostate	IDR/QR	74.12	7.00
	ILDA/SSS	91.57	2.78
	LS-ILDA	91.57	2.78
	ICLDA	91.57	2.78
	ILDA/QR(Chunk)	91.57	2.78
SRBCT	IDR/QR	94.19	3.76
	ILDA/SSS	97.74	2.07
	LS-ILDA	97.74	2.07
	ICLDA	97.74	2.07
	ILDA/QR(Chunk)	98.06	2.14



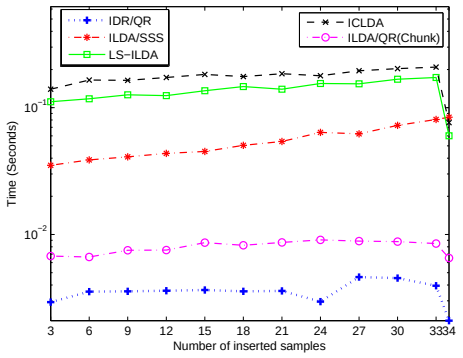
(a) CPU time for $K1a$



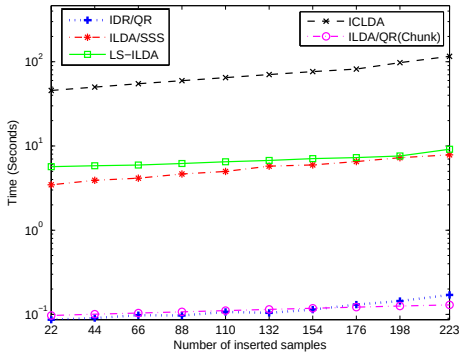
(b) CPU time for $K1b$



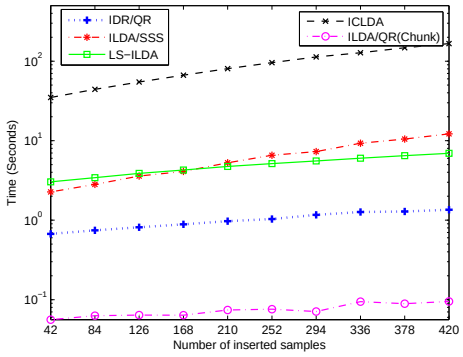
(c) CPU time for $Tr12$



(d) CPU time for $Tr23$

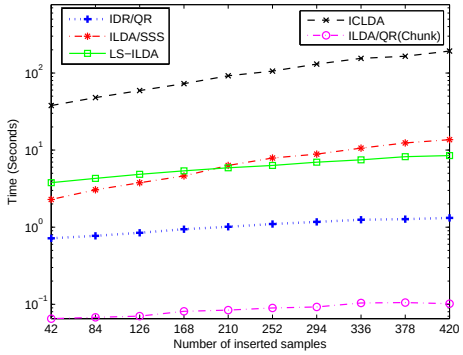


(e) CPU time for Wap

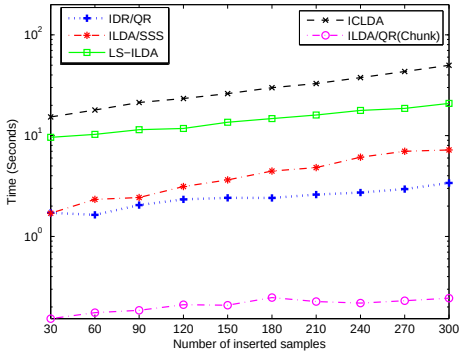


(f) CPU time for $AR_{50\times 40}$

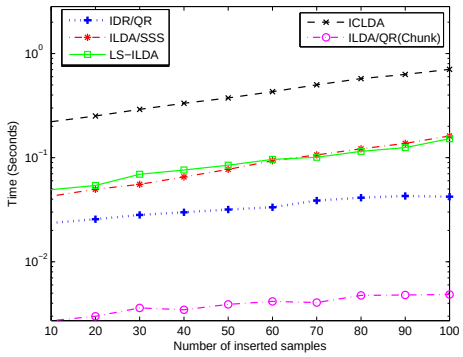
Figure 3.13: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)



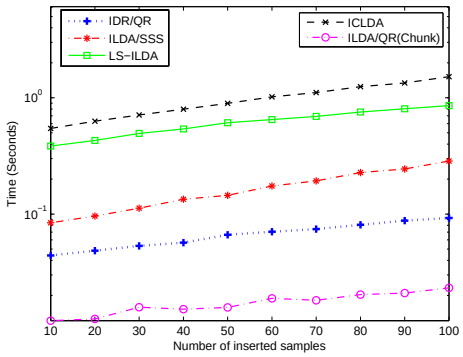
(a) CPU time for $AR_{50\times45}$



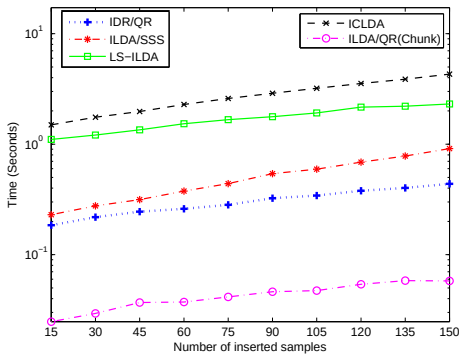
(b) CPU time for $Feret$



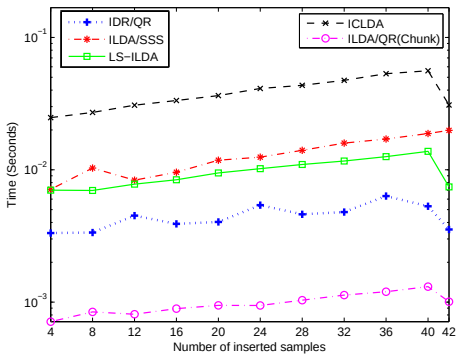
(c) CPU time for $ORL_{32\times32}$



(d) CPU time for $ORL_{64\times64}$

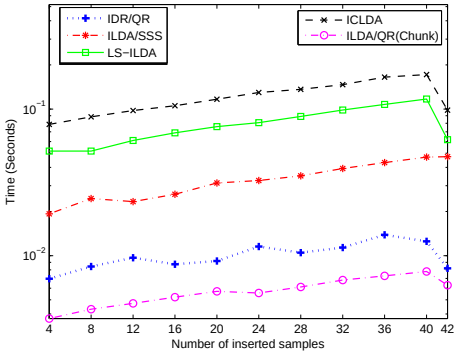


(e) CPU time for $Palmprint$

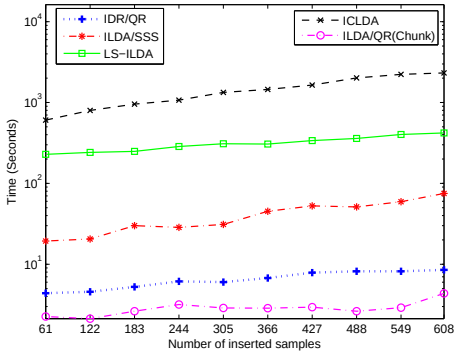


(f) CPU time for $Yale_{32\times32}$

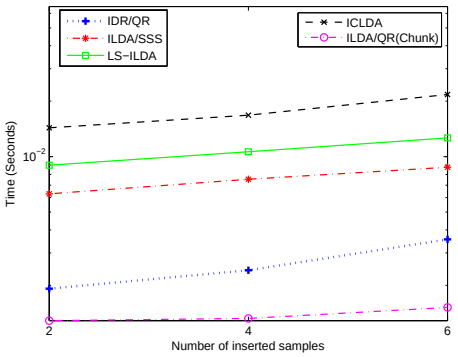
Figure 3.14: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)



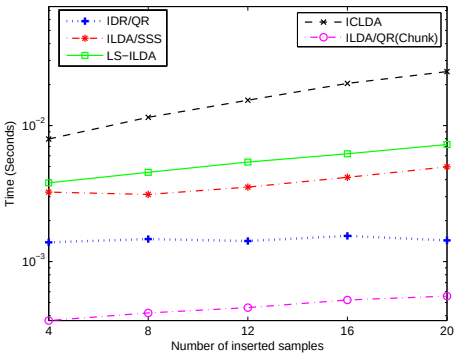
(a) CPU time for $Yale_{64\times64}$



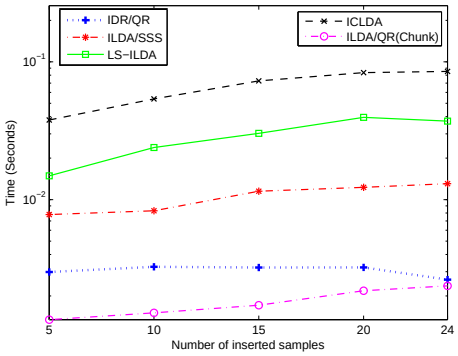
(b) CPU time for $YaleB$



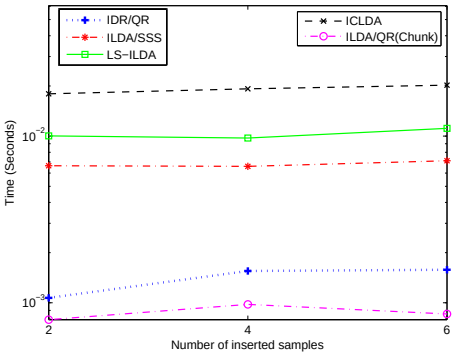
(c) CPU time for $Brain$



(d) CPU time for $Colon$

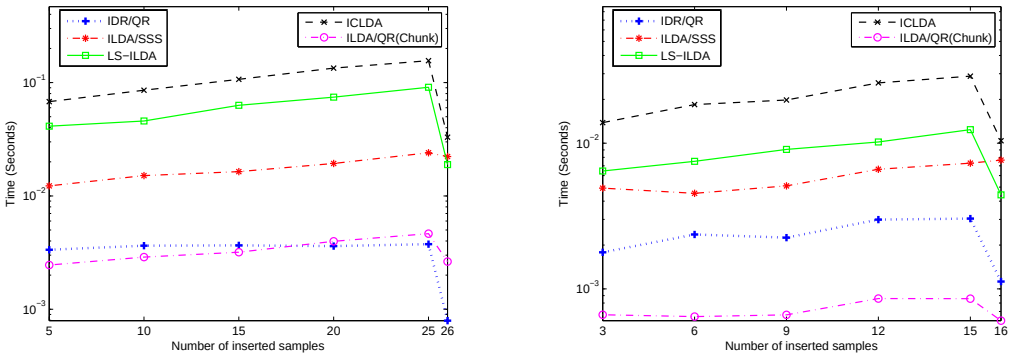


(e) CPU time for $Leukemia$



(f) CPU time for $Lymphoma$

Figure 3.15: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)



(a) CPU time for *Prostate* (b) CPU time for *SRBCT*

Figure 3.16: Comparing the CPU time of IDR/QR, ILDA/SSS, LS-ILDA, ICLDA, ILDA/QR and ILDA/QR(Chunk) (measured in log scale)

3.5 Conclusions

In this chapter, we have proposed a novel incremental linear discriminant analysis algorithm, called ILDA/QR. ILDA/QR incrementally updates the optimal transformation of LDA with exactness. In addition, ILDA/QR can easily handle not only the case that only one new sample is inserted (sequential ILDA/QR) but also the case that a chunk of new samples are added (chunk ILDA/QR). The computational complexity of one update in sequential ILDA/QR is $O(mn)$ and in chunk ILDA/QR is $O(mns)$, where there are n samples in m dimensions and s samples are added as a chunk. Experiments on several real-world datasets, show that ILDA/QR achieves the same accuracy as its batch version with far lower computational cost, which is consistent with our theoretical analysis. Compared with four recently proposed incremental LDA algorithms, IDR/QR, ILDA/SSS, LS-ILDA and ICLDA, our new proposed ILDA/QR algorithm works well. ILDA/QR is comparable with IDR/QR in terms of execution time whilst requires much less computation than the others, and is comparable with ICLDA from the perspective of classification accuracy whilst outperforms the others.

However, like LS-ILDA, ILDA/QR requires that samples in the training set

are linearly independent. Although most of our real-world datasets satisfy this condition, to extend ILDA/QR to the general case, regularization of the data matrix is a good choice. Similarly, to avoid the singularity of the scatter matrix, regularization is also introduced in IDR/QR. The big issue of the regularization problem is the selection of an appropriate regularization parameter. The work in Chapter 5 gives a novel solution to optimal parameter selection.

Chapter 4

Existing Regularized LDA

In this chapter, we will briefly outline the schemes of three existing regularized LDA methods: shrunken centroids regularized discriminant analysis (SCRDA)[39], regularized linear discriminant analysis (RLDA)[104, 54] and regularized discriminant analysis (RDA)[105].

For all these three existing regularized LDA approaches, given a candidate set for the regularization parameter, model selection by $\mathcal{K}$-fold cross-validation [31, 67] is applied on the training set to select an optimal parameter. Then the test error based on the tuning parameter is calculated by the corresponding regularized algorithm.

Remark 4.1. *Model selection by $\mathcal{K}$-fold cross-validation*

- *Divide the data set into mutually exclusive $\mathcal{K}$ folds of (approximately) equal size. Select the i-th (for $i = 1, \cdots, \mathcal{K}$) fold as the test set and all the other $\mathcal{K} - 1$ folds are used for training.*
- *For each regularization parameter, compute the cross-validation accuracy defined as the mean of the $\mathcal{K}$ accuracies, each of which is obtained by applying regularized LDA on the i-th (for $i = 1, \cdots, \mathcal{K}$) training set and test set.*
- *The optimal regularization parameter is the one that maximizes all the cross-validation accuracies with respect to the given regularization parameter set.*

4.1 Shrunken Centroids Regularized Discriminant Analysis (SCRDA)

The method of shrunken centroids regularized discriminant analysis (SCRDA)[39] generalizes the idea of nearest shrunken centroids (NSC) [85] into the classical discriminant analysis. Given the data matrix $A = \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} \in \mathbf{R}^{m\times n}$ which has been defined in Chapter 1, denote $cl(j)$ as the class label of data point a_j $(1 \leq j \leq n)$, i.e., $cl(j) = i$ if a_j belongs to the i-th cluster. In [39], Guo et al., would classify a data point x to a cluster i^* which maximizes the sample version discriminant function $d_i(x)$, that is,

$$i^* = \arg\max_{1\leq i\leq k} d_i(x),$$

where

$$d_i(x) = x^T \hat{S}_w^{-1} c_i - \frac{1}{2} c_i^T \hat{S}_w^{-1} c_i + \log \pi_i,$$

c_i is the local centroid of cluster i, π_i is the proportion of cluster i such that $\pi_1 + \cdots + \pi_k = 1$, e.g., $\pi_i = \frac{n_i}{n}$. $\hat{S}_w$ is the covariance matrix of data set A defined as

$$\hat{S}_w = \frac{1}{n} \sum_{i=1}^{k} \sum_{j\in\mathcal{N}_i} (a_j - c_i)(a_j - c_i)^T. \tag{4.1}$$

When the dimensionality of the data point is greater than the size of the sample, i.e., $m > n$, the covariance matrix $\hat{S}_w$ is singular and cannot be inverted. To resolve this singularity problem, instead of using $\hat{S}_w$ directly, the authors introduced

$$\hat{S}_\alpha = \alpha \hat{S}_w + (1-\alpha) I$$

for some α, $0 \leq \alpha < 1$. The corresponding regularized discriminant function is

$$\hat{d}_i(x) = x^T \hat{S}_\alpha^{-1} c_i - \frac{1}{2} c_i^T \hat{S}_\alpha^{-1} c_i + \log \pi_i. \tag{4.2}$$

Similar to the idea of NSC [85], the authors applied shrunken centroids to the regularized discriminant function (4.2). That is to shrink the centroids in (4.2)

before calculating the discriminant function, i.e.,

$$\hat{c}_i = \mathrm{sgn}(c_i)(|c_i| - \Delta)_+,$$

where $\Delta \geq 0$ is the shrinkage parameter, $\mathrm{sgn}(\cdot)$ is the sign function, $|\cdot|$ is the absolute value function. In addition to directly shrink the centroids , there are also two other ways, such as to shrink $\hat{S}_\alpha^{-1}c_i$, i.e.,

$$\overline{c}_i = \mathrm{sgn}(\hat{S}_\alpha^{-1}c_i)(|\hat{S}_\alpha^{-1}c_i| - \Delta)_+, \tag{4.3}$$

and the other way is to shrink $\hat{S}_\alpha^{-\frac{1}{2}}c_i$, i.e.,

$$\underline{c}_i = \mathrm{sgn}(\hat{S}_\alpha^{-\frac{1}{2}}c_i)(|\hat{S}_\alpha^{-\frac{1}{2}}c_i| - \Delta)_+.$$

In this thesis, we choose (4.3) as the way to shrink the centroids, which is the same as the authors did in [39].

Unlike other LDA approaches, there is no explicit expression of the optimal transformation, so the output of SCRDA shown in the following algorithm is classification accuracy.

Algorithm 4.1. *(SCRDA)*

Input: *Training set* $A \in \mathbf{R}^{m\times n}$ *with cluster label, test set* $B \in \mathbf{R}^{m\times \hat{n}}$ *with cluster label* $cl(j)$ $(j = 1, \cdots, \hat{n})$*, cluster number* k*, regularization parameter* (α, Δ)*.*

Output: accuracy*.*

Step 1. *Form proportion* π_i $(i = 1, \cdots, k)$ *of training data* A*.*

Step 2. *Compute centroids* c_i $(i = 1, \cdots, k)$ *of* A *and form its covariance matrix* $\hat{S}_w$ *by (4.1), compute*

$$\hat{S}_\alpha = \alpha\hat{S}_w + (1-\alpha)I.$$

Step 3. *Shrink* c_i *as*

$$\overline{c}_i = \mathrm{sgn}(\hat{S}_\alpha^{-1}c_i)(|\hat{S}_\alpha^{-1}c_i| - \Delta)_+, \quad i = 1, \cdots, k.$$

Step 4. *Compute*

$$d(j,i) = (B(:,j))^T\overline{c}_i - \frac{1}{2}c_i^T\overline{c}_i + \log\pi_i, \quad j = 1, \cdots, \hat{n}, \quad i = 1, \cdots, k.$$

Step 5. *Obtain* $cl_{\max}(j) = \arg\max\limits_{1\leq i\leq k} d(j,i)$*,* $j = 1, \cdots, \hat{n}$*.*

Step 6. *Compute* accuracy *by comparing* cl *with* $cl_{\max}$*.*

4.2 Regularized Linear Discriminant Analysis (RLDA)

Given a data matrix A as defined in Chapter 1, between-class, within-class and total scatter matrices are redefined as

$$\begin{aligned}\hat{S}_b &= \frac{1}{n}\sum_{i=1}^{k} n_i(c_i - c)(c_i - c)^T,\\ \hat{S}_w &= \frac{1}{n}\sum_{i=1}^{k}\sum_{j\in\mathcal{N}_i}(a_j - c_i)(a_j - c_i)^T,\\ \hat{S}_t &= \frac{1}{n}\sum_{i=1}^{n}(a_j - c)(a_j - c)^T.\end{aligned} \tag{4.4}$$

Correspondingly,

$$\begin{aligned}\hat{H}_b &= \frac{1}{\sqrt{n}}[\sqrt{n_1}(c_1 - c) \ \cdots \ \sqrt{n_k}(c_k - c)],\\ \hat{H}_w &= \frac{1}{\sqrt{n}}\left[\mathcal{A}_1 - c_1 e_1^T \ \cdots \ \mathcal{A}_k - c_k e_k^T\right],\\ \hat{H}_t &= \frac{1}{\sqrt{n}}(A - ce^T),\end{aligned} \tag{4.5}$$

and

$$\hat{S}_b = \hat{H}_b\hat{H}_b^T, \quad \hat{S}_w = \hat{H}_w\hat{H}_w^T, \quad \hat{S}_t = \hat{H}_t\hat{H}_t^T.$$

Regularized linear discriminant analysis (RLDA) [104, 54] aims to solve the following regularized optimization problem

$$G = \arg\max_{G\in\mathbf{R}^{m\times l}} \operatorname{trace}((G^T(\hat{S}_t + \mu I)G)^{-1}G^T\hat{S}_b G), \tag{4.6}$$

for some $\mu > 0$. The optimal solution of (4.6) is computed by the eigenvalue decomposition of $(\hat{S}_t + \mu I)^{-1}\hat{S}_b$.

Let $\hat{H}_t = U\Sigma V^T$ be the reduced SVD of $\hat{H}_t$, where $U \in \mathrm{R}^{m\times\gamma}$ and $V \in \mathrm{R}^{n\times\gamma}$ are column orthogonal, $\Sigma \in \mathrm{R}^{\gamma\times\gamma}$ with $\gamma = \operatorname{rank}(\hat{H}_t) = \operatorname{rank}(\hat{S}_t)$. Denote $U^\perp \in \mathbf{R}^{m\times(m-\gamma)}$ as the orthogonal complement of U, then for any $\mu > 0$, the equality

$$(\hat{S}_t + \mu I)^{-1}\hat{S}_b = U(\Sigma^2 + \mu I)^{-1}U^T\hat{S}_b$$

holds since $(U^{\perp})^T \hat{S}_b = 0$. Let y be any eigenvector of $(\hat{S}_t + \mu I)^{-1}\hat{S}_b$ corresponding to a nonzero eigenvalue λ, and $y = Ux$ for some x, then multiplying both sides of the following equation by U^T:

$$U(\Sigma^2 + \mu I)^{-1}U^T \hat{S}_b y = \lambda y,$$

we have

$$(\Sigma^2 + \mu I)^{-1}U^T \hat{S}_b(Ux) = \lambda U^T(Ux) = \lambda x.$$

Thus, computing the eigenvalue decomposition of $(\hat{S}_t + \mu I)^{-1}\hat{S}_b$ is reduced to compute the eigenvalue decomposition of $(\Sigma^2 + \mu I)^{-1}U^T \hat{S}_b U$. To further reduce the computational cost, denote $U_b \Sigma_b V_b$ as the SVD of $\hat{\Sigma}^{-\frac{1}{2}}U^T \hat{H}_b$, where $\hat{\Sigma} = \Sigma^2 + \mu I$, $U_b \in \mathbf{R}^{\gamma \times \gamma}$ and $V_b \in \mathbf{R}^{k \times k}$ are orthogonal, and $\Sigma_b \in \mathbf{R}^{\gamma \times k}$. Then we have

$$\begin{aligned}\hat{\Sigma}^{-1}U^T \hat{S}_b U &= \hat{\Sigma}^{-\frac{1}{2}}(\hat{\Sigma}^{-\frac{1}{2}}U^T \hat{H}_b)(\hat{\Sigma}^{-\frac{1}{2}}U^T \hat{H}_b)^T \hat{\Sigma}^{\frac{1}{2}} \\ &= (\hat{\Sigma}^{-\frac{1}{2}}U_b)\Sigma_b^2(\hat{\Sigma}^{-\frac{1}{2}}U_b)^{-1}.\end{aligned}$$

That is, $\hat{\Sigma}^{-\frac{1}{2}}U_b$ diagonalizes matrix $\hat{\Sigma}^{-1}U^T S_b U$. Thus, the columns of $\hat{\Sigma}^{-\frac{1}{2}}U_b$ form the eigenvectors of $\hat{\Sigma}^{-1}U^T \hat{S}_b U$.

The algorithm of RLDA is shown in the following:

Algorithm 4.2. *(RLDA)*

Input: *Data matrix $A \in \mathbf{R}^{m \times n}$ with cluster label, cluster number k, regularization parameter μ.*

Output: *Optimal transformation $G \in \mathbf{R}^{m \times k}$*

Step 1. *Construct $\hat{H}_b \in \mathbf{R}^{m \times k}$ and $\hat{H}_t \in \mathbf{R}^{m \times n}$ from (4.5).*

Step 2. *Compute the reduced SVD of $\hat{H}_t$ as*

$$\hat{H}_t = U\Sigma V^T,$$

where $U \in \mathbf{R}^{m \times \gamma}$, $\Sigma \in \mathbf{R}^{\gamma \times \gamma}$, $V \in \mathbf{R}^{n \times \gamma}$, $\gamma = \operatorname{rank}(\hat{H}_t)$.

Step 3. $\hat{\Sigma} = \Sigma^2 + \mu I$.

Step 4. *Compute SVD of $\hat{\Sigma}^{-\frac{1}{2}}U^T H_b$ as*

$$\hat{\Sigma}^{-\frac{1}{2}}U^T H_b = U_b \Sigma_b V_b^T, \quad U_b \in \mathbf{R}^{\gamma \times k}, \quad V_b \in \mathbf{R}^{k \times k}, \quad \Sigma \in \mathbf{R}^{k \times k}.$$

Step 5. $G = U\hat{\Sigma}^{-1/2}U_b$.

4.3 Regularized Discriminant Analysis (RDA)

Different from SCRDA and RLDA, Friedman[34] proposed a compromise between LDA and quadratic discriminant analysis(QDA), called regularized discriminant analysis(RDA), which allows one to shrink the separate covariances of QDA toward a common covariance as in LDA by employing regularization techniques. However, the computational cost of the model selection in [34] is high, especially, when the dimensionality m is large. In [105], Ye et. al., extended the applicability of RDA to high dimensional, low sample size data.

Consider the data matrix $A = \begin{bmatrix} a_1 & \cdots & a_n \end{bmatrix} \in \mathbf{R}^{m\times n}$ defined in Chapter 1, the notations $\hat{S}_w$, $\hat{S}_t$, $\hat{S}_b$ and $\hat{H}_b$ are the same as RLDA in section 4.2. Denote S_i as the covariance matrix of the i-th cluster, i.e.,

$$S_i = \frac{1}{n_i}\sum_{j\in\mathcal{N}_i}(a_j - c_i)(a_j - c_i)^T,$$

then the regularized class covariance matrix,

$$\hat{S}_i = \beta(\tau S_i + (1-\tau)\hat{S}_t) + (1-\beta)I$$

was used in [105] to overcome the singularity problem. A data point x is classified to class i^* if

$$i^* = \arg\min_{1\le i\le k} d_i(x), \tag{4.7}$$

where

$$d_i(x) = (x - c_i)^T\hat{S}_i^{-1}(x - c_i) + \log|\hat{S}_i|, \tag{4.8}$$

$|\cdot|$ denotes the matrix determinant.

Let $\hat{H}_t = U\Sigma V^T$ be the reduced SVD of $\hat{H}_t$, where $U \in \mathbf{R}^{m\times\gamma}$ and $V \in \mathbf{R}^{n\times\gamma}$ are column orthogonal, $\Sigma \in \mathbf{R}^{\gamma\times\gamma}$ is diagonal and $\gamma = \text{rank}(\hat{H}_t) = \text{rank}(\hat{S}_t)$. Denote $U^\perp \in \mathbf{R}^{m\times(m-\gamma)}$ as the orthogonal complement of U, then

$$\hat{S}_t = \begin{bmatrix} U & U^\perp \end{bmatrix}\begin{bmatrix} \Sigma^2 & \\ & 0 \end{bmatrix}\begin{bmatrix} U & U^\perp \end{bmatrix}^T$$

is the eigenvalue decomposition of $\hat{S}_t$. By knowing that $(U^{\perp})^T S_i = 0$, $\hat{S}_i$ can be expressed as

$$\hat{S}_i = \begin{bmatrix} U & U^{\perp} \end{bmatrix} \begin{bmatrix} M_i & \\ & (1-\beta)I_{m-\gamma} \end{bmatrix} \begin{bmatrix} U & U^{\perp} \end{bmatrix}^T, \tag{4.9}$$

where

$$M_i = \beta(\tau U^T S_i U + (1-\tau)\Sigma^2) + (1-\beta)I_{\gamma}. \tag{4.10}$$

Then the discriminant function in (4.8) is changed to be:

$$\begin{aligned} d_i(x) = (x-c_i)^T U M_i^{-1} U^T (x-c_i) + \log|M_i| + \log((1-\beta)^{m-\gamma}) \\ + (1-\beta)^{-1}(x-c_i)^T U^{\perp}(U^{\perp})^T (x-c_i). \end{aligned} \tag{4.11}$$

Since $(U^{\perp})^T \hat{S}_b = 0$, i.e., $(U^{\perp})^T \hat{H}_b = 0$, the last term of $d_i(x)$ in (4.11) is a constant for different i. Therefore, the classification rule in (4.7) is equivalent to

$$i^* = \arg\min_{1 \le i \le k} \hat{d}_i(x),$$

where

$$\begin{aligned} \hat{d}_i(x) &= (x-c_i)^T U M_i^{-1} U^T (x-c_i) + \log|M_i| \\ &= (\hat{x}-\hat{c}_i) M_i^{-1} (\hat{x}-\hat{c}_i) + \log|M_i|, \end{aligned}$$

with $\hat{x} = U^T x$, and $\hat{c}_i = U^T c_i$. Denote

$$\hat{H}_i = \frac{1}{\sqrt{n_i}}(\hat{\mathcal{A}}_i - \hat{c}_i e_i^T), \tag{4.12}$$

where $\hat{\mathcal{A}}_i = U^T \mathcal{A}_i$, $\mathcal{A}_i$ is the data set of the i-th class, we have

$$U^T S_i U = \hat{H}_i \hat{H}_i^T.$$

Denote $\Sigma_{\tau\beta} = (1-\tau)\beta\Sigma^2 + (1-\beta)I$, then M_i in (4.10) becomes

$$M_i = \beta\tau \hat{H}_i \hat{H}_i^T + \Sigma_{\tau\beta} = \Sigma_{\tau\beta}^{\frac{1}{2}}(X_i X_i^T + I_{\gamma})\Sigma_{\tau\beta}^{\frac{1}{2}},$$

where $X_i = \sqrt{\tau\beta}\Sigma_{\tau\beta}^{-\frac{1}{2}} \hat{H}_i$. It follows from Sherman-Woodbury-Morrison formula[38] that

$$M_i^{-1} = \Sigma_{\tau\beta}^{-\frac{1}{2}}(I_{\gamma} - X_i(I_{n_i} + X_i^T X_i)^{-1} X_i^T)\Sigma_{\tau\beta}^{-\frac{1}{2}}. \tag{4.13}$$

Thus, the first term of $\hat{d}_i(x)$ can be computed as

$$\begin{aligned}&(\hat{x}-\hat{c}_i)M_i^{-1}(\hat{x}-\hat{c}_i)\\ =&(\hat{x}-\hat{c}_i)\Sigma_{\tau\beta}^{-1}(\hat{x}-\hat{c}_i)-(\hat{x}-\hat{c}_i)\Sigma_{\tau\beta}^{-\frac{1}{2}}(X_i(I_{n_i}+X_i^TX_i)^{-1}X_i^T)\Sigma_{\tau\beta}^{-\frac{1}{2}}(\hat{x}-\hat{c}_i),\end{aligned}$$

and in the second term,

$$|M_i|=|X_iX_i^T+I_\gamma||\Sigma_{\tau\beta}|=|X_i^TX_i+I_{n_i}||\Sigma_{\tau\beta}|. \tag{4.14}$$

Like SCRDA in section 4.1, there is no explicit expression of optimal transformation matrix, in the following RDA algorithm, the output is classification accuracy. Likewise, we use notation $cl(\cdot)$ to denote the class label of data sample.

Algorithm 4.3. *(RDA)*

Input: *Training set* $A\in\mathbf{R}^{m\times n}$ *with cluster label, test set* $B\in\mathbf{R}^{m\times\hat{n}}$ *with cluster label* $cl(j)$ $(j=1,\cdots,\hat{n})$, *cluster number* k, *regularization parameters* (β,τ).

Output: accuracy.

Step 1. *Construct local centroids* c_i $(i=1,\cdots,k)$ *and* $\hat{H}_t$ *of* A.

Step 2. *Compute the reduced SVD of* $\hat{H}_t$ *as*

$$\hat{H}_t=U\Sigma V^T,$$

where $U\in\mathbf{R}^{m\times\gamma}$, $V\in\mathbf{R}^{n\times\gamma}$ *and* $\Sigma\in\mathbf{R}^{\gamma\times\gamma}$, $\gamma=\mathrm{rank}(\hat{H}_t)$.

Step 3. $\hat{A}=U^TA$, $\hat{B}=U^TB$, $\hat{c}_i=U^Tc_i$ $(i=1,\cdots,k)$.

Step 5. *Form* $\hat{H}_i$ $(i=1,\cdots,k)$ *by (4.12)*

Step 6. $\Sigma_{\beta,\tau}=(1-\beta)\tau\Sigma^2+(1-\tau)I$.

Step 7. *Compute*

$$X_i=\sqrt{\beta\tau}\Sigma_{\beta,\tau}^{-\frac{1}{2}}\hat{H}_i,\quad |M_i|=|X_i^TX_i+I_{n_i}||\Sigma_{\beta,\tau}|,\quad i=1,\cdots,k,$$

$$M_i^{-1}=\Sigma_{\beta,\tau}^{-\frac{1}{2}}(I_\gamma-X_i(I_{n_i}+X_i^TX_i)^{-1}X_i^T)\Sigma_{\beta,\tau}^{-\frac{1}{2}},\quad i=1,\cdots,k.$$

Step 8. *Compute*

$$d(j,i)=(\hat{B}(:,j)-\hat{c}_i)M_i^{-1}(\hat{B}(:,j)-\hat{c}_i)+\log|M_i|,\quad j=1,\cdots,\hat{n},\quad i=1,\cdots,k.$$

Step 9. *Obtain* $cl_{\max}(j)=\arg\max\limits_{1\le i\le k}d(j,i)$, $j=1,\cdots,\hat{n}$.

Step 10. *Compute* accuracy *by comparing* cl *with* $cl_{\max}$

Chapter 5

New Regularized OLDA

Classical linear discriminant analysis (LDA) is not applicable for small sample size problems due to the singularity of the scatter matrices involved. Regularized LDA as a well-known extension of the classical LDA provides a simple strategy to overcome the singularity problem by applying a regularization term. The great advantage of regularized LDA over some other extensions of classical LDA is that it captures essential features of the training data without discarding some useful information, thus usually obtaining a higher classification accuracy given an appropriate regularization parameter. Although regularized LDA has been studied by some researchers as shown in Chapter 4, it is still lack of a mathematical theory for selecting an optimal regularization parameter. The work in this chapter is to fill this gap.

We first characterize all solutions of orthogonal LDA (OLDA):

$$G = \arg \max_{G \in \mathbf{R}^{m \times l},\ G^T G = I} \operatorname{trace}((G^T S_t G)^{(+)} G^T S_b G), \tag{5.1}$$

and all solutions of regularized OLDA (ROLDA):

$$G = \arg \max_{G \in \mathbf{R}^{m \times l},\ G^T G = I} \operatorname{trace}((G^T (S_t + \lambda I) G)^{-1} G^T S_b G), \tag{5.2}$$

for some $\lambda > 0$, then establish the intrinsic relationship between OLDA and ROLDA. Based on this relationship we find a mathematical criterion for selecting the regularization parameter λ in ROLDA and consequently develop a new

regularized orthogonal linear discriminant analysis method, in which no candidate set of regularization parameter is needed.

The rest of this chapter is organized as follows. Some useful supporting lemmas that are critical for subsequent discussions are given in Section 5.1. Theoretical analysis of our new proposed OLDA and ROLDA methods is shown in Section 5.2 and algorithm depiction of both methods is shown in Section 5.3. Numerical results and conclusions are presented in Section 5.4 and Section 5.5, respectively.

5.1 Preliminaries

Lemma 5.1. *Let* $X,\ Z \in \mathbf{R}^{\mu\times\mu}$ *be symmetric positive definite. Let* $W \in \mathbf{R}^{\mu\times\nu}$ *and* $W \neq 0$. *Then*

$$\operatorname{trace}((W^TXW)^{(+)}(W^TZW)) \leq \operatorname{trace}(X^{-1}Z),$$

and the equality holds if and only if

$$\operatorname{rank}(W) = \mu.$$

Proof. We can assume without loss of generality that $W = \begin{bmatrix} I_{\mu_1} & 0 \\ 0 & 0 \end{bmatrix}$, $\mu_1 = \operatorname{rank}(W)$. Denote

$$X = \begin{bmatrix} X_{1,1} & X_{1,2} \\ X_{1,2}^T & X_{2,2} \end{bmatrix},\ Z = \begin{bmatrix} Z_{1,1} & Z_{1,2} \\ Z_{1,2}^T & Z_{2,2} \end{bmatrix},$$

where $X_{1,1},\ Z_{1,1} \in \mathbf{R}^{\mu_1\times\mu_1},\ X_{2,2},\ Z_{2,2} \in \mathbf{R}^{(\mu-\mu_1)\times(\mu-\mu_1)}$. Then,

$$\operatorname{trace}((W^TXW)^{(+)}(W^TZW)) = \operatorname{trace}(X_{1,1}^{-1}Z_{1,1}),$$

and

$$\begin{aligned}
&\operatorname{trace}(X^{-1}Z) \\
&= \operatorname{trace}\left(\begin{bmatrix} X_{1,1} & X_{1,2} \\ X_{1,2}^T & X_{2,2} \end{bmatrix}^{-1}\begin{bmatrix} Z_{1,1} & Z_{1,2} \\ Z_{1,2}^T & Z_{2,2} \end{bmatrix}\right) \\
&= \operatorname{trace}(X_{1,1}^{-1}Z_{1,1}) + \operatorname{trace}((Z_{2,2} - X_{1,2}^TX_{1,1}^{-1}Z_{1,2} - Z_{1,2}^TX_{1,1}^{-1}X_{1,2} + X_{1,2}^TX_{1,1}^{-1}Z_{1,1}X_{1,1}^{-1}X_{1,2}) \\
&\qquad \times (X_{2,2} - X_{1,2}^TX_{1,1}^{-1}X_{1,2})^{-1}),
\end{aligned}$$

where the second equality holds due to the same method as in (3.1). Note that X and Z are positive definite, and $Z_{2,2}-X_{1,2}^T X_{1,1}^{-1} Z_{1,2}-Z_{1,2}^T X_{1,1}^{-1} X_{1,2}+X_{1,2}^T X_{1,1}^{-1} Z_{1,1} X_{1,1}^{-1} X_{1,2}$ is the principal submatrix of

$$\begin{bmatrix} I & \\ -X_{1,2}^T X_{1,1}^{-1} & I \end{bmatrix} \begin{bmatrix} Z_{1,1} & Z_{1,2} \\ Z_{1,2}^T & Z_{2,2} \end{bmatrix} \begin{bmatrix} I & -X_{1,1}^{-1} X_{1,2} \\ & I \end{bmatrix},$$

so $X_{2,2}-X_{1,2}^T X_{1,1}^{-1} X_{1,2}$ and $Z_{2,2}-X_{1,2}^T X_{1,1}^{-1} Z_{1,2}-Z_{1,2}^T X_{1,1}^{-1} X_{1,2}+X_{1,2}^T X_{1,1}^{-1} Z_{1,1} X_{1,1}^{-1} X_{1,2}$ are positive definite if they are not vanish, that is, if $\mu > \mu_1$. This yields that

$$\begin{aligned} &\operatorname{trace}((Z_{2,2} - X_{1,2}^T X_{1,1}^{-1} Z_{1,2} - Z_{1,2}^T X_{1,1}^{-1} X_{1,2} + X_{1,2}^T X_{1,1}^{-1} Z_{1,1} X_{1,1}^{-1} X_{1,2}) \\ &\quad \times (X_{2,2} - X_{1,2}^T X_{1,1}^{-1} X_{1,2})^{-1}) > 0 \end{aligned}$$

if $\mu > \mu_1$.

Therefore we obtain

$$\operatorname{trace}((W^T X W)^{(+)} (W^T Z W)) \leq \operatorname{trace}(X^{-1} Z),$$

and the equality holds if and only if

$$\mu = \mu_1, \quad \text{i.e.,} \quad \operatorname{rank}(W) = \mu.$$

□

Lemma 5.2. *Let $X \in \mathbf{R}^{\mu_1 \times \nu}$, $Y \in \mathbf{R}^{\mu_2 \times \nu}$, and $Z \in \mathbf{R}^{\mu_3 \times \nu}$, then*

$$\operatorname{rank} \begin{bmatrix} X^T X & Y^T Y \end{bmatrix} = \operatorname{rank} \begin{bmatrix} X \\ Y \end{bmatrix},$$

and

$$\operatorname{rank} \begin{bmatrix} X^T X & Y^T Y + Z^T Z \end{bmatrix} = \operatorname{rank} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}.$$

Proof. Note that

$$\begin{bmatrix} X^T X & Y^T Y \end{bmatrix} = \begin{bmatrix} X^T & Y^T \end{bmatrix} \begin{bmatrix} X & \\ & Y \end{bmatrix},$$

so

$$\operatorname{rank}\begin{bmatrix} X^TX & Y^TY \end{bmatrix} \leq \operatorname{rank}\begin{bmatrix} X \\ Y \end{bmatrix},$$

and the other side inequality directly follows from

$$\operatorname{rank}\begin{bmatrix} X^TX & Y^TY \end{bmatrix} \geq \operatorname{rank}(X^TX + Y^TY) = \operatorname{rank}\left(\begin{bmatrix} X^T & Y^T \end{bmatrix}\begin{bmatrix} X \\ Y \end{bmatrix}\right)$$
$$= \operatorname{rank}\begin{bmatrix} X \\ Y \end{bmatrix}.$$

The second equality can be derived similarly. □

Lemma 5.3. *Let scatter matrices S_t and S_b be defined in (1.1). Then*

$$\operatorname{trace}(S_t^{(+)}S_b) = \max_G \operatorname{trace}((G^TS_tG)^{(+)}G^TS_bG) = \max_{G^TG=I} \operatorname{trace}((G^TS_tG)^{(+)}G^TS_bG),$$

and

$$\operatorname{trace}((S_t+\lambda I)^{-1}S_b) = \max_G \operatorname{trace}((G^T(S_t+\lambda I)G)^{-1}G^TS_bG)$$
$$= \max_{G^TG=I} \operatorname{trace}((G^T(S_t+\lambda I)G)^{-1}G^TS_bG), \quad \forall \lambda > 0.$$

Proof. By applying GSVD [72] in Lemma 3.5 to matrices $H_b \in \mathbf{R}^{m\times k}$ and $H_w \in \mathbf{R}^{m\times n}$ defined in (1.3), we have

$$\Phi^TS_b\Phi = \Phi^TH_bUU^TH_b^T\Phi$$
$$= \begin{bmatrix} I_p & & & \\ & \Theta^2 & & \\ & & 0_{\gamma-p-s} & \\ & & & 0_{m-\gamma} \end{bmatrix},$$

and

$$\Phi^TS_w\Phi = \Phi^TH_wUU^TH_w^T\Phi$$
$$= \begin{bmatrix} 0_p & & & \\ & \Xi^2 & & \\ & & I_{\gamma-p-s} & \\ & & & 0_{m-\gamma} \end{bmatrix},$$

where $\Phi \in \mathbf{R}^{m\times m}$ is nonsingular, $U \in \mathbf{R}^{k\times k}$ and $V \in \mathbf{R}^{n\times n}$ are orthogonal, $\Theta = \mathrm{diag}(\theta_1, \cdots, \theta_s) \in \mathbf{R}^{s\times s}$ and $\Xi = \mathrm{diag}(\xi_1, \cdots, \xi_s) \in \mathbf{R}^{s\times s}$ are diagonal as defined in Lemma 3.5, with $\gamma = \mathrm{rank}\begin{bmatrix} H_b & H_w \end{bmatrix} = \mathrm{rank}(S_t)$, $p = \gamma - \mathrm{rank}(H_w)$, and $s = \mathrm{rank}(H_b) + \mathrm{rank}(H_w) - \gamma$.

Next, from equality

$$S_t = S_w + S_b,$$

we have

$$\Phi^T S_t \Phi = \Phi^T S_b \Phi + \Phi^T S_w \Phi = \begin{bmatrix} I_\gamma & \\ & 0_{m-\gamma} \end{bmatrix}.$$

Thus, by the nonsingularity of Φ, the following equality holds:

$$\mathrm{trace}(S_t^{(+)} S_b) = \mathrm{trace}((\Phi^T S_t \Phi)^{(+)} \Phi^T S_b \Phi) = \mathrm{trace} \begin{bmatrix} I_p & \\ & \Theta^2 \end{bmatrix} = p + \sum_{i=1}^{s} \theta_i^2.$$

For any $G \in \mathbf{R}^{m\times l}$, we have

$$G^T S_t G = G^T \Phi^{-T} \Phi^T S_t \Phi \Phi^{-1} G = \mathcal{G}^T \begin{bmatrix} I_\gamma & \\ & 0_{m-\gamma} \end{bmatrix} \mathcal{G},$$

$$G^T S_b G = G^T \Phi^{-T} \Phi^T S_b \Phi \Phi^{-1} G = \mathcal{G}^T \begin{bmatrix} I_p & & & \\ & \Theta^2 & & \\ & & 0_{\gamma-p-s} & \\ & & & 0_{m-\gamma} \end{bmatrix} \mathcal{G},$$

where $\mathcal{G} = \Phi^{-1} G \in \mathbf{R}^{m\times l}$. Let

$$\mathcal{G} = \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \\ \mathcal{G}_3 \\ \mathcal{G}_4 \end{bmatrix}, \quad \mathcal{G}_1 \in \mathbf{R}^{p\times l}, \quad \mathcal{G}_2 \in \mathbf{R}^{s\times l}, \quad \mathcal{G}_3 \in \mathbf{R}^{(\gamma-p-s)\times l}, \quad \mathcal{G}_4 \in \mathbf{R}^{(m-\gamma)\times l},$$

be the partition of $\mathcal{G}$. It follows that

$$G^T S_t G = \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \\ \mathcal{G}_3 \end{bmatrix}^T \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \\ \mathcal{G}_3 \end{bmatrix},$$

and

$$G^T S_b G = \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}^T \begin{bmatrix} I_p & \\ & \Theta^2 \end{bmatrix} \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}.$$

Hence, Lemma 3.1 and Lemma 5.1 result in

$$\begin{aligned}
&\text{trace}((G^T S_t G)^{(+)} G^T S_b G) \\
=&\text{trace}\left(\left(\begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}^T \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix} + \mathcal{G}_3^T \mathcal{G}_3\right)^{(+)} \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}^T \begin{bmatrix} I_p & \\ & \Theta^2 \end{bmatrix} \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}\right) \\
\leq&\text{trace}\left(\left(\begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}^T \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}\right)^{(+)} \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}^T \begin{bmatrix} I_p & \\ & \Theta^2 \end{bmatrix} \begin{bmatrix} \mathcal{G}_1 \\ \mathcal{G}_2 \end{bmatrix}\right) \\
\leq&\text{trace}\begin{bmatrix} I_p & \\ & \Theta^2 \end{bmatrix} \\
=&p + \sum_{i=1}^{s} \theta_i^2.
\end{aligned} \tag{5.3}$$

Let

$$\mathcal{G} = \begin{bmatrix} I_l \\ 0 \end{bmatrix} \in \mathbf{R}^{m \times l}$$

with $l \geq p + s$, then $G = \Phi\mathcal{G} = \Phi_1$ is the optimal transformation matrix such that

$$\text{trace}(S_t^{(+)} S_b) = \text{trace}((G^T S_t G)^{(+)} G^T S_b G),$$

where $\Phi_1 \in \mathbf{R}^{m \times l}$ is the submatrix of Φ that consists of its first l columns.

Compute the economic QR factorization of Φ_1 as

$$\Phi_1 = \Psi_1 R,$$

where $\Psi_1 \in \mathbf{R}^{m \times l}$ is column orthogonal and $R \in \mathbf{R}^{l \times l}$ is upper triangular and nonsingular, then $G = \Psi_1$ is the optimal column orthogonal transformation matrix.

For the regularized optimization problem, since

$$\begin{aligned}
S_t + \lambda I &= S_b + S_w + \lambda I \\
&= H_b H_b^T + \begin{bmatrix} H_w & \sqrt{\lambda} I \end{bmatrix} \begin{bmatrix} H_w & \sqrt{\lambda} I \end{bmatrix}^T \\
&= H_b H_b^T + \bar{H}_w \bar{H}_w^T,
\end{aligned}$$

where $\bar{H}_w = \begin{bmatrix} H_w & \sqrt{\lambda}I \end{bmatrix}$, we can apply GSVD to matrices H_b and $\bar{H}_w$, then the results will be obtained. The proof for regularized optimization problems just follows the above process, we omit it here. □

Lemma 5.3 implies that $G \in \mathbf{R}^{m\times l}$ is an optimal solution of OLDA (5.1) if $\text{trace}((G^T S_t G)^{(+)} G^T S_b G)$ achieves its maximum value $\text{trace}(S_t^{(+)} S_b)$, and $G^\lambda \in \mathbf{R}^{m\times l}$ is an optimal solution of ROLDA (5.2) if $\text{trace}(((G^\lambda)^T(S_t+\lambda I)G^\lambda)^{-1}(G^\lambda)^T S_b G^\lambda)$ achieves its maximum value $\text{trace}((S_t+\lambda I)^{-1}S_b)$. This lemma is crucial for the general solution acquisition in OLDA and ROLDA.

Lemma 5.4. *Let $X \in \mathbf{R}^{\mu_1\times\nu}$, $Y \in \mathbf{R}^{\mu_2\times\nu}$, then*

$$\begin{cases} \text{rank}\begin{bmatrix} X \\ Y \end{bmatrix} = \text{rank}(X) + \text{rank}(Y), \\ \text{rank}(X) = \mu_1, \end{cases} \tag{5.4}$$

is equivalent to

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} Z_{1,1} & Z_{1,2} \\ 0 & Z_{2,2} \end{bmatrix} F,$$

where $Z_{1,1} \in \mathbf{R}^{\mu_1\times\mu_1}$, $Z_{1,2} \in \mathbf{R}^{\mu_1\times(\nu-\mu_1)}$, $Z_{2,2} \in \mathbf{R}^{\mu_2\times(\nu-\mu_1)}$, $F \in \mathbf{R}^{\nu\times\nu}$ is orthogonal and $\text{rank}(Z_{1,1}) = \mu_1$.

Proof. Decompose Y^T as,

$$Y^T = \mathcal{F}^T \begin{bmatrix} 0 \\ \mathcal{Z}_{2,2}^T \end{bmatrix}, \tag{5.5}$$

where $\mathcal{F} \in \mathbf{R}^{\nu\times\nu}$ is orthogonal, $\mathcal{Z}_{2,2} \in \mathbf{R}^{\mu_2\times s}$ is of full column rank, s is the rank of Y. Then

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \mathcal{Z}_{1,1} & \mathcal{Z}_{1,2} \\ 0 & \mathcal{Z}_{2,2} \end{bmatrix} \mathcal{F},$$

if we denote

$$\begin{bmatrix} \mathcal{Z}_{1,1} & \mathcal{Z}_{1,2} \end{bmatrix} = X\mathcal{F}^T,$$

where $\mathcal{Z}_{1,1} \in \mathbf{R}^{\mu_1 \times (\nu - s)}$, $\mathcal{Z}_{1,2} \in \mathcal{R}^{\mu_1 \times s}$. Therefore,

$$\operatorname{rank} \begin{bmatrix} X \\ Y \end{bmatrix} = \operatorname{rank}(\mathcal{Z}_{1,1}) + \operatorname{rank}(\mathcal{Z}_{2,2}) = \operatorname{rank}(\mathcal{Z}_{1,1}) + \operatorname{rank}(Y),$$

which together with (5.4) yields that $\operatorname{rank}(\mathcal{Z}_{1,1}) = \mu_1$. Permute the columns of matrix $\mathcal{Z}_{1,1}$ such that the first μ_1 columns are linearly independent and adopt the corresponding matrix row permutation to $\mathcal{F}$, which results in the new decomposition as

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} Z_{1,1} & Z_{1,2} \\ 0 & Z_{2,2} \end{bmatrix} F,$$

where $Z_{1,1} \in \mathbf{R}^{\mu_1 \times \mu_1}$ is nonsingular, $Z_{1,2} \in \mathbf{R}^{\mu_1 \times (\nu - \mu_1)}$, $Z_{2,2} \in \mathbf{R}^{\mu_2 \times (\nu - \mu_1)}$, and $F \in \mathbf{R}^{\nu \times \nu}$ is orthogonal.

The other direction is trivial, we omit it here. □

Remark 5.1. *The decomposition of Y^T in (5.5) can be accomplished in the same way as QR factorization except that columns are annihilated above the diagonal.*

Lemma 5.5. *Let $X \in \mathbf{R}^{\mu \times \mu}$ be an orthogonal matrix that has partition*

$$X = \begin{bmatrix} X_{1,1} & X_{1,2} \\ X_{2,1} & X_{2,2} \end{bmatrix},$$

where $X_{1,1} \in \mathbf{R}^{(\mu - \nu) \times \nu}$, $X_{2,1} \in \mathbf{R}^{\nu \times \nu}$, $X_{1,2} \in \mathbf{R}^{(\mu - \nu) \times (\mu - \nu)}$ and $X_{2,2} \in \mathbf{R}^{\nu \times (\mu - \nu)}$. If $X_{2,1}$ is nonsingular, then $X_{1,2}$ is nonsingular, and vice versa.

Proof. Suppose $X_{2,1}$ is nonsingular, but $X_{1,2}$ is singular. There exists a nonzero vector $z \in \mathbf{R}^{\mu - \nu}$ such that $X_{1,2} z = 0$. Thus, follows from the orthogonality of X, we have

$$\begin{aligned} z^T z &= \begin{bmatrix} 0 & z^T \end{bmatrix} X^T X \begin{bmatrix} 0 \\ z \end{bmatrix} \\ &= \begin{bmatrix} 0 & z^T \end{bmatrix} \begin{bmatrix} X_{1,1} & X_{1,2} \\ X_{2,1} & X_{2,2} \end{bmatrix}^T \begin{bmatrix} X_{1,1} & X_{1,2} \\ X_{2,1} & X_{2,2} \end{bmatrix} \begin{bmatrix} 0 \\ z \end{bmatrix} \\ &= \begin{bmatrix} z^T X_{1,2}^T & z^T X_{2,2}^T \end{bmatrix} \begin{bmatrix} X_{1,2} z \\ X_{2,2} z \end{bmatrix} \\ &= z^T X_{2,2}^T X_{2,2} z, \end{aligned}$$

which results in

$$X_{2,2}z \neq 0. \tag{5.6}$$

Since $\begin{bmatrix} X_{1,1} \\ X_{2,1} \end{bmatrix}$ is column orthogonal to $\begin{bmatrix} X_{1,2} \\ X_{2,2} \end{bmatrix}$, i.e.,

$$X_{1,1}^T X_{1,2} + X_{2,1}^T X_{2,2} = 0,$$

multiplying z on both sides of the above equality yields

$$X_{2,1}^T X_{2,2} z = 0,$$

which contradicts nonequality (5.6) as $X_{2,1}$ is nonsingular.

Similarly, we can prove that if $X_{1,2}$ is nonsingular, then $X_{2,1}$ is nonsingular. □

Lemma 5.6. *[81] Let* $X,\ \mathcal{X} \in \mathrm{R}^{\mu\times\nu}$ *with* $\mathrm{rank}(X) = \mathrm{rank}(\mathcal{X}) = \nu$ *and*

$$\left\|X^{(+)}\right\|_2 \|X - \mathcal{X}\|_2 < 1.$$

Let

$$X = QR, \quad \mathcal{X} = \mathcal{Q}\mathcal{R}$$

be the economic QR factorizations of X *and* $\mathcal{X}$*, respectively, where* $R,\ \mathcal{R} \in \mathrm{R}^{\nu\times\nu}$ *are upper triangular with all diagonal elements being positive, and* $Q,\ \mathcal{Q} \in \mathrm{R}^{\mu\times\nu}$ *are column orthogonal, i.e.,*

$$Q^T Q = \mathcal{Q}^T \mathcal{Q} = I.$$

Then

$$\|Q - \mathcal{Q}\|_F \leq (1+\sqrt{2}) \frac{\left\|X^{(+)}\right\|_2}{1 - \left\|X^{(+)}\right\|_2 \|X - \mathcal{X}\|_2} \|X - \mathcal{X}\|_F, \tag{5.7}$$

where $X^{(+)}$ *denotes the Moore-Penrose inverse of* X*.*

Remark 5.2. *The perturbation inequality (5.7) for QR factorization also holds when* R *and* $\mathcal{R}$ *are two lower triangular matrix with all diagonal elements being positive.*

5.2 Theoretical Basis

In this section, we develop a new regularized orthogonal linear discriminant analysis method (ROLDA), in which no parameter selection is needed. Before we embark on the development of new ROLDA, based on the above preliminaries, we first characterize all solutions of OLDA (5.1) and ROLDA (5.2) with regularization parameter λ. These characterizations will play critical role for establishing the relationship between optimization problems OLDA (5.1) and ROLDA (5.2) in the later development.

5.2.1 Characterization of All Solutions to OLDA

In this section, we characterize all solutions of optimization problem OLDA (5.1) by the following theorem.

Theorem 5.7. *Let $A_2 \in \mathbf{R}^{m\times(k-1)}$ and $A_3 \in \mathbf{R}^{m\times(n-k)}$ be determined by (3.6). Let orthogonal matrix $Q \in \mathrm{R}^{m\times m}$ be such that*

$$\begin{bmatrix} A_2 & A_3 \end{bmatrix} = Q \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \\ 0 & 0 \end{bmatrix}, \tag{5.8}$$

where $R_{1,1} \in \mathbf{R}^{q\times(k-1)}$ and $R_{2,2} \in \mathbf{R}^{(\gamma-q)\times(n-k)}$ are of full row rank. Next, let the economic QR factorization of $R_{2,2}^T$ be

$$R_{2,2}^T = \mathcal{V}_1 \mathcal{R}_{2,2}^T, \tag{5.9}$$

where $\mathcal{V}_1 \in \mathbf{R}^{(n-k)\times(\gamma-q)}$ is column orthogonal, $\mathcal{R}_{2,2} \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ is lower triangular with all diagonal elements being positive. Finally, let

$$\mathcal{R}_{1,2} = R_{1,2}\mathcal{V}_1, \tag{5.10}$$

and the QR factorization of $\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix}$ be

$$\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix} = V \begin{bmatrix} \Pi \\ 0 \end{bmatrix}, \tag{5.11}$$

where $\Pi \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ *is upper triangular with all diagonal elements being positive,* $V \in \mathbf{R}^{\gamma\times\gamma}$ *is orthogonal,*

$$V = \begin{bmatrix} V_{1,1} & V_{1,2} \\ V_{2,1} & V_{2,2} \end{bmatrix},$$

$V_{1,2} \in \mathbf{R}^{q\times q}$ *is lower triangular with all diagonal elements being non-negative and* $V_{2,1} \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$. *Then* $V_{1,2}$ *and* $V_{2,1}$ *are nonsingular,*

$$q = \operatorname{rank}(S_b), \quad \gamma = \operatorname{rank}(S_t),$$

and all solutions $G \in \mathbf{R}^{m\times l}$ *of OLDA (5.1) are parameterized by*

$$G = Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & -V_{2,2}^T W_{2,2} - W_{3,1}^T W_{3,2} \\ 0 & W_{2,2} \\ W_{3,1} & W_{3,2} \end{bmatrix} \begin{bmatrix} W_{1,1} & \\ & I \end{bmatrix} W, \tag{5.12}$$

where $W \in \mathbf{R}^{l\times l}$ *is orthogonal,* $W_{1,1} \in \mathbf{R}^{q\times q}$ *is nonsingular,* $W_{3,1} \in \mathbf{R}^{(m-\gamma)\times q}$ *and* $W_{2,2} \in \mathbf{R}^{(\gamma-q)\times(l-q)}$ *are arbitrary,* $\begin{bmatrix} I \\ W_{3,1} \end{bmatrix} W_{1,1}$ *and* $\begin{bmatrix} V_{2,1}^T W_{2,2} \\ W_{3,2} \\ W_{3,1}^T W_{3,2} \end{bmatrix}$ *are column orthogonal.*

Proof. By (5.8), the following hold

$$A_2 = Q \begin{bmatrix} R_{1,1} \\ 0 \\ 0 \end{bmatrix}, \quad A_3 = Q \begin{bmatrix} R_{1,2} \\ R_{2,2} \\ 0 \end{bmatrix},$$

so, follows from the new expression of scatter matrices in (3.7), i.e.,

$$S_b = A_2 A_2^T, \quad S_t = \begin{bmatrix} A_2 & A_3 \end{bmatrix} \begin{bmatrix} A_2 & A_3 \end{bmatrix}^T,$$

$\gamma = \operatorname{rank} \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \end{bmatrix} = \operatorname{rank} \begin{bmatrix} A_2 & A_3 \end{bmatrix} = \operatorname{rank}(S_t)$ and $q = \operatorname{rank}(R_{1,1}) = \operatorname{rank}(A_2) = \operatorname{rank}(S_b)$. According to (5.11), the QR factorization of $\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix}$,

$$\mathcal{R}_{2,2} = V_{2,1}\Pi,$$

with $\mathcal{R}_{2,2}$ and Π are nonsingular, we have $V_{2,1}$ is nonsingular, which together with that

$$V = \begin{bmatrix} V_{1,1} & V_{1,2} \\ V_{2,1} & V_{2,2} \end{bmatrix}$$

is orthogonal and Lemma 5.5 yields that $V_{1,2}$ is also nonsingular .

Let $\mathcal{V}_2 \in \mathbf{R}^{(n-k)\times(n+q-k-\gamma)}$ be such that $\begin{bmatrix} \mathcal{V}_1 & \mathcal{V}_2 \end{bmatrix}$ is orthogonal, and denote

$$\mathcal{R}_{1,3} := R_{1,2}\mathcal{V}_2,$$

then together with (5.9) and (5.10), we have

$$R_{2,2} = \begin{bmatrix} \mathcal{R}_{2,2} & 0 \end{bmatrix} \begin{bmatrix} \mathcal{V}_1 & \mathcal{V}_2 \end{bmatrix}^T, \quad R_{1,2} = \begin{bmatrix} \mathcal{R}_{1,2} & \mathcal{R}_{1,3} \end{bmatrix} \begin{bmatrix} \mathcal{V}_1 & \mathcal{V}_2 \end{bmatrix}^T. \tag{5.13}$$

After some straightforward manipulation, it can be seen that

$$\left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \right)^T A_2 = \begin{bmatrix} V_{1,2}^T R_{1,1} \\ 0 \\ 0 \end{bmatrix},$$

and

$$\left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \right)^T A_3 \begin{bmatrix} \mathcal{V}_1 & \mathcal{V}_2 \end{bmatrix} = \begin{bmatrix} 0 & V_{1,2}^T \mathcal{R}_{1,3} \\ \mathcal{R}_{2,2} & 0 \\ 0 & 0 \end{bmatrix},$$

where the second equation is deduced from (5.13) and the QR factorization (5.11), i.e.,

$$V_{1,2}^T \mathcal{R}_{1,2} + V_{2,2}^T \mathcal{R}_{2,2} = 0.$$

Hence

$$\left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \right)^T S_b \left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \right) = \begin{bmatrix} V_{1,2}^T R_{11} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} V_{1,2}^T R_{11} \\ 0 \\ 0 \end{bmatrix}^T,$$

and

$$\left(Q\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)^T S_t \left(Q\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)$$
$$= \begin{bmatrix}V_{1,2}^T R_{1,1} & 0 & V_{1,2}^T \mathcal{R}_{1,3}\\ 0 & \mathcal{R}_{2,2} & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}V_{1,2}^T R_{1,1} & 0 & V_{1,2}^T \mathcal{R}_{1,3}\\ 0 & \mathcal{R}_{2,2} & 0\\ 0 & 0 & 0\end{bmatrix}^T.$$

Furthermore, by the nonsingularity of $V_{1,2}$ and Q, it holds that:

$$\begin{aligned}
&\mathrm{trace}(S_t^{(+)} S_b)\\
=&\mathrm{trace}\left\{\left(\left(Q\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)^T S_t \left(Q\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)\right)^{(+)}\right.\\
&\left.\times\left(Q\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)^T S_b \left(Q\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)\right\}\\
=&\mathrm{trace}\left\{\left(\begin{bmatrix}V_{1,2}^T R_{1,1} & 0 & V_{1,2}^T \mathcal{R}_{1,3}\\ 0 & \mathcal{R}_{2,2} & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}V_{1,2}^T R_{1,1} & 0 & V_{1,2}^T \mathcal{R}_{1,3}\\ 0 & \mathcal{R}_{2,2} & 0\\ 0 & 0 & 0\end{bmatrix}^T\right)^{-1}\right.\\
&\left.\times\begin{bmatrix}V_{1,2}^T R_{1,1} R_{1,1}^T V_{1,2} & 0\\ 0 & 0\end{bmatrix}\right\}\\
=&\mathrm{trace}\left(\begin{bmatrix}V_{1,2}^T(R_{1,1}R_{1,1}^T + \mathcal{R}_{1,3}\mathcal{R}_{1,3}^T)V_{1,2} & \\ & \mathcal{R}_{2,2}\mathcal{R}_{2,2}^T\end{bmatrix}^{-1}\begin{bmatrix}V_{1,2}^T R_{1,1} R_{1,1}^T V_{1,2} & 0\\ 0 & 0\end{bmatrix}\right)\\
=&\mathrm{trace}((R_{1,1}R_{1,1}^T + \mathcal{R}_{1,3}\mathcal{R}_{1,3}^T)^{-1}(R_{1,1}R_{1,1}^T)). \qquad (5.14)
\end{aligned}$$

For any $G \in \mathbf{R}^{m\times l}$, let

$$\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}^{-1} Q^T G = \mathcal{W} = \begin{bmatrix}\mathcal{W}_1\\ \mathcal{W}_2\\ \mathcal{W}_3\end{bmatrix} \in \mathbf{R}^{m\times l},$$

with $\mathcal{W}_1 \in \mathbf{R}^{q\times l},\ W_2 \in \mathbf{R}^{(\gamma-q)\times l},\ W_3 \in \mathbf{R}^{(m-\gamma)\times l}$. Then

$$\begin{aligned}
G^T S_t G &= \left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix} \right)^T S_t \left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix} \right) \\
&= \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix}^T \begin{bmatrix} V_{1,2}^T R_{1,1} & 0 & V_{1,2}^T \mathcal{R}_{1,3} \\ 0 & \mathcal{R}_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} V_{1,2}^T R_{1,1} & 0 & V_{1,2}^T \mathcal{R}_{1,3} \\ 0 & \mathcal{R}_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix}^T \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix} \\
&= \begin{bmatrix} \mathcal{W}_1^T V_{1,2}^T R_{1,1} & \mathcal{W}_2^T \mathcal{R}_{2,2} & \mathcal{W}_1^T V_{1,2}^T \mathcal{R}_{1,3} \end{bmatrix} \begin{bmatrix} R_{1,1}^T V_{1,2} \mathcal{W}_1 \\ \mathcal{R}_{2,2}^T \mathcal{W}_2 \\ \mathcal{R}_{1,3}^T V_{1,2} \mathcal{W}_1 \end{bmatrix} \\
&= \mathcal{W}_1^T V_{1,2}^T (R_{1,1} R_{1,1}^T + \mathcal{R}_{1,3} \mathcal{R}_{1,3}^T) V_{1,2} \mathcal{W}_1 + \mathcal{W}_2^T \mathcal{R}_{2,2} \mathcal{R}_{2,2}^T \mathcal{W}_2,
\end{aligned}$$

and

$$\begin{aligned}
G^T S_b G &= \left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix} \right)^T S_b \left(Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix} \right) \\
&= \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix}^T \begin{bmatrix} V_{1,2}^T R_{1,1} \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} V_{1,2}^T R_{1,1} \\ 0 \\ 0 \end{bmatrix}^T \begin{bmatrix} \mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3 \end{bmatrix} \\
&= \mathcal{W}_1^T V_{1,2}^T R_{1,1} R_{1,1}^T V_{1,2} \mathcal{W}_1.
\end{aligned}$$

Thus, by Lemma 3.1 and Lemma 5.1,

$$\begin{aligned}
&\text{trace}((G^T S_t G)^{(+)} (G^T S_b G)) \\
=&\text{trace}((\mathcal{W}_1^T V_{1,2}^T (R_{1,1} R_{1,1}^T + \mathcal{R}_{1,3} \mathcal{R}_{1,3}^T) V_{1,2} \mathcal{W}_1 + \mathcal{W}_2^T \mathcal{R}_{2,2} \mathcal{R}_{2,2}^T \mathcal{W}_2)^{(+)} \\
&\times (\mathcal{W}_1^T V_{1,2}^T R_{1,1} R_{1,1}^T V_{1,2} \mathcal{W}_1)) \\
\leq&\text{trace}((\mathcal{W}_1^T V_{1,2}^T (R_{1,1} R_{1,1}^T + \mathcal{R}_{1,3} \mathcal{R}_{1,3}^T) V_{1,2} \mathcal{W}_1)^{(+)} (\mathcal{W}_1^T V_{1,2}^T R_{1,1} R_{1,1}^T V_{1,2} \mathcal{W}_1)) \\
\leq&\text{trace}((R_{1,1} R_{1,1}^T + \mathcal{R}_{1,3} \mathcal{R}_{1,3}^T)^{-1} (R_{1,1} R_{1,1}^T)).
\end{aligned} \tag{5.15}$$

By using lemma 5.3 and equality (5.14), G is a solution of OLDA (5.1) if and only if the two inequalities in (5.15) hold at equality. Lemma 3.1 implies that the second

equality in (5.15) holds if and only if

$$\begin{aligned} & \operatorname{rank}\left[\mathcal{W}_1^T V_{1,2}^T (R_{1,1}R_{1,1}^T + \mathcal{R}_{1,3}\mathcal{R}_{1,3}^T)V_{1,2}\mathcal{W}_1 \quad \mathcal{W}_2^T \mathcal{R}_{2,2}\mathcal{R}_{2,2}^T \mathcal{W}_2\right] \\ = \; & \operatorname{rank}(\mathcal{W}_1^T V_{1,2}^T (R_{1,1}R_{1,1}^T + \mathcal{R}_{1,3}\mathcal{R}_{1,3}^T)V_{1,2}\mathcal{W}_1) + \operatorname{rank}(\mathcal{W}_2^T \mathcal{R}_{2,2}\mathcal{R}_{2,2}^T \mathcal{W}_2), \end{aligned}$$

which is further equivalent to

$$\operatorname{rank}\begin{bmatrix}\mathcal{W}_1 \\ \mathcal{W}_2\end{bmatrix} = \operatorname{rank}(\mathcal{W}_1) + \operatorname{rank}(\mathcal{W}_2), \tag{5.16}$$

by applying Lemma 5.2, whilst Lemma 5.1 implies that the third equality in (5.15) holds if and only if

$$\operatorname{rank}(\mathcal{W}_1) = q. \tag{5.17}$$

Lemma 5.4 together with (5.16) and (5.17) leads to the expression of $\begin{bmatrix}\mathcal{W}_1 \\ \mathcal{W}_2\end{bmatrix}$ as

$$\begin{bmatrix}\mathcal{W}_1 \\ \mathcal{W}_2\end{bmatrix} = \begin{bmatrix}W_{1,1} & W_{1,2} \\ 0 & W_{2,2}\end{bmatrix} W = \begin{bmatrix}I & W_{1,2} \\ 0 & W_{2,2}\end{bmatrix} \begin{bmatrix}W_{1,1} & \\ & I\end{bmatrix} W, \tag{5.18}$$

where $W \in \mathbf{R}^{l\times l}$ is orthogonal, $W_{1,1} \in \mathbf{R}^{q\times q}$ is of full rank, $W_{1,2} \in \mathbf{R}^{q\times(l-q)}$ and $W_{2,2} \in \mathbf{R}^{(\gamma-q)\times(l-q)}$ are arbitrary. Let

$$\begin{bmatrix}W_{3,1} & W_{3,2}\end{bmatrix} := \mathcal{W}_3 W^T \begin{bmatrix}W_{1,1} & \\ & I\end{bmatrix}^{-1},$$

with $W_{3,1} \in \mathbf{R}^{(m-\gamma)\times q}$, $W_{3,2} \in \mathbf{R}^{(m-\gamma)\times(l-q)}$. We have that

$$G = Q \begin{bmatrix}V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I\end{bmatrix} \begin{bmatrix}\mathcal{W}_1 \\ \mathcal{W}_2 \\ \mathcal{W}_3\end{bmatrix} = Q \begin{bmatrix}V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I\end{bmatrix} \begin{bmatrix}I & W_{1,2} \\ 0 & W_{2,2} \\ W_{3,1} & W_{3,2}\end{bmatrix} \begin{bmatrix}W_{1,1} & \\ & I\end{bmatrix} W \tag{5.19}$$

is column orthogonal if and only if

$$
\begin{aligned}
I &= \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I \\ 0 \\ W_{3,1} \end{bmatrix} W_{1,1} \right)^T \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I \\ 0 \\ W_{3,1} \end{bmatrix} W_{1,1} \right) \\
&= \left(\begin{bmatrix} I \\ 0 \\ W_{3,1} \end{bmatrix} W_{1,1} \right)^T \begin{bmatrix} I & V_{2,2}^T & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \left(\begin{bmatrix} I \\ 0 \\ W_{3,1} \end{bmatrix} W_{1,1} \right) \\
&= \left(\begin{bmatrix} I \\ W_{3,1} \end{bmatrix} W_{1,1} \right)^T \left(\begin{bmatrix} I \\ W_{3,1} \end{bmatrix} W_{1,1} \right), \qquad (5.20)
\end{aligned}
$$

$$
\begin{aligned}
0 &= \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I \\ 0 \\ W_{3,1} \end{bmatrix} W_{1,1} \right)^T \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,2} \\ W_{2,2} \\ W_{3,2} \end{bmatrix} \right) \\
&= \begin{bmatrix} V_{1,2}W_{1,1} \\ V_{2,2}W_{1,1} \\ W_{3,1}W_{1,1} \end{bmatrix}^T \begin{bmatrix} V_{1,2}W_{1,2} \\ V_{2,2}W_{1,2} + W_{2,2} \\ W_{3,2} \end{bmatrix} \\
&= W_{1,1}^T(W_{1,2} + V_{2,2}^T W_{2,2} + W_{3,1}^T W_{3,2}),
\end{aligned}
$$

i.e.,

$$
W_{1,2} + V_{2,2}^T W_{2,2} + W_{3,1}^T W_{3,2} = 0, \qquad (5.21)
$$

and

$$
\begin{aligned}
I &= \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,2} \\ W_{2,2} \\ W_{3,2} \end{bmatrix} \right)^T \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,2} \\ W_{2,2} \\ W_{3,2} \end{bmatrix} \right) \\
&= \begin{bmatrix} V_{1,2}W_{1,2} \\ V_{2,2}W_{1,2} + W_{2,2} \\ W_{3,2} \end{bmatrix}^T \begin{bmatrix} V_{1,2}W_{1,2} \\ V_{2,2}W_{1,2} + W_{2,2} \\ W_{3,2} \end{bmatrix} \\
&= W_{1,2}^T(W_{1,2} + V_{2,2}^T W_{2,2}) + W_{2,2}^T(V_{2,2}W_{1,2} + W_{2,2}) + W_{3,2}^T W_{3,2} \\
&= - W_{1,2}^T W_{3,1}^T W_{3,2} + W_{2,2}^T(-V_{2,2}V_{2,2}^T W_{2,2} - V_{2,2}W_{3,1}^T W_{3,2} + W_{2,2}) + W_{3,2}^T W_{3,2} \\
&= - (W_{1,2} + V_{2,2}^T W_{2,2})^T W_{3,1}^T W_{3,2} + W_{2,2}^T V_{2,1} V_{2,1}^T W_{2,2} + W_{3,2}^T W_{3,2}
\end{aligned}
$$

$$
\begin{aligned}
=&W_{3,2}^T W_{3,1} W_{3,1}^T W_{3,2} + W_{2,2}^T V_{2,1} V_{2,1}^T W_{2,2} + W_{3,2}^T W_{3,2} \\
=&\begin{bmatrix} V_{2,1}^T W_{2,2} \\ W_{3,2} \\ W_{3,1}^T W_{3,2} \end{bmatrix}^T \begin{bmatrix} V_{2,1}^T W_{2,2} \\ W_{3,2} \\ W_{3,1}^T W_{3,2} \end{bmatrix},
\end{aligned} \tag{5.22}
$$

where the fourth and sixth equalities in (5.22) use the results in (5.21).

Therefore, (5.12) directly follows from (5.19), (5.20), (5.21) and (5.22). □

Remark 5.3. *Orthogonal matrix V in Theorem 5.7 can be computed as follows:*

- *Compute QR factorization of* $\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix}$*:*

$$
\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix} = \hat{V} \begin{bmatrix} \Pi \\ 0 \end{bmatrix},
$$

  where $\Pi \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ *is upper triangular with all diagonal elements being positive and* $\hat{V} \in \mathbf{R}^{\gamma\times\gamma}$ *is orthogonal. Denote*

$$
\hat{V} = \begin{bmatrix} V_{1,1} & \hat{V}_{1,2} \\ V_{2,1} & \hat{V}_{2,2} \end{bmatrix},
$$

  where $V_{1,1} \in \mathbf{R}^{q\times(\gamma-q)}$, $V_{2,1} \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$, $\hat{V}_{1,2} \in \mathbf{R}^{q\times q}$, $\hat{V}_{2,2} \in \mathbf{R}^{(\gamma-q)\times q}$.

- *Compute the QR factorization of* $\hat{V}_{1,2}^T$*:*

$$
\hat{V}_{1,2}^T = \Gamma\Lambda,
$$

  where $\Lambda \in \mathbf{R}^{q\times q}$ *is upper triangular with all diagonal elements being non-negative and* $\Gamma \in \mathbf{R}^{q\times q}$ *is orthogonal. Then*

$$
V = \begin{bmatrix} V_{1,1} & \Lambda^T \\ V_{2,1} & \hat{V}_{2,2}\Gamma \end{bmatrix}
$$

  is the orthogonal matrix satisfying conditions in Theorem 5.7.

5.2.2 Characterization of All Solutions to ROLDA

In this section we characterize all solutions of optimization problem ROLDA (5.2), the results are shown in Theorem 5.8.

Theorem 5.8. *Let A_2, A_3 be determined by (3.6), Q, $R_{1,1}, R_{1,2}$ and $R_{2,2}$ be determined by (5.8) and $\lambda > 0$. Let the economic QR factorization of $\begin{bmatrix} R_{2,2} & \sqrt{\lambda}I \end{bmatrix}^T$ be*

$$\begin{bmatrix} R_{2,2}^T \\ \sqrt{\lambda}I \end{bmatrix} = \mathcal{V}_1^\lambda (\mathcal{R}_{2,2}^\lambda)^T, \tag{5.23}$$

where $\mathcal{V}_1^\lambda \in \mathbf{R}^{(m+\gamma-k-q)\times(\gamma-q)}$ is column orthogonal, $\mathcal{R}_{2,2}^\lambda \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ is lower triangular with all diagonal elements being positive. Let

$$\mathcal{R}_{1,2}^\lambda = \begin{bmatrix} R_{1,2} & 0 \end{bmatrix} \mathcal{V}_1^\lambda. \tag{5.24}$$

and the QR factorization of $\begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix}$ be

$$\begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix} = V^\lambda \begin{bmatrix} \Pi^\lambda \\ 0 \end{bmatrix}, \tag{5.25}$$

where $\Pi^\lambda \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ is upper triangular with all diagonal elements being positive, $V^\lambda \in \mathbf{R}^{\gamma\times\gamma}$ is orthogonal,

$$V^\lambda = \begin{bmatrix} V_{1,1}^\lambda & V_{1,2}^\lambda \\ V_{2,1}^\lambda & V_{2,2}^\lambda \end{bmatrix},$$

$V_{1,2}^\lambda \in \mathbf{R}^{q\times q}$ is lower triangular with all diagonal elements being non-negative and $V_{2,1}^\lambda \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$. Then $V_{2,1}^\lambda$ and $V_{1,2}^\lambda$ are nonsingular, and all solutions $G^\lambda \in \mathbf{R}^{m\times l}$ of ROLDA (5.2) are parameterized by

$$G^\lambda = Q \begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & -(V_{2,2}^\lambda)^T W_{2,2}^\lambda \\ 0 & W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} W^\lambda, \tag{5.26}$$

where $W^\lambda \in \mathbf{R}^{l\times l}$ is orthogonal, $W_{2,2}^\lambda \in \mathbf{R}^{(\gamma-q)\times(l-q)}$, $W_{3,2}^\lambda \in \mathbf{R}^{(m-\gamma)\times(l-q)}$, and $\begin{bmatrix} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix}$ is column orthogonal.

Proof. First, by directly calculating (5.25),

$$\mathcal{R}_{2,2}^\lambda = V_{2,1}^\lambda \Pi^\lambda,$$

$\mathcal{R}_{2,2}^{\lambda}$ and Π^{λ} are nonsingular, so, $V_{2,1}^{\lambda}$ is nonsingular. Which, in return, together with that

$$V^{\lambda} = \begin{bmatrix} V_{1,1}^{\lambda} & V_{1,2}^{\lambda} \\ V_{2,1}^{\lambda} & V_{2,2}^{\lambda} \end{bmatrix}$$

is orthogonal and Lemma 5.5 gives that $V_{1,2}^{\lambda}$ is also nonsingular.

Next, let $\mathcal{V}_2^{\lambda}$ be such that $\begin{bmatrix} \mathcal{V}_1^{\lambda} & \mathcal{V}_2^{\lambda} \end{bmatrix}$ is orthogonal and denote

$$\mathcal{R}_{1,3}^{\lambda} := \begin{bmatrix} R_{1,2} & 0 \end{bmatrix} \mathcal{V}_2^{\lambda},$$

then together with (5.23) and (5.24), we have

$$\begin{bmatrix} R_{2,2} & \sqrt{\lambda} I \end{bmatrix} = \begin{bmatrix} \mathcal{R}_{2,2}^{\lambda} & 0 \end{bmatrix} \begin{bmatrix} \mathcal{V}_1^{\lambda} & \mathcal{V}_2^{\lambda} \end{bmatrix}^T, \quad \begin{bmatrix} R_{1,2} & 0 \end{bmatrix} = \begin{bmatrix} \mathcal{R}_{1,2}^{\lambda} & \mathcal{R}_{1,3}^{\lambda} \end{bmatrix} \begin{bmatrix} \mathcal{V}_1^{\lambda} & \mathcal{V}_2^{\lambda} \end{bmatrix}^T. \tag{5.27}$$

By the definition of Q in (5.8), new expression of S_t in (3.7) and (5.27),

$$\begin{aligned}
& Q^T (S_t + \lambda I) Q \\
=& Q^T \begin{bmatrix} A_2 & A_3 & \sqrt{\lambda} I \end{bmatrix} \begin{bmatrix} A_2 & A_3 & \sqrt{\lambda} I \end{bmatrix}^T Q \\
=& \begin{bmatrix} R_{1,1} & R_{1,2} & \sqrt{\lambda} I & 0 & 0 \\ 0 & R_{2,2} & 0 & \sqrt{\lambda} I & 0 \\ 0 & 0 & 0 & 0 & \sqrt{\lambda} I \end{bmatrix} \begin{bmatrix} R_{1,1} & R_{1,2} & \sqrt{\lambda} I & 0 & 0 \\ 0 & R_{2,2} & 0 & \sqrt{\lambda} I & 0 \\ 0 & 0 & 0 & 0 & \sqrt{\lambda} I \end{bmatrix}^T \\
=& \begin{bmatrix} R_{1,1} & \mathcal{R}_{1,2}^{\lambda} & \sqrt{\lambda} I & \mathcal{R}_{1,3}^{\lambda} & 0 \\ 0 & \mathcal{R}_{2,2}^{\lambda} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{\lambda} I \end{bmatrix} \begin{bmatrix} R_{1,1} & \mathcal{R}_{1,2}^{\lambda} & \sqrt{\lambda} I & \mathcal{R}_{1,3}^{\lambda} & 0 \\ 0 & \mathcal{R}_{2,2}^{\lambda} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{\lambda} I \end{bmatrix}^T.
\end{aligned}$$

Thus, similar to the proof of Theorem 5.7, we have

$$\begin{aligned}
& \left(Q \begin{bmatrix} V_{1,2}^{\lambda} & 0 & 0 \\ V_{2,2}^{\lambda} & I & 0 \\ 0 & 0 & I \end{bmatrix} \right)^T (S_t + \lambda I) \left(Q \begin{bmatrix} V_{1,2}^{\lambda} & 0 & 0 \\ V_{2,2}^{\lambda} & I & 0 \\ 0 & 0 & I \end{bmatrix} \right) \\
=& \begin{bmatrix} (V_{1,2}^{\lambda})^T (R_{1,1} R_{1,1}^T + \mathcal{R}_{1,3}^{\lambda} (\mathcal{R}_{1,3}^{\lambda})^T + \lambda I) V_{1,2}^{\lambda} & 0 & 0 \\ 0 & \mathcal{R}_{2,2}^{\lambda} (\mathcal{R}_{2,2}^{\lambda})^T & 0 \\ 0 & 0 & \lambda I \end{bmatrix},
\end{aligned}$$

where the second equation is deduced from the QR factorization (5.25), i.e.,

$$(V_{1,2}^{\lambda})^T \mathcal{R}_{1,2}^{\lambda} + (V_{2,2}^{\lambda})^T \mathcal{R}_{2,2}^{\lambda} = 0,$$

and

$$\left(Q\begin{bmatrix}V_{1,2}^{\lambda} & 0 & 0\\ V_{2,2}^{\lambda} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)^{T} S_b \left(Q\begin{bmatrix}V_{1,2}^{\lambda} & 0 & 0\\ V_{2,2}^{\lambda} & I & 0\\ 0 & 0 & I\end{bmatrix}\right) = \begin{bmatrix}(V_{1,2}^{\lambda})^T R_{11} R_{1,1}^T V_{1,2}^{\lambda} & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix},$$

which together with the nonsingularity of $V_{1,2}^{\lambda}$ and Q gives

$$\begin{aligned}
&\text{trace}((S_t+\lambda I)^{-1}S_b)\\
=&\text{trace}\left\{\left(\left(Q\begin{bmatrix}V_{1,2}^{\lambda} & 0 & 0\\ V_{2,2}^{\lambda} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)^{T}(S_t+\lambda I)\left(Q\begin{bmatrix}V_{1,2}^{\lambda} & 0 & 0\\ V_{2,2}^{\lambda} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)\right)^{-1}\right.\\
&\left.\times\left(Q\begin{bmatrix}V_{1,2}^{\lambda} & 0 & 0\\ V_{2,2}^{\lambda} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)^{T} S_b\left(Q\begin{bmatrix}V_{1,2}^{\lambda} & 0 & 0\\ V_{2,2}^{\lambda} & I & 0\\ 0 & 0 & I\end{bmatrix}\right)\right\}\\
=&\text{trace}\left(\begin{bmatrix}(V_{1,2}^{\lambda})^T(R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^{\lambda}(\mathcal{R}_{1,3}^{\lambda})^T+\lambda I)V_{1,2}^{\lambda} & 0 & 0\\ 0 & \mathcal{R}_{2,2}^{\lambda}(\mathcal{R}_{2,2}^{\lambda})^T & 0\\ 0 & 0 & \lambda I\end{bmatrix}^{-1}\right.\\
&\left.\times\begin{bmatrix}(V_{1,2}^{\lambda})^T R_{1,1}R_{1,1}^T V_{1,2}^{\lambda} & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\right)\\
=&\text{trace}((R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^{\lambda}(\mathcal{R}_{1,3}^{\lambda})^T+\lambda I)^{-1}(R_{1,1}R_{1,1}^T)).
\end{aligned}\tag{5.28}$$

For any $G^{\lambda}\in\mathbf{R}^{m\times l}$, let

$$\begin{bmatrix}V_{1,2}^{\lambda} & 0 & 0\\ V_{2,2}^{\lambda} & I & 0\\ 0 & 0 & I\end{bmatrix}^{-1} Q^T G^{\lambda} = \mathcal{W}^{\lambda} = \begin{bmatrix}\mathcal{W}_1^{\lambda}\\ \mathcal{W}_2^{\lambda}\\ \mathcal{W}_3^{\lambda}\end{bmatrix}\in\mathbf{R}^{m\times l},$$

with $\mathcal{W}_1^{\lambda}\in\mathbf{R}^{q\times l}$, $W_2^{\lambda}\in\mathbf{R}^{(\gamma-q)\times l}$, $W_3^{\lambda}\in\mathbf{R}^{(m-\gamma)\times l}$. Then

$$\begin{aligned}
(G^{\lambda})^T S_b G^{\lambda} &= \begin{bmatrix}\mathcal{W}_1^{\lambda}\\ \mathcal{W}_2^{\lambda}\\ \mathcal{W}_3^{\lambda}\end{bmatrix}^T\begin{bmatrix}(V_{1,2}^{\lambda})^T R_{1,1}R_{1,1}^T V_{1,2}^{\lambda} & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\mathcal{W}_1^{\lambda}\\ \mathcal{W}_2^{\lambda}\\ \mathcal{W}_3^{\lambda}\end{bmatrix}\\
&= (\mathcal{W}_1^{\lambda})^T(V_{1,2}^{\lambda})^T R_{1,1}R_{1,1}^T V_{1,2}^{\lambda}\mathcal{W}_1^{\lambda},
\end{aligned}$$

and

$$
\begin{aligned}
&(G^\lambda)^T(S_t+\lambda I)G^\lambda \\
=&\begin{bmatrix}\mathcal{W}_1^\lambda\\ \mathcal{W}_2^\lambda\\ \mathcal{W}_3^\lambda\end{bmatrix}^T
\begin{bmatrix}(V_{1,2}^\lambda)^T(R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^\lambda(\mathcal{R}_{1,3}^\lambda)^T+\lambda I)V_{1,2}^\lambda & 0 & 0\\ 0 & \mathcal{R}_{2,2}^\lambda(\mathcal{R}_{2,2}^\lambda)^T & 0\\ 0 & 0 & \lambda I\end{bmatrix}
\begin{bmatrix}\mathcal{W}_1^\lambda\\ \mathcal{W}_2^\lambda\\ \mathcal{W}_3^\lambda\end{bmatrix}\\
=&(\mathcal{W}_1^\lambda)^T(V_{1,2}^\lambda)^T(R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^\lambda(\mathcal{R}_{1,3}^\lambda)^T+\lambda I)V_{1,2}^\lambda\mathcal{W}_1^\lambda+(\mathcal{W}_2^\lambda)^T\mathcal{R}_{2,2}^\lambda(\mathcal{R}_{2,2}^\lambda)^T\mathcal{W}_2^\lambda\\
&+\lambda(\mathcal{W}_3^\lambda)^T\mathcal{W}_3^\lambda.
\end{aligned}
$$

Thus, Lemma 3.1 and Lemma 5.1 lead to

$$
\begin{aligned}
&\operatorname{trace}\{((G^\lambda)^T(S_t+\lambda I)G^\lambda)^{-1}(G^\lambda)^TS_bG^\lambda\}\\
=&\operatorname{trace}\{((\mathcal{W}_1^\lambda)^T(V_{1,2}^\lambda)^T(R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^\lambda(\mathcal{R}_{1,3}^\lambda)^T+\lambda I)V_{1,2}^\lambda\mathcal{W}_1^\lambda+\lambda(\mathcal{W}_3^\lambda)^T\mathcal{W}_3^\lambda\\
&\quad+(\mathcal{W}_2^\lambda)^T\mathcal{R}_{2,2}^\lambda(\mathcal{R}_{2,2}^\lambda)^T\mathcal{W}_2^\lambda)^{-1}(\mathcal{W}_1^\lambda)^T(V_{1,2}^\lambda)^TR_{1,1}R_{1,1}^TV_{1,2}^\lambda\mathcal{W}_1^\lambda\}\\
\le&\operatorname{trace}\{((\mathcal{W}_1^\lambda)^T(V_{1,2}^\lambda)^T(R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^\lambda(\mathcal{R}_{1,3}^\lambda)^T+\lambda I)V_{1,2}^\lambda\mathcal{W}_1^\lambda)^{-1}(\mathcal{W}_1^\lambda)^T(V_{1,2}^\lambda)^T\\
&\quad\times R_{1,1}R_{1,1}^TV_{1,2}^\lambda\mathcal{W}_1^\lambda\}\\
\le&\operatorname{trace}((R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^\lambda(\mathcal{R}_{1,3}^\lambda)^T+\lambda I)^{-1}R_{1,1}R_{1,1}^T).
\end{aligned}
\tag{5.29}
$$

Hence, by using lemma 5.3 and equality (5.28), G^λ is a solution of ROLDA (5.2) if and only if both of the last two equalities in (5.29) hold. According to Lemma 3.1 and the nonsingularity of $V_{1,2}^\lambda$, the second equality in (5.29) holds if and only if

$$
\begin{aligned}
&\operatorname{rank}\left[(\mathcal{W}_1^\lambda)^T(V_{1,2}^\lambda)^T(R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^\lambda(\mathcal{R}_{1,3}^\lambda)^T+\lambda I)V_{1,2}^\lambda\mathcal{W}_1^\lambda\quad\lambda(\mathcal{W}_3^\lambda)^T\mathcal{W}_3^\lambda\right.\\
&\qquad\left.+(\mathcal{W}_2^\lambda)^T\mathcal{R}_{2,2}^\lambda(\mathcal{R}_{2,2}^\lambda)^T\mathcal{W}_2^\lambda\right]\\
=&\operatorname{rank}((\mathcal{W}_1^\lambda)^T(V_{1,2}^\lambda)^T(R_{1,1}R_{1,1}^T+\mathcal{R}_{1,3}^\lambda(\mathcal{R}_{1,3}^\lambda)^T+\lambda I)V_{1,2}^\lambda\mathcal{W}_1^\lambda)+\operatorname{rank}(\lambda(\mathcal{W}_3^\lambda)^T\mathcal{W}_3^\lambda\\
&\qquad+(\mathcal{W}_2^\lambda)^T\mathcal{R}_{2,2}^\lambda(\mathcal{R}_{2,2}^\lambda)^T\mathcal{W}_2^\lambda),
\end{aligned}
$$

which, by Lemma 5.2, is equivalent to

$$
\operatorname{rank}\begin{bmatrix}\mathcal{W}_1^\lambda\\ \mathcal{W}_2^\lambda\\ \mathcal{W}_3^\lambda\end{bmatrix}=\operatorname{rank}(\mathcal{W}_1^\lambda)+\operatorname{rank}\begin{bmatrix}\mathcal{W}_2^\lambda\\ \mathcal{W}_3^\lambda\end{bmatrix}.
\tag{5.30}
$$

And Lemma 5.1 yields that the third equality in (5.29) holds if and only if

$$\operatorname{rank}(\mathcal{W}_1^\lambda) = q. \tag{5.31}$$

Note that, by Lemma 5.4, conditions (5.30) and (5.31) are equivalent to that

$$\begin{bmatrix} \mathcal{W}_1^\lambda \\ \mathcal{W}_2^\lambda \\ \mathcal{W}_3^\lambda \end{bmatrix} = \begin{bmatrix} I & W_{1,2}^\lambda \\ 0 & W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} \begin{bmatrix} W_{1,1}^\lambda & \\ & I \end{bmatrix} \hat{W}^\lambda,$$

where $\hat{W}^\lambda \in \mathbf{R}^{l\times l}$ is orthogonal , $W_{1,1}^\lambda \in \mathbf{R}^{q\times q}$ is nonsingular, $W_{1,2}^\lambda \in \mathbf{R}^{q\times(l-q)}$, $W_{2,2}^\lambda \in \mathbf{R}^{(\gamma-r)\times(l-q)}$ and $W_{3,2}^\lambda \in \mathbf{R}^{(m-\gamma)\times(l-q)}$. Furthermore, we have

$$G^\lambda = Q \begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \mathcal{W}_1^\lambda \\ \mathcal{W}_2^\lambda \\ \mathcal{W}_3^\lambda \end{bmatrix} = Q \begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & W_{1,2}^\lambda \\ 0 & W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} \begin{bmatrix} W_{1,1}^\lambda & \\ & I \end{bmatrix} \hat{W}^\lambda \tag{5.32}$$

is column orthogonal if and only if

$$\begin{aligned} I &= \left(\begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,1}^\lambda \\ 0 \\ 0 \end{bmatrix} \right)^T \left(\begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,1}^\lambda \\ 0 \\ 0 \end{bmatrix} \right) \\ &= (W_{1,1}^\lambda)^T W_{1,1}^\lambda, \end{aligned} \tag{5.33}$$

$$\begin{aligned} 0 &= \left(\begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,1}^\lambda \\ 0 \\ 0 \end{bmatrix} \right)^T \left(\begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,2}^\lambda \\ W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \right) \\ &= (W_{1,1}^\lambda)^T (W_{1,2}^\lambda + (V_{2,2}^\lambda)^T W_{2,2}^\lambda), \end{aligned}$$

i.e.,

$$W_{1,2}^\lambda = -(V_{2,2}^\lambda)^T W_{2,2}^\lambda, \tag{5.34}$$

and

$$I = \left(\begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,2}^\lambda \\ W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \right)^T \left(\begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} W_{1,2}^\lambda \\ W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \right)$$

$$
\begin{aligned}
=&(W_{1,2}^{\lambda})^T(W_{1,2}^{\lambda}+(V_{2,2}^{\lambda})^T W_{2,2}^{\lambda})+(W_{2,2}^{\lambda})^T(V_{2,2}^{\lambda}W_{1,2}^{\lambda}+W_{2,2}^{\lambda})+(W_{3,2}^{\lambda})^T W_{3,2}^{\lambda}\\
=&(W_{2,2}^{\lambda})^T(V_{2,2}^{\lambda}W_{1,2}^{\lambda}+W_{2,2}^{\lambda})+(W_{3,2}^{\lambda})^T W_{3,2}^{\lambda}\\
=&(W_{2,2}^{\lambda})^T(-V_{2,2}^{\lambda}(V_{2,2}^{\lambda})^T W_{2,2}^{\lambda}+W_{2,2}^{\lambda})+(W_{3,2}^{\lambda})^T W_{3,2}^{\lambda}\\
=&(W_{2,2}^{\lambda})^T V_{2,1}^{\lambda}(V_{2,1}^{\lambda})^T W_{2,2}^{\lambda}+(W_{3,2}^{\lambda})^T W_{3,2}^{\lambda}. \qquad (5.35)
\end{aligned}
$$

Notice that, the third and the fourth equalities in (5.35) hold due to the fact in (5.34). Let

$$
W^{\lambda}=\begin{bmatrix} W_{1,1}^{\lambda} & \\ & I \end{bmatrix}\hat{W}^{\lambda}, \qquad (5.36)
$$

then W^{λ} is orthogonal, and consequently Theorem 5.8 follows from (5.32)-(5.36). □

Orthogonal matrix V^{λ} with special form in Theorem 5.8 can be computed similarly as V shown in Remark 5.3.

5.2.3 Relationship between OLDA and ROLDA

The major issue of the regularized LDA is how to choose an appropriate regularization parameter. In the existing regularized LDA methods, they all select the "best" regularization parameter from given parameter candidate set by using cross-validation for classification. To the best of our knowledge, there is no concrete method available in selecting an appropriate regularization parameter in practical applications. To concur this limitation in regularized LDA, we first reveal the intrinsic relationship between optimization problems OLDA (5.1) and ROLDA (5.2) in this section, such relationship will lead to a new mathematical criterion for choosing regularization parameter for ROLDA in the next section.

Before we study the relationship between the optimal solutions G of OLDA (5.1) and G^{λ} of ROLDA (5.2), the distance between two orthogonal matrices V in Theorem 5.7 and V^{λ} in Theorem 5.8 is obtained based on the perturbation theory of QR factorization, see Lemma 5.6 in Section 5.1.

Lemma 5.9. *With the notations in Theorems 5.7 and 5.8, the following holds:*

$$\left\|V^{\lambda}-V\right\|_F \leq (1+\sqrt{2})\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(\sqrt{n-q}+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F)}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(1+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2)}\lambda$$

provided $\lambda > 0$ *satisfies*

$$\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(1+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2) < 1.$$

Proof. According to (5.13) and (5.11), we have

$$R_{2,2}R_{2,2}^T = \mathcal{R}_{2,2}\mathcal{R}_{2,2}^T, \quad R_{2,2}R_{1,2}^T = \mathcal{R}_{2,2}\mathcal{R}_{1,2}^T,$$

$$V_{1,2}^T\mathcal{R}_{1,2} + V_{2,2}^T\mathcal{R}_{2,2} = 0, \quad \text{i.e.,} \quad -\mathcal{R}_{2,2}^{-T}\mathcal{R}_{1,2}^T = V_{2,2}V_{1,2}^{-1},$$

and

$$\begin{bmatrix} R_{1,2}R_{2,2}^T \\ R_{2,2}R_{2,2}^T \end{bmatrix} = \begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix}\mathcal{R}_{2,2}^T = \begin{bmatrix} V_{1,1} \\ V_{2,1} \end{bmatrix}(\Pi\mathcal{R}_{2,2}^T), \tag{5.37}$$

where $\Pi\mathcal{R}_{2,2}^T$ is upper triangular with all diagonal elements being positive and $\begin{bmatrix} V_{1,1} \\ V_{2,1} \end{bmatrix}$ is column orthogonal, which means (5.37) is an economic QR factorization of matrix $\begin{bmatrix} R_{1,2}R_{2,2}^T \\ R_{2,2}R_{2,2}^T \end{bmatrix}$. It is easy to see that

$$\begin{bmatrix} I \\ -(R_{2,2}R_{2,2}^T)^{-1}R_{2,2}R_{1,2}^T \end{bmatrix} = \begin{bmatrix} I \\ -\mathcal{R}_{2,2}^{-T}\mathcal{R}_{1,2}^T \end{bmatrix} = \begin{bmatrix} V_{1,2} \\ V_{2,2} \end{bmatrix}V_{1,2}^{-1}, \tag{5.38}$$

where $V_{1,2}^{-1}$ is a lower triangular with positive diagonal elements and $\begin{bmatrix} V_{1,2} \\ V_{2,2} \end{bmatrix}$ is column orthogonal, so (5.38) is an economic QR factorization of

$$\begin{bmatrix} I \\ -(R_{2,2}R_{2,2}^T)^{-1}R_{2,2}R_{1,2}^T \end{bmatrix}, \quad \text{i.e.,} \quad \begin{bmatrix} I \\ -\mathcal{R}_{2,2}^{-T}\mathcal{R}_{1,2}^T \end{bmatrix}.$$

Similarly, resulting from (5.27) and (5.25), the following hold,

$$R_{2,2}R_{2,2}^T + \lambda I = \mathcal{R}_{2,2}^{\lambda}(\mathcal{R}_{2,2}^{\lambda})^T, \quad R_{2,2}R_{1,2}^T = \mathcal{R}_{2,2}^{\lambda}(\mathcal{R}_{1,2}^{\lambda})^T,$$

$$(V_{1,2}^{\lambda})^T\mathcal{R}_{1,2}^{\lambda} + (V_{2,2}^{\lambda})^T\mathcal{R}_{2,2}^{\lambda} = 0, \quad \text{i.e.,} \quad -(\mathcal{R}_{2,2}^{\lambda})^{-T}(\mathcal{R}_{1,2}^{\lambda})^T = V_{2,2}^{\lambda}(V_{1,2}^{\lambda})^{-1},$$

and

$$\begin{bmatrix} R_{1,2}R_{2,2}^T \\ R_{2,2}R_{2,2}^T + \lambda I \end{bmatrix} = \begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix} (\mathcal{R}_{2,2}^\lambda)^T = \begin{bmatrix} V_{1,1}^\lambda \\ V_{2,1}^\lambda \end{bmatrix} (\Pi^\lambda (\mathcal{R}_{2,2}^\lambda)^T), \tag{5.39}$$

where $(\Pi^\lambda(\mathcal{R}_{2,2}^\lambda)^T)$ is upper triangular with all diagonal elements being positive and $\begin{bmatrix} V_{1,2}^\lambda \\ V_{2,2}^\lambda \end{bmatrix}$ is column orthogonal, which implies (5.39) is an economic QR factorization of matrix

$$\begin{bmatrix} R_{1,2}R_{2,2}^T \\ R_{2,2}R_{2,2}^T + \lambda I \end{bmatrix}.$$

Besides,

$$\begin{aligned} \begin{bmatrix} I \\ -(R_{2,2}R_{2,2}^T + \lambda I)^{-1} R_{2,2}R_{1,2}^T \end{bmatrix} &= \begin{bmatrix} I \\ -(\mathcal{R}_{2,2}^\lambda (\mathcal{R}_{2,2}^\lambda)^T)^{-1} \mathcal{R}_{2,2}^\lambda (\mathcal{R}_{1,2}^\lambda)^T \end{bmatrix} \\ &= \begin{bmatrix} I \\ -(\mathcal{R}_{2,2}^\lambda)^{-T} (\mathcal{R}_{1,2}^\lambda)^T \end{bmatrix} \\ &= \begin{bmatrix} V_{1,2}^\lambda \\ V_{2,2}^\lambda \end{bmatrix} (V_{1,2}^\lambda)^{-1} \end{aligned} \tag{5.40}$$

is an economic QR factorization of

$$\begin{bmatrix} I \\ -(R_{2,2}R_{2,2}^T + \lambda I)^{-1} R_{2,2}R_{1,2}^T \end{bmatrix} \quad \text{i.e.,} \quad \begin{bmatrix} I \\ -(\mathcal{R}_{2,2}^\lambda)^{-T} (\mathcal{R}_{1,2}^\lambda)^T \end{bmatrix},$$

since $(V_{1,2}^\lambda)^{-1}$ is a lower triangular with positive diagonal elements and $\begin{bmatrix} V_{1,2}^\lambda \\ V_{2,2}^\lambda \end{bmatrix}$ is column orthogonal.

By applying QR perturbation theory, Lemma 5.6, to QR factorizations (5.37) and (5.39), we have

$$\left\| \begin{bmatrix} V_{1,1}^\lambda - V_{1,1} \\ V_{2,1}^\lambda - V_{2,1} \end{bmatrix} \right\|_F \leq (1+\sqrt{2})\sqrt{\gamma - q} \frac{\left\| \begin{bmatrix} R_{1,2}R_{2,2}^T \\ R_{2,2}R_{2,2}^T \end{bmatrix}^{(+)} \right\|_2 \lambda}{1 - \lambda \left\| \begin{bmatrix} R_{1,2}R_{2,2}^T \\ R_{2,2}R_{2,2}^T \end{bmatrix}^{(+)} \right\|_2}$$

$$
\begin{aligned}
&= (1+\sqrt{2})\sqrt{\gamma-q}\frac{\left\|\begin{bmatrix}\mathcal{R}_{1,2}\mathcal{R}_{2,2}^T\\ \mathcal{R}_{2,2}\mathcal{R}_{2,2}^T\end{bmatrix}^{(+)}\right\|_2}{1-\lambda\left\|\begin{bmatrix}\mathcal{R}_{1,2}\mathcal{R}_{2,2}^T\\ \mathcal{R}_{2,2}\mathcal{R}_{2,2}^T\end{bmatrix}^{(+)}\right\|_2}\lambda\\
&= (1+\sqrt{2})\sqrt{\gamma-q}\frac{\left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}\begin{bmatrix}\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\\ I\end{bmatrix}^{(+)}\right\|_2}{1-\lambda\left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}\begin{bmatrix}\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\\ I\end{bmatrix}^{(+)}\right\|_2}\lambda\\
&\leq (1+\sqrt{2})\sqrt{n-q}\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2}\lambda,
\end{aligned}
\tag{5.41}
$$

in which the last inequality follows from

$$
\left\|\begin{bmatrix}\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\\ I\end{bmatrix}^{(+)}\right\|_2 = \frac{1}{\sigma_{\min}\left(\begin{bmatrix}\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\\ I\end{bmatrix}\right)} \leq 1,
$$

where $\sigma_{\min}(\cdot)$ denotes the minimum singular value of a matrix, provided

$$
\lambda\left\|\begin{bmatrix}R_{1,2}R_{2,2}^T\\ R_{2,2}R_{2,2}^T\end{bmatrix}^{(+)}\right\|_2 < 1. \tag{5.42}
$$

Similarly, by applying Lemma 5.6 to QR factorizations (5.38) and (5.40), and under the condition

$$
\begin{aligned}
1 &> \left\|\begin{bmatrix}I\\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\end{bmatrix}^{(+)}\right\|_2 \left\|(\mathcal{R}_{1,2}^{\lambda}(\mathcal{R}_{2,2}^{\lambda})^{-1})^T - (\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\right\|_2\\
&= \left\|\begin{bmatrix}I\\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\end{bmatrix}^{(+)}\right\|_2 \left\|(R_{2,2}R_{2,2}^T)^{-1}R_{2,2}R_{1,2}^T - (R_{2,2}R_{2,2}^T+\lambda I)^{-1}R_{2,2}R_{1,2}^T\right\|_2\\
&= \left\|\begin{bmatrix}I\\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\end{bmatrix}^{(+)}\right\|_2\\
&\quad\times \left\|(R_{2,2}R_{2,2}^T+\lambda I)^{-1}((R_{2,2}R_{2,2}^T+\lambda I)(R_{2,2}R_{2,2}^T)^{-1}-I)R_{2,2}R_{1,2}^T\right\|_2
\end{aligned}
$$

$$
\begin{aligned}
=&\lambda \left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2 \left\| (R_{2,2}R_{2,2}^T + \lambda I)^{-1}(R_{2,2}R_{2,2}^T)^{-1}R_{2,2}R_{1,2}^T \right\|_2 \\
=&\lambda \left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2 \left\| (\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T + \lambda I)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_2, \qquad (5.43)
\end{aligned}
$$

we have

$$
\begin{aligned}
&\frac{1}{1+\sqrt{2}} \left\| \begin{bmatrix} V_{1,2}^{\lambda} - V_{1,2} \\ V_{2,2}^{\lambda} - V_{2,2} \end{bmatrix} \right\|_F \\
\leq &\frac{\left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2}{1 - \left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2 \left\| (\mathcal{R}_{1,2}^{\lambda}(\mathcal{R}_{2,2}^{\lambda})^{-1})^T - (\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_2} \\
&\times \left\| (\mathcal{R}_{1,2}^{\lambda}(\mathcal{R}_{2,2}^{\lambda})^{-1})^T - (\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_F \\
= &\frac{\left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2}{1 - \lambda \left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2 \left\| (\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T + \lambda I)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_2} \\
&\times \left\| (\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T + \lambda I)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_F \lambda \\
\leq &\frac{\left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2}{1 - \lambda \left\| \begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)} \right\|_2 \left\| (\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_2} \\
&\times \left\| (\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_F \lambda \\
\leq &\frac{\left\| (\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_F}{1 - \lambda \left\| (\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \right\|_2} \lambda \\
\leq &\frac{\left\| \mathcal{R}_{2,2}^{-1} \right\|_2^2 \left\| \mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1} \right\|_F}{1 - \lambda \left\| \mathcal{R}_{2,2}^{-1} \right\|_2^2 \left\| \mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1} \right\|_2} \lambda, \qquad (5.44)
\end{aligned}
$$

where the second inequality in (5.44) follows from

$$\left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T+\lambda I)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\right\|_2 \le \left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\right\|_2,$$

$$\left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T+\lambda I)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\right\|_F \le \left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\right\|_F,$$

and the third inequality follows from

$$\left\|\begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)}\right\|_2 = \frac{1}{\sigma_{\min}\left(\begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}\right)} \le 1.$$

Moreover, since

$$\left\|\begin{bmatrix} R_{1,2}R_{2,2}^T \\ R_{2,2}R_{2,2}^T \end{bmatrix}^{(+)}\right\|_2 = \left\|\begin{bmatrix} \mathcal{R}_{1,2}\mathcal{R}_{2,2}^T \\ \mathcal{R}_{2,2}\mathcal{R}_{2,2}^T \end{bmatrix}^{(+)}\right\|_2 = \left\|\mathcal{R}_{2,2}^{-T}\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix}^{(+)}\right\|_2 \le \left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2,$$

and

$$\begin{aligned}
&\left\|\begin{bmatrix} I \\ -(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T \end{bmatrix}^{(+)}\right\|_2 \left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T+\lambda I)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\right\|_2 \\
&\le \left\|(\mathcal{R}_{2,2}\mathcal{R}_{2,2}^T+\lambda I)^{-1}(\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1})^T\right\|_2 \\
&\le \left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 \left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2,
\end{aligned}$$

for λ satisfying

$$\lambda \left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 (1+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2) < 1,$$

both (5.42) and (5.43) hold, an upper bound of the distance of V and V^λ derived from (5.41) and (5.44) is given by

$$\begin{aligned}
&\|V^\lambda - V\|_F \\
\le &\| \begin{bmatrix} V_{1,1}^\lambda - V_{1,1} \\ V_{2,1}^\lambda - V_{2,1} \end{bmatrix} \|_F + \| \begin{bmatrix} V_{1,2}^\lambda - V_{1,2} \\ V_{2,2}^\lambda - V_{2,2} \end{bmatrix} \|_F \\
\le &(1+\sqrt{2})\sqrt{n-q}\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2}\lambda + (1+\sqrt{2})\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2}\lambda \\
\le &(1+\sqrt{2})\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(\sqrt{n-q}+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F)}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(1+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2)}\lambda.
\end{aligned}$$

□

By Theorem 5.8, the optimal solution G^λ to optimization problem ROLDA (5.2) is of the form

$$G^\lambda = Q \begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & -(V_{2,2}^\lambda)^T W_{2,2}^\lambda \\ 0 & W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} W^\lambda, \tag{5.45}$$

where $W^\lambda \in \mathbf{R}^{l\times l}$ is orthogonal, $W_{2,2}^\lambda \in \mathbf{R}^{(\gamma-q)\times(l-q)}$, $W_{3,2}^\lambda \in \mathbf{R}^{(m-\gamma)\times(l-q)}$, and $\begin{bmatrix} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix}$ is column orthogonal. Let

$$G = Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & -V_{2,2}^T (V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ 0 & V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} W^\lambda, \tag{5.46}$$

where

$$\begin{aligned}
&\left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} -V_{2,2}^T (V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \right)^T \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \right. \\
&\quad \left. \times \begin{bmatrix} -V_{2,2}^T (V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \right) \\
&= \begin{bmatrix} -V_{1,2} V_{2,2}^T (V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix}^T \begin{bmatrix} -V_{1,2} V_{2,2}^T (V_{2,1}^{-T} (V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \\
&= \begin{bmatrix} V_{1,1} (V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix}^T \begin{bmatrix} V_{1,1} (V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \quad (\text{since } V_{1,2} V_{2,2}^T + V_{1,1} V_{2,1}^T = 0) \\
&= \left(\begin{bmatrix} V_{1,1} & \\ V_{2,1} & \\ & I \end{bmatrix} \begin{bmatrix} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \right)^T \left(\begin{bmatrix} V_{1,1} & \\ V_{2,1} & \\ & I \end{bmatrix} \begin{bmatrix} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} \right) = I,
\end{aligned}$$

$$\left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix} \right)^T \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} V_{1,2} \\ V_{2,2} \\ 0 \end{bmatrix}^T \begin{bmatrix} V_{1,2} \\ V_{2,2} \\ 0 \end{bmatrix} = I,$$

and

$$\left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix}\begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix}\right)^T \left(\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix}\begin{bmatrix} -V_{2,2}^T(V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix}\right)$$

$$= \begin{bmatrix} V_{1,2} \\ V_{2,2} \\ 0 \end{bmatrix}^T \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix}\begin{bmatrix} -V_{2,2}^T(V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix}$$

$$= \begin{bmatrix} I & V_{2,2}^T & 0 \end{bmatrix}\begin{bmatrix} -V_{2,2}^T(V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix} = 0,$$

it follows directly from Theorem 5.7 that $G \in \mathbf{R}^{m\times l}$ in (5.46) is a solution of the optimization problem OLDA (5.1).

The relationship between the two solutions (5.46) and (5.45) of optimization problems OLDA (5.1) and ROLDA (5.2), respectively, are shown as follows:

Theorem 5.10. *With notations in Theorems 5.7 and 5.8, for any solution $G^\lambda \in \mathbf{R}^{m\times l}$ of the optimization problem ROLDA (5.2) with $\lambda > 0$ satisfying*

$$\lambda \left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 (1 + \left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2) < 1,$$

there is a solution $G \in \mathbf{R}^{m\times l}$ of the optimization problem OLDA (5.1) such that

$$\left\|G^\lambda - G\right\|_F \le \left\|V^\lambda - V\right\|_F \le (1+\sqrt{2})\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 (\sqrt{n-q} + \left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F)}{1 - \lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 (1 + \left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2)}\lambda. \quad (5.47)$$

Proof. As shown above, the optimal solution G^λ to ROLDA (5.2) is of the form

$$G^\lambda = Q\begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix}\begin{bmatrix} I & -(V_{2,2}^\lambda)^T W_{2,2}^\lambda \\ 0 & W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} W^\lambda,$$

and

$$G = Q\begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix}\begin{bmatrix} I & -V_{2,2}^T(V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda) \\ 0 & V_{2,1}^{-T}(V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} W^\lambda,$$

is a solution of the optimization problem OLDA (5.1), where $W^{\lambda} \in \mathbf{R}^{l\times l}$ is orthogonal, $W^{\lambda}_{2,2} \in \mathbf{R}^{(\gamma-q)\times(l-q)}$, $W^{\lambda}_{3,2} \in \mathbf{R}^{(m-\gamma)\times(l-q)}$, and $\begin{bmatrix}(V^{\lambda}_{2,1})^T W^{\lambda}_{2,2}\\ W^{\lambda}_{3,2}\end{bmatrix}$ is column orthogonal. We have

$$
\begin{aligned}
&Q^T(G^{\lambda}-G)(W^{\lambda})^T\\
&=\begin{bmatrix}V^{\lambda}_{1,2} & 0 & 0\\ V^{\lambda}_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\begin{bmatrix}I & -(V^{\lambda}_{2,2})^T W^{\lambda}_{2,2}\\ 0 & W^{\lambda}_{2,2}\\ 0 & W^{\lambda}_{3,2}\end{bmatrix}-\begin{bmatrix}V_{1,2} & 0 & 0\\ V_{2,2} & I & 0\\ 0 & 0 & I\end{bmatrix}\begin{bmatrix}I & -V^T_{2,2}(V^{-T}_{2,1}(V^{\lambda}_{2,1})^T W^{\lambda}_{2,2})\\ 0 & V^{-T}_{2,1}(V^{\lambda}_{2,1})^T W^{\lambda}_{2,2}\\ 0 & W^{\lambda}_{3,2}\end{bmatrix}\\
&=\begin{bmatrix}V^{\lambda}_{1,2} & -V^{\lambda}_{1,2}(V^{\lambda}_{2,2})^T W^{\lambda}_{2,2}\\ V^{\lambda}_{2,2} & V^{\lambda}_{2,1}(V^{\lambda}_{2,1})^T W^{\lambda}_{2,2}\\ 0 & W^{\lambda}_{3,2}\end{bmatrix}-\begin{bmatrix}V_{1,2} & -V_{1,2}V^T_{2,2}V^{-T}_{2,1}(V^{\lambda}_{2,1})^T W^{\lambda}_{2,2}\\ V_{2,2} & V_{2,1}(V^{\lambda}_{2,1})^T W^{\lambda}_{2,2}\\ 0 & W^{\lambda}_{3,2}\end{bmatrix}\\
&=\begin{bmatrix}V^{\lambda}_{1,2}-V_{1,2} & (-V^{\lambda}_{1,2}(V^{\lambda}_{2,2})^T+V_{1,2}V^T_{2,2}V^{-T}_{2,1}(V^{\lambda}_{2,1})^T)W^{\lambda}_{2,2}\\ V^{\lambda}_{2,2}-V_{2,2} & (V^{\lambda}_{2,1}(V^{\lambda}_{2,1})^T-V_{2,1}(V^{\lambda}_{2,1})^T)W^{\lambda}_{2,2}\\ 0 & 0\end{bmatrix},
\end{aligned}
$$

in which,

$$
\begin{aligned}
&\begin{bmatrix}(-V^{\lambda}_{1,2}(V^{\lambda}_{2,2})^T+V_{1,2}V^T_{2,2}V^{-T}_{2,1}(V^{\lambda}_{2,1})^T)W^{\lambda}_{2,2}\\ (V^{\lambda}_{2,1}(V^{\lambda}_{2,1})^T-V_{2,1}(V^{\lambda}_{2,1})^T)W^{\lambda}_{2,2}\end{bmatrix}\\
&=\begin{bmatrix}-V^{\lambda}_{1,2}(V^{\lambda}_{2,2})^T-V_{1,1}(V^{\lambda}_{2,1})^T\\ (V^{\lambda}_{2,1}-V_{2,1})(V^{\lambda}_{2,1})^T\end{bmatrix}W^{\lambda}_{2,2}\ (\text{since } V_{1,1}V^T_{2,1}+V_{1,2}V^T_{2,2}=0)\\
&=\begin{bmatrix}-V^{\lambda}_{1,2} & -V_{1,1}\\ 0 & V^{\lambda}_{2,1}-V_{2,1}\end{bmatrix}\begin{bmatrix}(V^{\lambda}_{2,2})^T\\ (V^{\lambda}_{2,1})^T\end{bmatrix}W^{\lambda}_{2,2}.
\end{aligned}
$$

Thus,

$$
\begin{aligned}
&Q^T(G^{\lambda}-G)(W^{\lambda})^T\\
&=\begin{bmatrix}V^{\lambda}_{1,2}-V_{1,2} & -V^{\lambda}_{1,2} & -V_{1,1}\\ V^{\lambda}_{2,2}-V_{2,2} & 0 & V^{\lambda}_{2,1}-V_{2,1}\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}I & \\ & (V^{\lambda}_{2,2})^T\\ & (V^{\lambda}_{2,1})^T\end{bmatrix}\begin{bmatrix}I & \\ & W^{\lambda}_{2,2}\end{bmatrix}\\
&=\left(\begin{bmatrix}V^{\lambda}_{1,2}-V_{1,2} & 0 & V^{\lambda}_{1,1}-V_{1,1}\\ V^{\lambda}_{2,2}-V_{2,2} & 0 & V^{\lambda}_{2,1}-V_{2,1}\\ 0 & 0 & 0\end{bmatrix}+\begin{bmatrix}0 & -V^{\lambda}_{1,2} & -V^{\lambda}_{1,1}\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}\right)\begin{bmatrix}I & \\ & (V^{\lambda}_{2,2})^T W^{\lambda}_{2,2}\\ & (V^{\lambda}_{2,1})^T W^{\lambda}_{2,2}\end{bmatrix}
\end{aligned}
$$

$$= \begin{bmatrix} V_{1,2}^{\lambda} - V_{1,2} & 0 & V_{1,1}^{\lambda} - V_{1,1} \\ V_{2,2}^{\lambda} - V_{2,2} & 0 & V_{2,1}^{\lambda} - V_{2,1} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} I & \\ & (V_{2,2}^{\lambda})^T W_{2,2}^{\lambda} \\ & (V_{2,1}^{\lambda})^T W_{2,2}^{\lambda} \end{bmatrix} \text{ (since } V_{1,2}^{\lambda}(V_{2,2}^{\lambda})^T = -V_{1,1}^{\lambda}(V_{2,1}^{\lambda})^T)$$

$$= \begin{bmatrix} V_{1,1}^{\lambda} - V_{1,1} & V_{1,2}^{\lambda} - V_{1,2} & 0 \\ V_{2,1}^{\lambda} - V_{2,1} & V_{2,2}^{\lambda} - V_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & (V_{2,1}^{\lambda})^T W_{2,2}^{\lambda} \\ I & 0 \\ 0 & 0 \end{bmatrix}.$$

Note that matrix

$$\begin{bmatrix} (V_{2,1}^{\lambda})^T W_{2,2}^{\lambda} \\ W_{3,2}^{\lambda} \end{bmatrix}$$

is column orthogonal, we have

$$\begin{aligned} \left\| G^{\lambda} - G \right\|_F &\leq \left\| \begin{bmatrix} V_{1,1}^{\lambda} - V_{1,1} & V_{1,2}^{\lambda} - V_{1,2} & 0 \\ V_{2,1}^{\lambda} - V_{2,1} & V_{2,2}^{\lambda} - V_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix} \right\|_F \left\| \begin{bmatrix} 0 & (V_{2,1}^{\lambda})^T W_{2,2}^{\lambda} \\ I & 0 \\ 0 & 0 \end{bmatrix} \right\|_2 \\ &\leq \left\| V^{\lambda} - V \right\|_F \\ &\leq (1+\sqrt{2}) \frac{\left\| \mathcal{R}_{2,2}^{-1} \right\|_2^2 (\sqrt{n-q} + \left\| \mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1} \right\|_F)}{1 - \lambda \left\| \mathcal{R}_{2,2}^{-1} \right\|_2^2 (1 + \left\| \mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1} \right\|_2)} \lambda. \end{aligned}$$

□

LDA is a technique for data dimensionality reduction, it seeks an optimal linear transformation of the data to a low dimensional subspace, preferably the dimension of the reduced space is as small as possible. Hence, a solution of the optimization ROLDA (5.2) with minimum dimension is of particular interest. For such minimum solutions, the bound (5.47) can be simplified as follows:

Theorem 5.11. *With notations in Theorems 5.7 and 5.8, for any $\lambda > 0$, any solution G^{λ} of the optimization problem ROLDA (5.2) with minimum dimension is of the form*

$$G^{\lambda} = Q \begin{bmatrix} V_{1,2}^{\lambda} \\ V_{2,2}^{\lambda} \\ 0 \end{bmatrix} W^{\lambda} = Q(:, 1:n) \begin{bmatrix} V_{1,2}^{\lambda} \\ V_{2,2}^{\lambda} \\ 0 \end{bmatrix} W^{\lambda}, \tag{5.48}$$

where $W^\lambda \in \mathbf{R}^{q\times q}$ *is orthogonal. Moreover, for any such a minimum solution, there is a solution* $G \in \mathbf{R}^{m\times q}$ *of the optimization problem OLDA (5.1) such that*

$$\left\|G^\lambda - G\right\|_F \le (1+\sqrt{2})\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 \left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 \left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2}\lambda \tag{5.49}$$

provided

$$\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2 \left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2 < 1.$$

Proof. By Theorem 5.7,

$$G = Q\begin{bmatrix} V_{1,2} \\ V_{2,2} \\ 0 \end{bmatrix} W^\lambda = Q(:,1:n)\begin{bmatrix} V_{1,2} \\ V_{2,2} \\ 0 \end{bmatrix} W^\lambda \in \mathbf{R}^{m\times q}$$

is a solution of the optimization problem OLDA (5.1) with minimum dimension. For such G^λ and G, the inequality (5.49) follows directly from (5.44) in the proof of Lemma 5.9, provided

$$\lambda\|\mathcal{R}_{2,2}^{-1}\|_2^2\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\|_2 < 1.$$

□

Remark 5.4. *In numerical computing,* $Q(:,1:n)$ *in Theorem 5.11 can be computed efficiently as follows:*

- *Compute the economic QR factorization of A:*

$$A = Q_1 R,$$

  where $Q_1 \in \mathbf{R}^{m\times n}$ *is column orthogonal,* $R \in \mathbf{R}^{n\times n}$ *is upper triangular;*

- *Compute*

$$\begin{bmatrix} R_1 & R_2 & R_3 \end{bmatrix} := R\begin{bmatrix} \mathcal{H}_1 & & \\ & \ddots & \\ & & \mathcal{H}_k \end{bmatrix}\mathcal{P}\begin{bmatrix} \mathcal{H} & \\ & I \end{bmatrix},$$

  with $R_1 \in \mathbf{R}^{n\times 1}$, $R_2 \in \mathbf{R}^{n\times(k-1)}$, $R_3 \in \mathbf{R}^{n\times(n-k)}$, *where* $\mathcal{H}_i$ $(i = 1,\cdots,k)$, $\mathcal{H}$ *and* $\mathcal{P}$ *are defined in Chapter 3;*

- *Compute QR factorization of* $\begin{bmatrix} R_2 & R_3 \end{bmatrix}$ *with column pivoting as*

$$\begin{bmatrix} R_2 & R_3 \end{bmatrix} = Q_2 \begin{bmatrix} \hat{R}_2 & \hat{R}_3 \\ 0 & 0 \end{bmatrix},$$

  where $Q_2 \in \mathbf{R}^{n\times n}$ *is orthogonal,* $\hat{R}_2 \in \mathbf{R}^{\gamma\times(k-1)}$ *and* $\hat{R}_3 \in \mathbf{R}^{\gamma\times(n-k)}$ *with* $\gamma = \text{rank}\begin{bmatrix} R_2 & R_3 \end{bmatrix}$.

- *Compute QR factorization of* $\hat{R}_2$ *as*

$$\hat{R}_2 = Q_3 \begin{bmatrix} R_{1,1} \\ 0 \end{bmatrix},$$

  where $Q_3 \in \mathbf{R}^{\gamma\times\gamma}$ *is orthogonal and* $R_{1,1} \in \mathbf{R}^{q\times(k-1)}$ *with* $q = \text{rank}(R_2)$. *Denote*

$$\begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} := Q_3^T \hat{R}_3,$$

  where $R_{1,2} \in \mathbf{R}^{q\times(n-k)}$ *and* $R_{2,2} \in \mathbf{R}^{(\gamma-q)\times(n-k)}$.

- *Then*

$$\begin{bmatrix} A_2 & A_3 \end{bmatrix} = Q \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \\ 0 & 0 \end{bmatrix},$$

  where $Q(:, 1:n) = Q_1 Q_2 \begin{bmatrix} Q_3 & \\ & I \end{bmatrix}$.

This economic QR factorization of A can largely reduce computational cost in numerical experiments when $m >> n$, of which most large size undersampled data satisfy.

5.2.4 A New Regularized OLDA

In this section, we derive a mathematical criterion for choosing the regularization parameter λ in ROLDA and consequently we develop a new regularized orthogonal linear discriminant analysis method, in which no candidate set of regularization parameter is needed.

Theorem 5.10 implies that for any $G^\lambda \in \mathbf{R}^{m\times l}$ of the optimization problem ROLDA (5.2) there is a solution $G \in \mathbf{R}^{m\times l}$ of the optimization problem OLDA (5.1) such that

$$\left\|G^\lambda - G\right\|_F \leq (1+\sqrt{2})\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(\sqrt{n-q}+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F)}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(1+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2)}\lambda$$

when

$$\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2(1+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2)<1.$$

Thus, for any given small $\epsilon > 0$,

$$\|G^\lambda - G\|_F \leq \epsilon \tag{5.50}$$

provided that

$$\begin{aligned}\sqrt{\lambda} &\leq \frac{\sqrt{\epsilon}}{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2\sqrt{\epsilon(1+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2)+(1+\sqrt{2})(\sqrt{n-q}+\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F)}}\\ &\leq \frac{\sqrt{\epsilon}}{\left\|R_{2,2}^{(+)}\right\|_2\sqrt{\epsilon(1+\left\|R_{1,2}R_{2,2}^{(+)}\right\|_2)+(1+\sqrt{2})(\sqrt{n-q}+\left\|R_{1,2}R_{2,2}^{(+)}\right\|_F)}}.\end{aligned} \tag{5.51}$$

In particular, if $l = q$, Theorem 5.11 implies that for any $G^\lambda \in \mathbf{R}^{m\times l}$ of the optimization problem ROLDA (5.2) there is a solution $G \in \mathbf{R}^{m\times l}$ of the optimization problem OLDA (5.1) such that

$$\left\|G^\lambda - G\right\|_F \leq (1+\sqrt{2})\frac{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F}{1-\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2}\lambda,$$

with

$$\lambda\left\|\mathcal{R}_{2,2}^{-1}\right\|_2^2\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2<1,$$

then (5.50) holds provided

$$\begin{aligned}\sqrt{\lambda} &\leq \frac{\sqrt{\epsilon}}{\left\|\mathcal{R}_{2,2}^{-1}\right\|_2\sqrt{\epsilon\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_2+(1+\sqrt{2})\left\|\mathcal{R}_{1,2}\mathcal{R}_{2,2}^{-1}\right\|_F}}\\ &\leq \frac{\sqrt{\epsilon}}{\left\|R_{2,2}^{(+)}\right\|_2\sqrt{\epsilon\left\|R_{1,2}R_{2,2}^{(+)}\right\|_2+(1+\sqrt{2})\left\|R_{1,2}R_{2,2}^{(+)}\right\|_F}}.\end{aligned} \tag{5.52}$$

Let $\mathcal{N}_b$ denote a column orthogonal matrix whose columns span the null space of H_b^T, and $\mathcal{N}_b^{\perp}$ denote its orthogonal complement. Then we have

$$\left\|R_{2,2}^{(+)}\right\|_2 = \left\|(\mathcal{N}_b^T H_w)^{(+)}\right\|_2,$$

and

$$\left\|R_{1,2}R_{2,2}^{(+)}\right\|_2 = \left\|(\mathcal{N}_b^{\perp})^T H_w(\mathcal{N}_b^T H_w)^{(+)}\right\|_2,$$
$$\left\|R_{1,2}R_{2,2}^{(+)}\right\|_F = \left\|(\mathcal{N}_b^{\perp})^T H_w(\mathcal{N}_b^T H_w)^{(+)}\right\|_F.$$

As a result, inequalities (5.51) and (5.52) are reduced to

$$\sqrt{\lambda} \le \frac{\sqrt{\epsilon}}{\|(\mathcal{N}_b^T H_w)^{(+)}\|_2 \sqrt{(1+\eta_1)\epsilon + (1+\sqrt{2})(\sqrt{n-q}+\eta_2)}},$$

and

$$\sqrt{\lambda} \le \frac{\sqrt{\epsilon}}{\|(\mathcal{N}_b^T H_w)^{(+)}\|_2 \sqrt{\epsilon\eta_1 + (1+\sqrt{2})\eta_2}},$$

respectively, where

$$\eta_1 = \left\|(\mathcal{N}_b^{\perp})^T H_w(\mathcal{N}_b^T H_w)^{(+)}\right\|_2, \quad \eta_2 = \left\|(\mathcal{N}_b^{\perp})^T H_w(\mathcal{N}_b^T H_w)^{(+)}\right\|_F.$$

Now, we are ready to present our new ROLDA:

New ROLDA method:

- For a given small $\epsilon > 0$, select the regularization parameter λ by

  $$\sqrt{\lambda} \le \frac{\sqrt{\epsilon}}{\|(\mathcal{N}_b^T H_w)^{(+)}\|_2 \sqrt{(1+\eta_1)\epsilon + (1+\sqrt{2})(\sqrt{n-q}+\eta_2)}},$$

  if $l > q = \text{rank}(S_b) = \text{rank}(H_b)$; otherwise, select the regularization parameter λ by

  $$\sqrt{\lambda} \le \frac{\sqrt{\epsilon}}{\|(\mathcal{N}_b^T H_w)^{(+)}\|_2 \sqrt{\epsilon\eta_1 + (1+\sqrt{2})\eta_2}},$$

- Compute a solution $G^{\lambda} \in \mathbf{R}^{m\times l}$ of the optimization problem ROLDA (5.2).

5.3 Algorithms

5.3.1 Algorithm for OLDA

An implementation of optimization problem OLDA has been given in [96], this implementation computes the optimal linear transformation G of OLDA by computing some eigen-decompositions and involving some matrix inversions. However, the eigen-decomposition is computationally expensive [38], at least much more expensive than QR factorizations especially when the data size is very large, and the involvement of matrix inverses may lead to that the methods are not numerically stable if the related matrices are ill-conditioned [38]. In this section, we give the algorithm of computing an optimal solution G of optimization problem OLDA (5.1), which needs QR factorizations only.

Theorem 5.7 leads to the following implementation of the OLDA:

Algorithm 5.1. *(Implementation 1 of OLDA)*

Input: *Data matrix $A \in \mathbf{R}^{m\times n}$ with cluster label, cluster number k.*

Output: *Column orthogonal transformation matrix $G \in \mathbf{R}^{m\times l}$.*

Step 1. *Compute factorization*

$$[A_2 \quad A_3] = Q \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \\ 0 & 0 \end{bmatrix},$$

where $Q \in \mathbf{R}^{m\times m}$, $R_{1,1} \in \mathbf{R}^{q\times(k-1)}$, $R_{1,2} \in \mathbf{R}^{q\times(n-k)}$ and $R_{2,2} \in \mathbf{R}^{(\gamma-q)\times(n-k)}$ with $q = \operatorname{rank}(A_2)$, $\gamma = \operatorname{rank}[A_2 \quad A_3]$.

Step 2. *Compute the economic QR factorization of $R_{2,2}^T$ as*

$$R_{2,2}^T = \mathcal{V}_1 \mathcal{R}_{2,2}^T,$$

where $\mathcal{V}_1 \in \mathbf{R}^{(n-k)\times(\gamma-q)}$, $\mathcal{R}_{2,2} \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ is lower triangular with positive diagonal elements, and compute $\mathcal{R}_{1,2} = R_{1,2}\mathcal{V}_1$.

Step 3. *Compute the QR factorization of $\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix}$ as*

$$\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix} = \begin{bmatrix} V_{1,1} & V_{1,2} \\ V_{2,1} & V_{2,2} \end{bmatrix} \begin{bmatrix} \Pi \\ 0 \end{bmatrix},$$

by Remark 5.3, where $\Pi \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ *is upper triangular with positive diagonal elements,* $V_{1,2} \in \mathbf{R}^{q\times q}$ *is lower triangular with non-negative diagonal elements.*

Step 4. *Compute* $G \in \mathbf{R}^{m\times l}$ *by*

$$G = Q \begin{bmatrix} V_{1,2} & 0 & 0 \\ V_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & -V_{2,2}^T W_{2,2} - W_{3,1}^T W_{3,2} \\ 0 & W_{2,2} \\ W_{3,1} & W_{3,2} \end{bmatrix} \begin{bmatrix} W_{1,1} & \\ & I \end{bmatrix} W,$$

where $W \in \mathbf{R}^{l\times l}$ *is orthogonal,* $W_{1,1} \in \mathbf{R}^{q\times q}$ *is nonsingular,* $W_{3,1} \in \mathbf{R}^{(m-\gamma)\times q}$, $W_{2,2} \in \mathbf{R}^{(\gamma-q)\times(l-q)}$, $\begin{bmatrix} I \\ W_{3,1} \end{bmatrix} W_{1,1}$ *and* $\begin{bmatrix} V_{2,1}^T W_{2,2} \\ W_{3,2} \\ W_{3,1}^T W_{3,2} \end{bmatrix}$ *are column orthogonal.*

In Theorem 5.7, the economic QR factorization of $R_{2,2}^T$ and $\begin{bmatrix} \mathcal{R}_{1,2} \\ \mathcal{R}_{2,2} \end{bmatrix}$ are of special structures. For comparing our new ROLDA with OLDA, these special structures are important. However, for practical applications, these special structures are not need. We present another implementation of OLDA for the case $l = q$, which is much more simple than the above algorithm.

Algorithm 5.2. *(Implementation 2 of OLDA)*

Input: *Data matrix* $A \in \mathbf{R}^{m\times n}$ *with cluster label, cluster number* k.

Output: *Column orthogonal transformation matrix* $G \in \mathbf{R}^{m\times q}$.

Step 1. *Compute factorization*

$$[A_2 \quad A_3] = Q(:, 1:n) \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \\ 0 & 0 \end{bmatrix},$$

by Remark 5.4, where $Q \in \mathbf{R}^{m\times m}$, $R_{1,1} \in \mathbf{R}^{q\times(k-1)}$, $R_{1,2} \in \mathbf{R}^{q\times(n-k)}$ *and* $R_{2,2} \in \mathbf{R}^{(\gamma-q)\times(n-k)}$ *with* $q = \operatorname{rank}(A_2)$, $\gamma = \operatorname{rank}[A_2 \quad A_3]$.

Step 2. *Compute the economic QR factorization of* $R_{2,2}^T$ *as*

$$R_{2,2}^T = \mathcal{V}_1 \mathcal{R}_{2,2}^T,$$

where $\mathcal{V}_1 \in \mathbf{R}^{(n-k)\times(\gamma-q)}$, $\mathcal{R}_{2,2} \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$, *and compute* $\mathcal{R}_{1,2} = R_{1,2}\mathcal{V}_1$.

Step 3. *Compute the QR factorization of* $\begin{bmatrix}\mathcal{R}_{1,2}\\ \mathcal{R}_{2,2}\end{bmatrix}$ *as*

$$\begin{bmatrix}\mathcal{R}_{1,2}\\ \mathcal{R}_{2,2}\end{bmatrix} = \begin{bmatrix}V_{1,1} & V_{1,2}\\ V_{2,1} & V_{2,2}\end{bmatrix}\begin{bmatrix}\Pi\\ 0\end{bmatrix},$$

where $\Pi \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ *and* $V_{1,2} \in \mathbf{R}^{q\times q}$.

Step 4. *Compute* $G \in \mathbf{R}^{m\times q}$ *by*

$$G = Q(:, 1:n)\begin{bmatrix}V_{1,2}\\ V_{2,2}\\ 0\end{bmatrix} W,$$

where $W \in \mathbf{R}^{q\times q}$ *is orthogonal.*

The optimal transformation matrix G is obtained easily by several QR factorizations without computing any eigen-decomposition and matrix inverse, consequently, our implementation is inverse-free and numerically stable [38].

5.3.2 Algorithm for ROLDA

In this section, we present the algorithm to compute an optimal solution G^λ of ROLDA (5.2) based on Theorem 5.8, in addition, an improved version which is faster and easier to implement is given.

Theorem 5.8 and the upper bound of λ, (5.51) and (5.52), lead to the following numerical implementation of the new proposed ROLDA:

Algorithm 5.3. *(Implementation 1 of ROLDA)*

Input: *Data matrix* $A \in \mathbf{R}^{m\times n}$ *with cluster label, cluster number* k*, and a small* $\epsilon > 0$.

Output: *Column orthogonal transformation matrix* $G \in \mathbf{R}^{m\times l}$.

Step 1. *Compute factorization*

$$[A_2 \quad A_3] = Q\begin{bmatrix}R_{1,1} & R_{1,2}\\ 0 & R_{2,2}\\ 0 & 0\end{bmatrix},$$

where $Q \in \mathbf{R}^{m\times m}$, $R_{1,1} \in \mathbf{R}^{q\times(k-1)}$, $R_{1,2} \in \mathbf{R}^{q\times(n-k)}$ *and* $R_{2,2} \in \mathbf{R}^{(\gamma-q)\times(n-k)}$ *with* $q = \operatorname{rank}(A_2)$, $\gamma = \operatorname{rank}[A_2 \quad A_3]$.

Step 2. *If $l > q$ compute $\sqrt{\lambda}$ by*

$$\sqrt{\lambda} \leq \frac{\sqrt{\epsilon}}{\left\|R_{2,2}^{(+)}\right\|_2 \sqrt{\epsilon(1+\left\|R_{1,2}R_{2,2}^{(+)}\right\|_2)+(1+\sqrt{2})(\sqrt{n-q}+\left\|R_{1,2}R_{2,2}^{(+)}\right\|_F)}}.$$

If $l = q$ compute $\sqrt{\lambda}$ by

$$\sqrt{\lambda} \leq \frac{\sqrt{\epsilon}}{\left\|R_{2,2}^{(+)}\right\|_2 \sqrt{\epsilon\left\|R_{1,2}R_{2,2}^{(+)}\right\|_2+(1+\sqrt{2})\left\|R_{1,2}R_{2,2}^{(+)}\right\|_F}}.$$

Step 3. *Compute the economic QR factorization of $\begin{bmatrix} R_{2,2} & \sqrt{\lambda}I \end{bmatrix}^T$ as*

$$\begin{bmatrix} R_{2,2}^T \\ \sqrt{\lambda}I \end{bmatrix} = \mathcal{V}_1^\lambda (\mathcal{R}_{2,2}^\lambda)^T,$$

where $\mathcal{V}_1^\lambda \in \mathbf{R}^{(n-k+\gamma-q)\times(\gamma-q)}$ and $\mathcal{R}_{2,2}^\lambda \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ is lower triangular with positive diagonal elements, and compute $\mathcal{R}_{1,2}^\lambda = [R_{1,2} \quad 0]\,\mathcal{V}_1^\lambda$.

Step 4. *Compute the QR factorization of $\begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix}$ as*

$$\begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix} = \begin{bmatrix} V_{1,1}^\lambda & V_{1,2}^\lambda \\ V_{2,1}^\lambda & V_{2,2}^\lambda \end{bmatrix} \begin{bmatrix} \Pi^\lambda \\ 0 \end{bmatrix},$$

by Remark 5.3, where $\Pi^\lambda \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$ is upper triangular with positive diagonal elements, $V_{1,2}^\lambda \in \mathbf{R}^{q\times q}$ is lower triangular with non-negative diagonal elements.

Step 5. *Compute $G^\lambda \in \mathbf{R}^{m\times l}$ by*

$$G^\lambda = Q \begin{bmatrix} V_{1,2}^\lambda & 0 & 0 \\ V_{2,2}^\lambda & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} I & -(V_{2,2}^\lambda)^T W_{2,2}^\lambda \\ 0 & W_{2,2}^\lambda \\ 0 & W_{3,2}^\lambda \end{bmatrix} W^\lambda,$$

where $W^\lambda \in \mathbf{R}^{l\times l}$ is orthogonal, $W_{2,2}^\lambda \in \mathbf{R}^{(\gamma-q)\times(l-q)}$, $W_{3,2}^\lambda \in \mathbf{R}^{(m-\gamma)\times(l-q)}$, and $\begin{bmatrix} (V_{2,1}^\lambda)^T W_{2,2}^\lambda \\ W_{3,2}^\lambda \end{bmatrix}$ is column orthogonal.

In Theorem 5.8, the economic QR factorization of $\begin{bmatrix} R_{2,2} & \sqrt{\lambda}I \end{bmatrix}^T$ and the QR factorization of $\begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix}$ are of special structures. Like OLDA, for practical applications of our new ROLDA, these special structures are not needed. We present another implementation of ROLDA for the case that $l = q$ as follows.

Algorithm 5.4. *(Implementation 2 of ROLDA)*

Input: *Data matrix* $A \in \mathbf{R}^{m\times n}$ *with cluster label, cluster number* k*, and a small* $\epsilon > 0$.

Output: *Column orthogonal transformation matrix* $G \in \mathbf{R}^{m\times q}$.

Step 1. *Compute factorization*

$$[A_2 \quad A_3] = Q(:, 1:n) \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \\ 0 & 0 \end{bmatrix},$$

by Remark 5.4, where $Q \in \mathbf{R}^{m\times m}$, $R_{1,1} \in \mathbf{R}^{q\times(k-1)}$, $R_{1,2} \in \mathbf{R}^{q\times(n-k)}$ *and* $R_{2,2} \in \mathbf{R}^{(\gamma-q)\times(n-k)}$ *with* $q = \operatorname{rank}(A_2)$, $\gamma = \operatorname{rank}[A_2 \quad A_3]$.

Step 2. *Compute* $\sqrt{\lambda}$ *by*

$$\sqrt{\lambda} \le \frac{\sqrt{\epsilon}}{\left\|R_{2,2}^{(+)}\right\|_2 \sqrt{\epsilon \left\|R_{1,2}R_{2,2}^{(+)}\right\|_2 + (1+\sqrt{2})\left\|R_{1,2}R_{2,2}^{(+)}\right\|_F}}.$$

Step 3. *Compute the economic QR factorization of* $\begin{bmatrix} R_{2,2} & \sqrt{\lambda}I \end{bmatrix}^T$ *as*

$$\begin{bmatrix} R_{2,2}^T \\ \sqrt{\lambda}I \end{bmatrix} = \mathcal{V}_1^\lambda (\mathcal{R}_{2,2}^\lambda)^T,$$

where $\mathcal{V}_1^\lambda \in \mathbf{R}^{(n-k+\gamma-q)\times(\gamma-q)}$ *and* $\mathcal{R}_{2,2}^\lambda \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$*, and compute* $\mathcal{R}_{1,2}^\lambda = [R_{1,2} \quad 0]\,\mathcal{V}_1^\lambda$.

Step 4. *Compute the QR factorization of* $\begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix}$ *as*

$$\begin{bmatrix} \mathcal{R}_{1,2}^\lambda \\ \mathcal{R}_{2,2}^\lambda \end{bmatrix} = \begin{bmatrix} V_{1,1}^\lambda & V_{1,2}^\lambda \\ V_{2,1}^\lambda & V_{2,2}^\lambda \end{bmatrix} \begin{bmatrix} \Pi^\lambda \\ 0 \end{bmatrix},$$

where $\Pi^\lambda \in \mathbf{R}^{(\gamma-q)\times(\gamma-q)}$, $V_{1,2}^\lambda \in \mathbf{R}^{q\times q}$.

Step 5. *Compute* $G^\lambda \in \mathbf{R}^{m\times q}$ *by*

$$G^\lambda = Q(:, 1:n) \begin{bmatrix} V_{1,2}^\lambda \\ V_{2,2}^\lambda \\ 0 \end{bmatrix} W^\lambda,$$

where $W^\lambda \in \mathbf{R}^{q\times q}$ *is orthogonal.*

5.4 Numerical Experiments

In this section we perform extensive experiments to evaluate the efficiency of our new proposed ROLDA by comparing with OLDA and three existing regularized LDA methods SCRDA [39], RLDA [104, 54] and RDA [105].

Experimental Platforms[1] The experiments were conducted by using computer in Computer Center with 2.67GHz CPU and 16GB memory, National University of Singapore.

Experimental Data Sets: Our experiments were performed on the following 15 real-world data sets from three different sources, including face image and gene

Table 5.1: Data Structures

Type	Data	m	n			k
			total	training	test	
Gene Expression	*Brain*	5597	42	21	21	5
	Colon	2000	62	31	31	2
	Leukemia	3571	72	37	35	2
	Lymphoma	4026	62	32	30	3
	Prostate	6033	102	51	51	2
	SRBCT	2308	63	32	31	4
Face Image	$AR_{50\times40}$	2000	1680	840	840	120
	$AR_{50\times45}$	2250	1680	840	840	120
	Feret	6400	1000	600	400	200
	$ORL_{32\times32}$	1024	400	200	200	40
	$ORL_{64\times64}$	4096	400	200	200	40
	Palmprint	4096	600	300	300	100
	$Pictures_{64\times64}$	4096	565	290	275	55
	$Yale_{32\times32}$	1024	165	90	75	15
	$Yale_{64\times64}$	4096	165	90	75	15

[1]The computer processor in the experiment of this thesis is different from the one used in [19], thus the experimental results are different, especially the CPU time.

expression. The structures of these data sets are summarized in Table 5.1, where m is the dimension of data set, n is sample size and k is the total class number. For more description and sources of these datasets, please refer to Appendix C.

For all data sets used here, we performed our study by repeated random splitting into training and test sets using the following algorithm: within each class, we randomly reorder the data and then for each class with size n_i, the first $\lceil 0.5n_i \rceil$ data are used as the training data and the others are used as test data,whereby $\lceil \cdot \rceil$ is the ceiling function. The splitting was repeated 10 times, and the average results are recorded.

K-Nearest Neighbor method (K-NN) [30] with $K = 1$, based on the Euclidean distance, is used as the classification algorithm in this experiment. MATLAB command *cputime* is used to record the execution time of each algorithm.

5.4.1 Comparison with OLDA

In this subsection we compare our new ROLDA (Algorithm 5.3) with OLDA (Algorithm 5.1) from the perspective of classification accuracy. The distance between solutions of ROLDA and OLDA is also evaluated in terms of different tolerance ϵ.

As LDA seeks an optimal linear transformation of the data to a low dimensional subspace, preferably the dimension of the reduced space is as small as possible. In all our experiments for comparing our new ROLDA with OLDA and three existing regularized LDA methods SCRDA [39], RLDA [104] and RDA [105], we take $l = q = \text{rank}(S_b)$, and thus, for any given small $\epsilon > 0$, $\sqrt{\lambda}$ is given by

$$\sqrt{\lambda} = \frac{\sqrt{\epsilon}}{\left\| R_{2,2}^{(+)} \right\|_2 \sqrt{\epsilon \left\| R_{1,2} R_{2,2}^{(+)} \right\|_2 + (1+\sqrt{2}) \left\| R_{1,2} R_{2,2}^{(+)} \right\|_F}},$$

and a solution $G^\lambda \in \mathbf{R}^{m \times q}$ of optimization problem ROLDA (5.2) is reduced to

$$G^\lambda = Q(:, 1:n) \begin{bmatrix} V_{1,2}^\lambda \\ V_{2,2}^\lambda \\ 0 \end{bmatrix} \in \mathbf{R}^{m \times q}.$$

Consequently,

$$G = Q(:, 1:n) \begin{bmatrix} V_{1,2} \\ V_{2,2} \\ 0 \end{bmatrix} \in \mathbf{R}^{m \times q}$$

is an optimal solution of optimization problem OLDA (5.1).

For the comparison with OLDA, we set ϵ with five different values, that is

$$\epsilon = \begin{bmatrix} 1 & 10^{-1} & 10^{-2} & 10^{-3} & 10^{-4} \end{bmatrix}.$$

The resulting 1-NN average accuracies of OLDA (Algorithm 5.1) and ROLDA (Algorithm 5.3) with different parameter ϵ's as well as the distances between the solutions of OLDA and ROLDA are summarized in Table 5.2 below.

The main observations from Table 5.2 are:

- Our new proposed ROLDA produces similar classification accuracies for $\epsilon = \begin{bmatrix} 1 & 10^{-1} & 10^{-2} & 10^{-3} & 10^{-4} \end{bmatrix}$, thus, it is robust with parameter ϵ. Hence, in practical application, we can take, for example, $\epsilon = 10^{-2}$.

- ROLDA and OLDA yield similar classification accuracies. Therefore, our new ROLDA is comparative with OLDA.

- When ϵ decreases, the distance between the solutions of ROLDA and OLDA decreases adaptively, which coincides with our theoretical analysis about the relationship of ROLDA and OLDA.

Table 5.2: Comparison with OLDA

Data	Method	ϵ	$\left\|G^{\lambda} - G\right\|_F$	Accuracy	Standard Deviation
Brain	OLDA	-	-	86.19	6.88
	ROLDA	$1.0E0$	$1.11E-1$	85.71	6.73
		$1.0E-1$	$1.61E-2$	86.19	6.88
		$1.0E-2$	$1.69E-3$	86.19	6.88
		$1.0E-3$	$1.70E-4$	86.19	6.88
		$1.0E-4$	$1.70E-5$	86.19	6.88
Colon	OLDA	-	-	84.84	4.09
	ROLDA	$1.0E0$	$5.69E-2$	85.48	4.39
		$1.0E-1$	$8.47E-3$	85.16	3.87
		$1.0E-2$	$8.91E-4$	84.84	4.09
		$1.0E-3$	$8.96E-5$	84.84	4.09
		$1.0E-4$	$8.97E-6$	84.84	4.09
Leukemia	OLDA	-	-	97.14	1.81
	ROLDA	$1.0E0$	$6.96E-2$	97.14	1.81
		$1.0E-1$	$1.06E-2$	97.14	1.81
		$1.0E-2$	$1.11E-3$	97.14	1.81
		$1.0E-3$	$1.12E-4$	97.14	1.81
		$1.0E-4$	$1.12E-5$	97.14	1.81
Lymphoma	OLDA	-	-	100.00	0.00
	ROLDA	$1.0E0$	$5.70E-2$	100.00	0.00
		$1.0E-1$	$8.21E-3$	100.00	0.00
		$1.0E-2$	$8.59E-4$	100.00	0.00
		$1.0E-3$	$8.63E-5$	100.00	0.00
		$1.0E-4$	$8.64E-6$	100.00	0.00
Prostate	OLDA	-	-	90.78	2.64
	ROLDA	$1.0E0$	$3.81E-2$	90.98	3.06
		$1.0E-1$	$5.60E-3$	90.98	2.93
		$1.0E-2$	$5.88E-4$	90.98	2.93

Continued on next page

Table 5.2 – continued from previous page

Data	Method	ϵ	$\left\|G^{\lambda} - G\right\|_F$	Accuracy	Standard Deviation
		$1.0E-3$	$5.91E-5$	90.78	2.64
		$1.0E-4$	$5.92E-6$	90.78	2.64
$SRBCT$	OLDA	-	-	99.03	1.48
	ROLDA	$1.0E0$	$5.92E-2$	99.03	1.48
		$1.0E-1$	$8.59E-3$	99.03	1.48
		$1.0E-2$	$9.00E-4$	99.03	1.48
		$1.0E-3$	$9.04E-5$	99.03	1.48
		$1.0E-4$	$9.05E-6$	99.03	1.48
$AR_{50\times40}$	OLDA	-	-	97.58	0.41
	ROLDA	$1.0E0$	$1.54E-2$	97.61	0.43
		$1.0E-1$	$1.83E-3$	97.58	0.41
		$1.0E-2$	$1.86E-4$	97.58	0.41
		$1.0E-3$	$1.86E-5$	97.58	0.41
		$1.0E-4$	$1.86E-6$	97.58	0.41
$AR_{50\times45}$	OLDA	-	-	83.92	0.64
	ROLDA	$1.0E0$	$1.20E-2$	83.92	0.64
		$1.0E-1$	$1.45E-3$	83.92	0.64
		$1.0E-2$	$1.48E-4$	83.92	0.64
		$1.0E-3$	$1.48E-5$	83.92	0.64
		$1.0E-4$	$1.48E-6$	83.92	0.64
$Feret$	OLDA	-	-	85.95	1.76
	ROLDA	$1.0E0$	$5.10E-2$	86.05	1.75
		$1.0E-1$	$5.71E-3$	85.97	1.74
		$1.0E-2$	$5.78E-4$	85.95	1.76
		$1.0E-3$	$5.78E-5$	85.95	1.76
		$1.0E-4$	$5.78E-6$	85.95	1.76
	OLDA	-	-	96.25	1.71
		$1.0E0$	$4.43E-2$	96.25	1.71

Continued on next page

Table 5.2 – continued from previous page

Data	Method	ϵ	$\left\|G^{\lambda} - G\right\|_F$	Accuracy	Standard Deviation
$ORL_{32\times32}$	ROLDA	$1.0E-1$	$5.13E-3$	96.25	1.71
		$1.0E-2$	$5.21E-4$	96.25	1.71
		$1.0E-3$	$5.22E-5$	96.25	1.71
		$1.0E-4$	$5.22E-6$	96.25	1.71
$ORL_{64\times64}$	OLDA	-	-	96.85	1.05
	ROLDA	$1.0E0$	$2.56E-2$	96.85	1.05
		$1.0E-1$	$3.05E-3$	96.85	1.05
		$1.0E-2$	$3.11E-4$	96.85	1.05
		$1.0E-3$	$3.12E-5$	96.85	1.05
		$1.0E-4$	$3.12E-6$	96.85	1.05
$Palmprint$	OLDA	-	-	98.10	0.83
	ROLDA	$1.0E0$	$3.17E-2$	98.10	0.83
		$1.0E-1$	$3.80E-3$	98.10	0.83
		$1.0E-2$	$3.88E-4$	98.10	0.83
		$1.0E-3$	$3.88E-5$	98.10	0.83
		$1.0E-4$	$3.88E-6$	98.10	0.83
$Pictures_{64\times64}$	OLDA	-	-	93.02	1.29
	ROLDA	$1.0E0$	$1.89E-2$	93.02	1.29
		$1.0E-1$	$2.25E-3$	93.02	1.29
		$1.0E-2$	$2.30E-4$	93.02	1.29
		$1.0E-3$	$2.30E-5$	93.02	1.29
		$1.0E-4$	$2.30E-6$	93.02	1.29
$Yale_{32\times32}$	OLDA	-	-	82.93	3.52
	ROLDA	$1.0E0$	$4.19E-2$	83.07	3.82
		$1.0E-1$	$5.16E-3$	82.93	3.52
		$1.0E-2$	$5.28E-4$	82.93	3.52
		$1.0E-3$	$5.29E-5$	82.93	3.52
		$1.0E-4$	$5.29E-6$	82.93	3.52

Continued on next page

Table 5.2 – continued from previous page

Data	Method	ϵ	$\left\|G^{\lambda}-G\right\|_F$	Accuracy	Standard Deviation
$Yale_{64\times64}$	OLDA	-	-	89.07	3.31
	ROLDA	$1.0E0$	$2.93E-2$	89.20	3.29
		$1.0E-1$	$3.61E-3$	89.07	3.31
		$1.0E-2$	$3.69E-4$	89.07	3.31
		$1.0E-3$	$3.70E-5$	89.07	3.31
		$1.0E-4$	$3.70E-6$	89.07	3.31

5.4.2 Comparison with Some Existing Regularized LDA

In this subsection we compare our new ROLDA with three existing regularized LDA: SCRDA [39], RLDA [104, 54], and RDA [105] (which improves the RDA in [34]). We apply 5-fold cross-validation for parameter selection for SCRDA [39], RLDA [104, 54] and RDA [105]. For the parameter candidate sets, we take

- $\alpha = \begin{bmatrix} 0 & 0.2 & 0.4 & 0.6 & 0.8 & 0.9999 \end{bmatrix}$ and $\Delta = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 \end{bmatrix}$ for SCRDA [39];
- $\mu = \begin{bmatrix} 0 & 0.01 & 0.1 & 1 & 5 & 10 \end{bmatrix}$ for RLDA [104, 54];
- $\tau = \begin{bmatrix} 0 : 0.0333 : 1 \end{bmatrix}$ and $\beta = \begin{bmatrix} 0 : 0.0333 : 1 \end{bmatrix}$ for RDA [105];

To compare the efficiency with other regularized LDA, we use Algorithm 5.4 for ROLDA here. The computation of the regularization parameter is a core part of algorithms SCRDA[39], RLDA [104, 54] and RDA[105]. For Algorithm 5.4, the regularization parameter is computed in its Step 2, so Algorithm 5.4 also includes the computation of regularization parameter. Thus, the CPU times of SCRDA[39], RLDA [104], RDA[105] and our new ROLDA (Algorithm 5.4) include

the computation of the regularization parameter. The comparisons of average accuracy and CPU time of SCRDA[39], RLDA [104] and RDA[105] with our new ROLDA (Algorithm 5.4) with $\epsilon = 10^{-2}$ in 15 experiments are presented in Table 5.3.

The following observations can be made from Table 5.3:

- ROLDA does not need a candidate set of regularization parameter, this indicates that ROLDA is much faster than SCRDA [39], RLDA [104, 54], and RDA [105].
- ROLDA always produces reasonable classification accuracies.
- RLDA [104] is very fast compared with SCRDA [39] and RDA [105] since it requires only a single SVD for each fold cross validation.
- The classification performances of SCRDA [39], RLDA [104], and RDA [105] depend on the given candidate set of regularization parameter. If the given candidate set of regularization parameter is appropriate, they may achieve higher classification accuracies, but, if these candidate sets are not appropriate, they may lead to relative lower accuracies.
- It is still not clear how to choose an appropriate candidate set of regularization parameter for SCRDA [39], RLDA [104], and RDA [105]. In general, in order to obtain an appropriate regularization parameter for them, a larger candidate sets of regularization parameters can be chosen. However, for larger candidate sets, the computational complexities would increase significantly.

Table 5.3: Comparison with existing regularized LDA

Data	Method	CPU-time(s)	Accuracy	Standard Deviation
Brain	*SCRDA*	1403.66	80.95	7.68
	RLDA	0.15	83.33	4.88
	RDA	2.57	84.76	7.00
	ROLDA	0.02	86.19	6.88
Colon	*SCRDA*	137.27	85.48	5.45
	RLDA	0.09	85.81	4.61
	RDA	2.04	84.19	5.09
	ROLDA	0.01	84.84	4.09
Leukemia	*SCRDA*	425.60	97.43	1.54
	RLDA	0.18	97.14	1.81
	RDA	2.31	96.57	3.08
	ROLDA	0.01	97.14	1.81
Lymphoma	*SCRDA*	576.20	99.00	1.53
	RLDA	0.16	100.00	0.00
	RDA	2.60	99.00	1.53
	ROLDA	0.02	100.00	0.00
Prostate	*SCRDA*	2058.34	90.98	2.93
	RLDA	0.45	90.59	2.75
	RDA	4.14	90.39	3.66
	ROLDA	0.05	90.98	2.93
SRBCT	*SCRDA*	239.85	98.06	2.96
	RLDA	0.10	98.71	1.58
	RDA	3.49	98.06	2.14
	ROLDA	0.01	99.03	1.48
$AR_{50\times40}$	*SCRDA*	240.88	97.70	0.44
	RLDA	32.40	98.79	0.28
	RDA	36607.92	30.11	44.72
	ROLDA	5.09	97.58	0.41

Continued on next page

Table 5.3 – continued from previous page

Data	Method	CPU-time(s)	Accuracy	Standard Deviation
$AR_{50\times45}$	$SCRDA$	310.87	88.95	0.81
	$RLDA$	33.89	88.54	0.88
	RDA	41647.55	0.83	0.00
	$ROLDA$	5.69	83.92	0.64
$Feret$	$SCRDA$	3827.75	88.50	1.46
	$RLDA$	34.80	74.38	1.41
	RDA	31955.04	77.78	2.46
	$ROLDA$	4.12	85.95	1.76
$ORL_{32\times32}$	$SCRDA$	41.29	96.25	1.71
	$RLDA$	1.30	94.80	1.40
	RDA	455.88	95.05	1.27
	$ROLDA$	0.33	96.25	1.71
$ORL_{64\times64}$	$SCRDA$	1240.89	96.85	1.05
	$RLDA$	2.75	94.35	1.36
	RDA	515.53	95.20	1.49
	$ROLDA$	0.32	96.85	1.05
$Palmprint$	$SCRDA$	1078.46	98.53	0.98
	$RLDA$	5.93	99.07	0.51
	RDA	2290.12	98.83	0.82
	$ROLDA$	0.67	98.10	0.83
$Pictures_{64\times64}$	$SCRDA$	989.05	92.73	1.47
	$RLDA$	4.80	90.33	0.98
	RDA	1240.21	82.07	26.76
	$ROLDA$	0.77	93.02	1.29
$Yale_{32\times32}$	$SCRDA$	34.55	82.53	4.11
	$RLDA$	0.41	78.00	4.27
	RDA	32.46	76.00	6.37
	$ROLDA$	0.13	82.93	3.52

Continued on next page

Table 5.3 – continued from previous page

Data	Method	CPU-time(s)	Accuracy	Standard Deviation
$Yale_{64\times64}$	$SCRDA$	1253.69	88.93	3.32
	$RLDA$	0.75	92.27	2.44
	RDA	29.47	92.13	2.34
	$ROLDA$	0.08	89.07	3.31

5.5 Conclusions

In this chapter, the regularized orthogonal linear discriminant analysis has been studied. All solutions of optimization problems OLDA (5.1) and ROLDA (5.2), which are the aims and objectives in establishing OLDA and ROLDA, are explicitly characterized in Theorem 5.7 and Theorem 5.8. The mathematical relationship between the orthogonal linear discriminant analysis and the regularized orthogonal linear discriminant analysis is presented in Theorem 5.10 and Theorem 5.11. Based on this relationship, a mathematical criterion for choosing the regularization parameter in ROLDA is obtained and consequently a new regularized orthogonal linear discriminant analysis method has been proposed. The effectiveness of our new regularized orthogonal linear discriminant analysis has been demonstrated and confirmed by some real-world data sets.

Chapter 6

Conclusions and Future Work

Linear discriminant analysis (LDA) can be beneficial to reduce the dimension of the data not only for reasons of computational efficiency but also because it can improve the accuracy of the analysis. In this thesis, we have considered the theory, implementation, and applications of linear discriminant analysis.

Original LDA requires that a complete dataset for training is given in advance, and learning is carried out in one batch. To conquer this problem, incremental learning is studied in this thesis. Incremental methods have proven to enable efficient training if not all data is available in advance or if large amounts of training data have to be processed. This thesis provides a novel LDA-based incremental dimensionality reduction algorithm, called ILDA/QR, of which the batch version, LDA/QR, is a new proposed, simple and efficient implementation of LDA. As a fast LDA algorithm, LDA/QR is illustrated by several real-world datasets to be comparative with ULDA/QR from the perspective of classification accuracy with lower cost. The ILDA/QR algorithm has an equivalent power to batch algorithm LDA/QR in terms of dicriminability. More importantly, the ILDA/QR algorithm has the promising feature that all processed data can be discarded and that the update is less time and memory consuming than LDA/QR. This is desirable for large datasets. In addition, our new incremental algorithm ILDA/QR can easily handle not only the case that only one new sample is inserted but also the case

that a chunk of new samples are added. Experimental results show that ILDA/QR achieves high accuracies with low complexity in both time and space compared with other LDA-based incremental algorithms: IDR/QR, ILDA/SSS, LS-ILDA and ICLDA.

Classical LDA is not applicable to the singularity problem (or the undersampled problem). A generalized method of LDA, regularized orthogonal linear discriminant analysis is proposed to overcome this limitation. In our method, no parameter candidate set is needed and therefore preprocessing technique, such as cross-validation, for parameter selection is no longer required. All solutions of two optimization problems: orthogonal linear discriminant analysis and regularized orthogonal linear discriminant analysis, were explicitly characterized. Then the mathematical relationship between the orthogonal linear discriminant analysis and the regularized orthogonal linear discriminant analysis was presented. Based on this relationship, a mathematical criterion for choosing the regularization parameter in ROLDA was obtained and consequently a new regularized orthogonal linear discriminant analysis method has been proposed. Compared with some other regularized discriminant analysis algorithms, SCRDA, RLDA, and RDA which need to choose an appropriate regularization parameter by cross-validation, our algorithm is much faster and produces reasonable classification accuracies. An appropriate parameter was determined by the data matrix, while the classification performances of SCRDA, RLDA and RDA depend on the given parameter candidate set of regularization. With appropriate given candidate set, they may achieve higher classification accuracies, but, if these candidate sets are not appropriate, they may lead to relatively lower accuracies.

There are several challenging directions for future work:

In Chapter 3, ILDA/QR is proposed for efficient and incremental dimensionality reduction. The incremental LDA algorithm can also be incorporated into a classic semi-supervised learning framework and applied to many other problems

in which LDA-like discriminant components are required. It would be interesting to extend the incremental algorithm to semi-supervised or even unsupervised learning.

ILDA/QR is not applicable to the training samples that are linearly dependent. As mentioned in Section 3.5, to address this problem, regularization of the data matrix is a good approach. In the future, we aim to investigate some other methods to conquer the linearly independency of the data set.

In Chapter 5, we applied regularization method to linear discriminant analysis. The problem of generalizing LDA to provide more flexible discrimination by "kernelizing" the formulation has received much attention in recent years [42]. One key advantage of these kernel methods over other approaches is that they avoid the need to work explicitly in very high, possibly infinite, dimensional feature spaces, instead leading to problems whose "size" is bounded by the sample size. In addition, kernels can be defined to deal with much more general data types than those that are simply represented in a vector of numbers, e.g. sequences, trees, graphs and more general data. Because of the intrinsic raise in dimension of samples, essentially all problems in kernelized discriminant analysis become singular. Thus, it is common to introduce regularization terms to overcome the singularity problem. The importance of the related work is therefore more acute since there are additional parameters including kernel parameter(s) to be optimized. Hence, it is worthy to generalize the idea used in the present work to obtain mathematical criterions for choosing the regularization parameter(s) and develop appropriate regularized methods for kernelized discriminant analysis.

In addition, motivated by our numerical results in Table 5.2 of Chapter 5 and the fact that for any data items x and y,

$$| \left\|G^{\lambda}x - G^{\lambda}y\right\|_F - \|Gx - Gy\|_F | \leq \left\|G^{\lambda} - G\right\|_F (\|x\|_F + \|y\|_F),$$

which indicates $\left\|G^{\lambda}x - G^{\lambda}y\right\|_F$ is a good approximation of $\|Gx - Gy\|_F$ provided that G^{λ} is close to G, we assume implicitly that if the regularization parameter is selected by calibrating the solution of ROLDA to the solution of OLDA,

ROLDA can achieve a satisfactory classification performance similar to OLDA. For this assumption, it would be an interesting future research topic why a good approximation to the OLDA solution is a good criterion for the selection of the regularization parameter for ROLDA. Furthermore, if the solution of OLDA is a deficient one, can the corresponding selection of the regularization parameter produce a better solution of the ROLDA? These are interesting issues for our future research.

Bibliography

[1] A. A. Alizadeh, et al. Distinct Types Of Diffuse Large B-Cell Lymphoma Identified by Gene Expression Profiling, *Nature*, 403:503-511, 2000.

[2] U. Alon, N. Barkai, D. Notterdam, K. Gish, S. Ybarra, D. Mack, and A. Levine, Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays, *Proc Natl Acad Sci USA*, 96:6745-6750, 1999.

[3] E. Alpaydin, *Introduction to Machine Learning*, second edition, MIT Press, 2010.

[4] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe and H. van der Vorst, editors, *Templates for the solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, 2000.

[5] Z. Bai, J. Demmel, and M. Gu, An Inverse Free Parallel Spectral Divide and Conquer Algorithm for Nonsymmetric Eigenproblems, *Numer. Math.*, 76:279-308, 1997.

[6] P. Baldi, and G. W. Hatfield, *DNA Microarrays and Gene Expression: From Experiments to Data Analysis and Modeling*, Cambridge, 2002.

[7] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman, Eigenfaces vs. Fisherfaces: Recognition using Class Specific Linear Projection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711-720, 1997.

[8] M. Brand, Fast Low-rank Modifications of the Thin Singular Value Decomposition, *Linear Algebra and its Applications*, 415:20-30, 2006.

[9] R. Bellman, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, 1961.

[10] P. Benner, and R. Byers, An Arithmetic for Matrix Pencils: Theory and New Algorithms, *Numer. Math.*, 103:539-573, 2006.

[11] P. Benner, and R. Byers, Evaluating Products of Matrix Pencils and Collapsing Matrix Products, *Numerical Linear Algebra with Applications*, 8:357-380, 2001.

[12] M. W. Berry, S. T. Dumais, and G. W. O'Brien, Using Linear Algebra for Intelligent Information retrieval, *SIAM Review*, 37:573-595, 1995.

[13] S. A. Billings, and K. L. Lee, Nonlinear Fisher Discriminant Analysis using a Minimum Squared Error Cost Function and the Orthogonal Least Squares Algorithm, *Neural Network*, 15:263-270, 2002.

[14] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[15] D. Boley, M. Gini, R. Gross, E. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moorey, Document Categorization and Query Generation on the World Wild Web using WebACE, *Journal Artificial Intelligence Review - Special issue on data mining on the Internet* , 11:365-391, 1999.

[16] D. Boley, M. Gini, R. Gross, E.H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore, Partitioning-based Clustering for Web Document Categorization, *Journal Decision Support Systems - Special issue on WITS '97* , 27(3), Dec. 1999.

[17] D. Cai, X. He, J. Han, and H. Zhang Orthogonal Laplacianfaces for Face Recognition *IEEE Transactions on Image Processing,* 15(11):3608-3614, 2006

[18] L. Chen, H. M. Liao, M. Ko, J. Lin, and G. Yu, A New LDA-based Face Recognition System Which Can Solve the Samll Sample Size Problem, *Pattern Recognition*, 33:1713-1726, 2000.

[19] W. K. Ching, D. L. Chu, L. Z. Liao, and X. Y. Wang, Regularized Orthogonal Linear Discriminant Analysis, *Pattern Recognition*, 45(7):2719-2732, 2012.

[20] D. L. Chu, S. T. Goh, A New and Fast Orthogonal Linear Discriminant Analysis on Undersampled Problems, *SIAM Journal on Scientific Computing archive*, 32(4):2274-2297, 2010.

[21] D. L. Chu, and S. T. Goh, A New and Fast Implementation for Null Space Based Linear Discriminant Analysis, *Pattern Recognition*, 43:1373-1379, 2010.

[22] D. L. Chu, S. T. Goh, and Y. S. Hung, Characterization of All Solutions for Undersampled Uncorrelated Linear Discriminant Analysis Problems, *SIAM. J. Matrix Anal. and Appl.*, 32:820-844, 2011.

[23] D. L. Chu, L. De. Lathauwer, and B. De. Moor, A QR-type Reduction for Computing the SVD of a General Matrix Product/Quotient, *Numer. Math.*, 95:101-121, 2003.

[24] D. Q. Dai, and P. C. Yuen, Regularized Discriminant Analysis and its Application to Face Recognition, *Pattern Recognition*, 36:845-847, 2003.

[25] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart, Reorthogonalization and Stable Algorithms for Updating the Gram-Schmidt QR Factorization, *Mathematics of Computation,* 30:772-795, 1976.

[26] M. Dettling, BagBoosting for Tumor Classification with Gene Expression data, *Oxford Journals. Life Sciences. Bioinformatics*, 20(18):3583-3593, 2004.

[27] M. Dettling, and P. Bhlmann, Supervised Clustering of Genes, *Genome Biology*, 3(12):research0069.1-0069.15, 2002.

[28] L. Duchene, and S. Leclerq, An Optimal Transformation for Discriminant and Principal Component analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:978-983, 1988.

[29] S. Dudoit, J. Fridlyand, and T. P. Speed, Comparison of Discrimination Methods for the Classification of Tumors using Gene Expression Data, *Journal of the American Statistical Association*, 97:77-87, 2002.

[30] R. Q. Duda, P. E., Hart, and D. G. Stork, *Pattern Classification*, second edition, John Wiley and Sons, Inc., 2001.

[31] B. Efron, and R. Tibshirani, Improvements on Cross-Validation: The .632 + Bootstrap Method, *Journal of the American Statistical Association* 92 (438): 548-560, 1997.

[32] D. H. Foley, and J. W. Sammon, An Optimal Set of Discriminant Vectors, *IEEE Transactions on Computers,* 24(3):281-289, 1975.

[33] W. B. Frakes, and R. Baeza-Yates, *Information Retrieval: Data Structures and Algorithms*, Prentice Hall PTR, 1992.

[34] J. H. Friedman, Regularized Discriminant Analysis, *Journal of the American statistical association*, 84:165-175, 1989.

[35] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, second edition, Academic Press, Inc., 1990.

[36] A.S. Georghiades, P.N. Belhumeur and D.J. Kriegman, From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose *IEEE Trans. Pattern Anal. Mach. Intelligence,* 23(6):643-660, 2001

[37] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M.L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring, *Science*, 286(5439):531-537, 1999.

[38] G. H. Golub, and C. F. Van Loan, *Matrix Computations*, third edition, The Johns Hopkins University Press, Baltimore, Maryland, 1996.

[39] Y. Guo, T. Hastie, and R. Tibshirani, Regularized Linear Discriminant Analysis and Its Application in Microarray, *Biostatistics*, 8:86-100, 2007.

[40] P. Hall, J. S. Marron, and A. Neeman, Geometric Representation of High Dimensional, Low Sample Size Data, *J. Royal Statistical Society Series B*, 67:427-444, 2005.

[41] E. H. Han, D. Boley, M. Gini, R. Gross, E. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moorey, WebACE: A Web Agent for Document Categorization and Exploration, *2nd Intl. Conference on Autonomous Agents*. 408-415, 1997.

[42] R. F. Harrison and K. Pasupa, Sparse Multinomial Kernel Discriminant Analysis (sMKDA), *Pattern Recoginition*, 42:1795-1802, 2009

[43] R. F. Harrison, and K. Pasupa, Sparse Multinomial Kernel Discriminant Analysis (sMKDA), *Pattern Recognition*, 42:1795C1802, 2009.

[44] T. Hastie, A. Buja, R. Tibshirani, Penalized Discriminant Analysis, *Annals of Statistics*, 23:73C102, 1995.

[45] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2001.

[46] W. Hersh, C. Buckley, T. J. Leone and D. Hickam, OHSUMED: An Interactive Retrieval Evaluation and New Large Test Collection for Research. *Proc. ACM SIGIR*, pp. 192-201, 1994.

[47] G. Hinton, T. J. Sejnowski, *Unsupervised Learning: Foundations of Neural Computation*, MIT Press, 1999.

[48] P. Howland, M. Jeon, and H. Park, Structure Preserving Dimension Reduction for Clustered Text Data based on the Generalized Singular Value Decomposition, *SIAM J. Matrix Anal. Appl.*, 25:165-179, 2003.

[49] P. Howland, and H. Park, Generalizing Discriminant Analysis Using the Generalized Singular Value Decomposition, *IEEE Transations on Pattern Analysis and Machine Intelligence*, 26:995-1006, 2004.

[50] P. Howland, and H. Park. Two-stage Methods for Linear Discriminant Analysis: Equivalent Results at a Lower Cost, Technical Report GT-CSE, 2009.

[51] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edition, Springer, 2009.

[52] R. Huang, Q. Liu, H. Lu, and S. Ma, Solving the Small Smaple Size Problem of LDA, In *Proc. International Conference on Pattern Recognition*, pp.29-32, 2002.

[53] A. K. Jain, and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

[54] S. Ji, and J. Ye, Generalized Linear Discriminant Analysis: A Unified Framework and Efficient Model Selection, *IEEE Transations on Neural Networks*, 19:1768-1782, 2008.

[55] Z. Jin, J. Y. Yang, Z. S. Hu, and Z. Lou, Face Recognition based on the Uncorrelated Discriminant Transformation, *Pattern Recognition*, 34:1405-1416, 2001.

[56] Z. Jin, J. Y. Yang, Z. M. Tang, and Z. S. Hu, A Theorem on the Uncorrelated Optimal Discriminant vectors, *Pattern Recognition*, 34:2041-2047, 2001.

[57] I. T. Jolliffe, *Principal Component Analysis*, second edition, Springer-Verlag, New York, 2002.

[58] M. Jrgens, *Index Structures for Data Warehouses*, 1st edition, Springer, 2002.

[59] J. Khan, J. Wei, M. Ringner, L. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C.R. Antonescu, C. Peterson, and P. Meltzer, Classification and Diagnostic Prediction of Cancers Using Expression Profiling and Artificial Neural Networks, *Nature Medicine*, 7:673-679, 2001.

[60] H. Kim, P. Howland, and H. Park, Text Classification using Support Vector Machines with Dimension Reduction, *Proceedings of Text Mining Workshop of the 3rd SIAM International Conference on Data Mining, San Francisco, CA*, 2003.

[61] http://www.iis.ee.ic.ac.uk/˜ tkkim/code.htm

[62] T. K. Kim, S. F. Wong, B. Stenger, J. Kittler, and R. Cipolla, Incremental Linear Discriminant Analysis using Sufficient Spanning Set Approximations. *In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Minneapolis, MN*, 2007.

[63] T. K. Kim, B. Stenger, J. Kittler, and R. Cipolla, Incremental Linear Discriminant Analysis using Sufficient Spanning Sets and its Applications, *International Journal of Computer Vision*, 91(2):216-232, 2011.

[64] T. G. Kolda and D. P. O'Leary, A Semidiscrete Matrix Decomposition for Latent Semantic Indexing in Information Retrieval, *ACM Trans. Inf. Syst.,* 16:322-346, 1998.

[65] G. Kowalski, *Information Retrieval Systems: Theory and Implementation*, Kluwer Academic Publishers, 1997.

[66] K. C. Lee, J. Ho, and D. Kriegman, Acquiring Linear Subspaces for Face Recognition under Variable Lighting, *IEEE Trans. Pattern Anal. Mach. Intelligence*, 27(5):684-698, 2005.

[67] A. Lendasse, V. Wertz, and M. Verleysen, Model Selection with Cross-Validations and Bootstraps - Application to Time Series Prediction with RBFN Models, *Artificial Neural Networks and Neural Information Processing*, 573-580, 2003.

[68] C. Li, The Study on Indexing Techniques in Data Warehouse, *Key Engineering Materials*, Vols. 439 - 440, pp. 1505-1510, 2010.

[69] L. P. Liu, Y. Jiang, and Z. H. Zhou, Least Square Incremental Linear Discriminant Analysis, *Proceeding ICDM '09 Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, pp. 298-306, 2009.

[70] G. F. Lu, J. Zou, and Y. Wang, Incremental Learning of Complete Linear Discriminant Analysis for Face Recognition, *Knowledge Based Systems*, 31:19-27, 2012.

[71] L. Nanni, and A. Lumini, Orthogonal Linear Discriminant Analysis and Feature Selection for Micro-array Data Classification, *Expert Systems with Applications*, 37(10): 7132-7137, 2010.

[72] C. C. Paige, and M. A. Saunders, Towards a Generalized Singular Value Decomposition, *SIAM J. Numer. Anal.*, 18:398-405, 1981.

[73] H. Park, B. Drake, S. Lee, and C. Park, Fast Linear Discriminant Analysis Using QR Decomposition and Regularization, *Technical Report GT-CSE-07-21*, 2007.

[74] H. Park, M. Jeon, and J. B. Rosen, Lower Dimensional Representation of Text Data based on Centroids and Least Squares, *BIT*, 43:1-22, 2003.

[75] C. H. Park, and H. Park, A Relationship between Linear Discriminant Analysis and the Generalized Minimum Squared Error Solution, *SIAM J. Matrix Anal. Appl.*, 27:474-492, 2005.

[76] R. Polikar, L. Udpa, and V. Honavar, Learn ++: An Incremental Learning Algorithm for Supervised Neural Networks, *IEEE Trans. Syst., Man, Cybern.*, 31(4):497-508, 2001.

[77] S. Pomeroy et al. Prediction of Central Nervous System Embryonal Tumor Outcome Based on Gene Expression, *Nature*, 415:436-442, 2002.

[78] B. Scholkopf, and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press Cambridge, 2001.

[79] H. Shinnou, and M. Sasaki, Spectral Clustering for a Large Data Set by Reducing the Similarity Matrix Size, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, European Language Resources Association (ELRA), 2008.

[80] A. Statnikov, C. F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy, A Comprehensive Evaluation of Multicategory Classification Methods for Microarray Gene Expression Cancer Diagnosis, *Bioinformatics*, 2004.

[81] J. G. Sun, Perturbation Bounds for the Cholesky and QR Factoriztion, *BIT* 31:341-352, 1991.

[82] L. Sun, B. Ceran, and J. Ye, A Scalable Two-Stage Approach for a Class of Dimensionality Reduction Techniques, *Proceeding KDD '10 Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 313-322, 2010.

[83] D. L. Swets, and J. Weng, Using Discriminant Eigenfeatures for Image Retrieval, *IEEE Transations on Pattern Analysis and Machine Intelligence*, 18:831-836, 1996.

[84] TREC, Text Retrieval Conference, http://trec.nist.gov/, 1999.

[85] T. Tibshiranai, T. Hastie, B. Harashimhan and G. Chu, Class Prediction by Nearest Shrunken Centroids, with Applications to DNA Microarrays, *Statistical Science,* 18:104-117, 2003.

[86] S. Theodoridis, and K. Koutroumbas, *Pattern Recognition*, Academic Press, New York, 1999.

[87] K. Torkkola, Linear Discriminant Analysis in Document Classification, In *IEEE ICDM Workshop on Text Mining*, 2001.

[88] C. F. Van Loan, Generalizing the Singular Value Decomposition, *SIAM J. Numer. Anal.*, 13:76-83, 1976.

[89] L. Wang, and X. Shen, On L_1-norm Multicalss Support Vector Machines: Methodology and Theory, *Journal of the American Statistical Association*, 102:583-594, 2007.

[90] X. Wang, and X. Tang, Random Sampling LDA for Face Recognition, In *CVPR (2)*, pp. 259-265, 2004.

[91] X. Wang, and X. Tang, Random Sampling for Subspace Face Recognition, *International Journal of Computer Vision*, 70:91-104, 2006.

[92] D. S. Watkins, *Fundamentals of Matrix Computations*, Second Edition, New York, 2002.

[93] Whitehead Institute Center for Genomic Research: cancer genomics [http://www-genome.wi.mit.edu/cancer].

[94] W. Yang, D. Dai, and H. Yan, Feature Extraction and Uncorrelated Discriminant Analysis for High-Dimensional Data, *IEEE Transactions on Knowledge and Data Engineering*, 20(5):601-614, 2008.

[95] J. Yang, J. Y. Yang, Why can LDA be Performed in PCA Transformed Space? *Pattern Recognition*, 36(3):563-566, 2003.

[96] J. Ye, Characterization of a Family of Algorithms for Generalized Discriminant Analysis on Undersampled Problems, *J. Mach. Learn. Res.*, 6:483-502, 2005.

[97] J. Ye, Least Squares Linear Discriminant Analysis, In *Proceedings of the 24th International Conference on Machine Learning*, pp. 1087-1094, Corvallis, OR, 2007.

[98] J. Ye, R. Janardan, Q. Li, and H. Park, Feature Extraction via Generalized Uncorrelated Linear Discriminant Analysis, In *The Twenty-First International Conference on Machine Learning,* pp. 895-902, 2004.

[99] J. Ye, R. Janardan, C. H. Park, and H. Park, An Optimization Criterion for Generalized Discriminant Analysis on Undersampled Problems, *IEEE Transations on Pattern Analysis and Machine Intelligence*, 26:982-994, 2004.

[100] J. Ye, and Q. Li, A Two-stage Linear Discriminant Analysis via QR-Decomposition, *IEEE Transations on Pattern Analysis and Machine Intelligence*, 27:929-941, 2005.

[101] J. Ye, T. Li, T. Xiong, and R. Janardan, Using Uncorrelated Discriminant Analysis for Tissue Classification with Gene Expression Data, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(4):181-190, 2004.

[102] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, and V. Kumar, IDR/QR: An Incremental Dimension Reduction Algorithm via QR Decomposition, *IEEE Transactions on Knowledge and Data Engineering*, 17(9):1208-1222, 2005.

[103] J. Ye, and T. Xiong, Computational and Theoretical Analysis of Null Space and Orthogonal Linear Discriminant Analysis, *J. Mach. Learn. Res.*, 7:1183-1204, 2006.

[104] J. Ye, T. Xiong, Q. Li, R. Janardan, and J. Bi, Efficient Model Selection for Regularized Linear Discriminant Analysis, *Proceeding CIKM '06 Proceedings of the 15th ACM international conference on Information and knowledge management ,* 2006.

[105] J. Ye, and T. Wang, Regularized Discriminant Analysis for High Dimensional, Low Sample Size Data. *Proceeding KDD '06 Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining,* 2006.

[106] L. Zhang, L. Liao and M. Ng, Fast Algorithms for the Generalized Foley-Sammon Discriminant Analysis, *SIAM J. Matrix Anal. Appl.*, 31:1584-1605, 2010.

[107] W. Zhao, R. Chellappa, and P. Phillips, Subspace Linear Discriminant Analysis for Face Recognition, Technical Report CAR-TR-914, Center for Automation Research, University of Maryland, 1999.

[108] Y. Zhao, and G. Karypis, Criterion Functions for Document Clustering: Experiments and Analysis, *Technical Report TR 01–40, Department of Computer Science, University of Minnesota, Minneapolis, MN*, 2001.

[109] Y. Zhao, and G. Karypis, Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering, *Machine Learning*, 55(3):311-331, 2004.

[110] J. Zobel, A. Moffat, and R. Sacks-Davis, An Efficient Indexing Technique for Full Text Databases, In *Proceedings of 18th International Conference on Very Large Databases*, pp. 352-362, 1992.

Appendix A

Moore-Penrose Inverse and Trace Operator

Moore-Penrose Inverse:

For a matrix $X \in \mathbf{R}^{\mu\times\nu}$, a Moore-Penrose inverse of X is defined as a matrix $X^{(+)} \in \mathbf{R}^{\nu\times\mu}$ satisfying all the following four criteria:

(1) $(X^{(+)X})^T = X^{(+)}X$;

(2) $(XX^{(+)})^T = XX^{(+)}$;

(3) $XX^{(+)}X = X$;

(4) $X^{(+)}XX^{(+)} = X^{(+)}$.

The Moore-Penrose inverse exists and is unique. For any matrix , there is precisely one matrix , that satisfies the four properties of the definition. In the following, we list some useful properties of Moore-Penrose inverse:

1. When X is nonsingular, $X^{(+)} = X^{-1}$;

2. Let

$$X = U \begin{bmatrix} \Sigma & \\ & 0 \end{bmatrix} V^T$$

be the singular value decomposition of X, then

$$X^{(+)} = V \begin{bmatrix} \Sigma^{-1} & \\ & 0 \end{bmatrix} U^T,$$

where $U \in \mathbf{R}^{\mu \times \mu}$ and $V \in \mathbf{R}^{\nu \times \nu}$ are orthogonal, $\Sigma \in \mathbf{R}^{\gamma \times \gamma}$ is diagonal with positive diagonal entries, and $\gamma = \text{rank}(X)$;

3. Let $Y \in \mathbf{R}^{\nu \times \kappa}$, then $(XY)^{(+)} = Y^{(+)}X^{(+)}$ if and only if X is of full column rank and Y is of full row rank.

Trace Operator:

The trace of matrix $X = [x_{ij}] \in \mathbf{R}^{\mu \times \mu}$ is defined to be the sum of the elements on the main diagonal of X, i.e.,

$$\text{trace}(X) = \sum_{i=1}^{\mu} x_{ii}.$$

We list two useful properties of trace as follows:

1. $\text{trace}(X) = \sum_{i=1}^{\mu} \lambda_i(X)$, where $\lambda_i(X)$ is the eigenvalue of X, $i = 1, \cdots, \mu$;

2. Let $Y \in \mathbf{R}^{\mu \times \mu}$, then $\text{trace}(XY) = \text{trace}(YX)$.

Appendix B

Computational Complexity

We summarize computational cost for some matrix computation that are used in this thesis from [38, 4, 20, 70, 92] as follows:

1. Economic QR factorization of $X \in \mathbf{R}^{\mu \times \nu}$ $(\mu \geq \nu)$ needs

$$4\mu\nu^2 - \frac{4}{3}\nu^3 \text{ flops;}$$

2. Economic QR factorization of $X \in \mathbf{R}^{\mu \times \nu}$ $(\mu \geq \nu)$ with column pivoting needs

$$2\mu\nu^2 - \frac{2}{3}\nu^3 + (4\mu\nu\tau - 2p^2(\mu + \nu) + \frac{4}{3}\tau^3) \text{ flops, } \tau = \text{rank}(X);$$

3. Eigenvalue decomposition of $X \in \mathbf{R}^{\mu \times \mu}$ needs about

$$28\mu^3 \text{ flops;}$$

4. Eigenvalue decomposition of symmetric matrix $X \in \mathbf{R}^{\mu \times \mu}$ needs

$$12\mu^3 \text{ flops;}$$

5. Generalized eigenvalue decomposition of (X, Y) with $X,\ Y \in \mathbf{R}^{\mu \times \mu}$ are symmetric needs

$$14\mu^3 \text{ flops;}$$

6. Rank one updating of economic QR factorization $QR \in \mathbf{R}^{\mu\times\nu}$ with $Q \in \mathbf{R}^{\mu\times\nu}$ and $R^{\nu\times\nu}$, $Q(R + wv^T) = \tilde{Q}\tilde{R}$, needs

$$12\mu\nu + 6\nu^2 \text{ flops;}$$

where $\tilde{Q} \in \mathbf{R}^{\mu\times\nu}$, $\tilde{R} \in \mathbf{R}^{\nu\times\nu}$ and $w, v \in \mathbf{R}^{\nu}$.

7. QR-updating of economic QR factorization $QR \in \mathbf{R}^{\mu\times\nu}$ with $Q \in \mathbf{R}^{\mu\times\nu}$ and $R^{\nu\times\nu}$, $\begin{bmatrix} Q & q \end{bmatrix} \begin{bmatrix} R \\ z^T \end{bmatrix} = \tilde{Q}\tilde{R}$, needs

$$6\mu\nu + 3\nu^2 \text{ flops.}$$

where $\tilde{Q} \in \mathbf{R}^{\mu\times\nu}$, $\tilde{R} \in \mathbf{R}^{\nu\times\nu}$, $q \in \mathbf{R}^{\mu}$ is orthogonal to Q and $z^T \in \mathbf{R}^{\nu}$ is the new inserted row;

8. QR-updating of inserting one row to full QR factorization $QR \in \mathbf{R}^{\mu\times\nu}$ with $Q \in \mathbf{R}^{\mu\times\mu}$ and $R^{\mu\times\nu}$, $\begin{bmatrix} QR \\ z^T \end{bmatrix} = \begin{bmatrix} Q & \\ & 1 \end{bmatrix} \begin{bmatrix} R \\ z^T \end{bmatrix} = \tilde{Q}\tilde{R}$, needs

$$6\mu\nu + 3\nu^2 \text{ flops.}$$

where $\tilde{Q} \in \mathbf{R}^{(\mu+1)\times(\mu+1)}$, $\tilde{R} \in \mathbf{R}^{(\mu+1)\times\nu}$, and $z^T \in \mathbf{R}^{\nu}$ is the new inserted row.

9. QR-updating of inserting one column to full QR factorization $QR \in \mathbf{R}^{\mu\times\nu}$ with $Q \in \mathbf{R}^{\mu\times\mu}$ and $R^{\mu\times\nu}$, $\begin{bmatrix} QR & z \end{bmatrix} = \tilde{Q}\tilde{R}$, needs

$$6\mu\nu + 3\nu^2 \text{ flops.}$$

where $\tilde{Q} \in \mathbf{R}^{\mu\times\mu}$, $\tilde{R} \in \mathbf{R}^{\mu\times(\nu+1)}$, and $z \in \mathbf{R}^{\mu}$ is the new inserted column.

10. Solving $X \in \mathbf{R}^{\mu\times\nu}$ from the upper triangular system $BX = Y$ with $B \in \mathbf{R}^{\mu\times\mu}$ is upper triangular and $Y \in \mathbf{R}^{\mu\times\nu}$ needs

$$\mu^2\nu \text{ flops.}$$

11. Solving $X \in \mathbf{R}^{\mu\times\mu}$ from the nonlinear system $BX = Y$ with $B,\ Y \in \mathbf{R}^{\mu\times\mu}$ by Gaussian elimination needs

$$\frac{7}{3}\mu^3 \text{ flops.}$$

Appendix C

Datasets

In this thesis, we have used three different types of data to evaluate our algorithms, including text document, face image and gene expression. For convenience and future reference, we summarize the description and sources of these datasets here.

Text Document:

- $Tr12$, $Tr23$ these two datasets are derived from TREC collection [84]. The processed datasets are also available at

  http://shi-zhong.com/software/docdata.zip.

  The categories correspond to the documents relevant to particular queries.

- Wap, $K1b$ and $K1a$ are from the WebACE project [15] [41] [16], where each document corresponds to a web page listed in the subject hierarchy of Yahoo (http://www.yahoo.com). The datasets $k1a$ and $K1b$ contain exactly the same set of documents but they differ in how the documents were assigned to different classes. In particular, $K1a$ contains a finer-grain categorization than that contained in $K1b$.

Gene Expression:

- $Brain$ tumor dataset, presented in [77], contains $n = 42$ microarray gene expression profiles from $k = 5$ different tumors of the central nervous system,

that is, 10 medulloblastomas, 10 malignant gliomas, 10 atypical teratoid/ rhabdoid tumors (AT/RTs), 8 primitive neuro-ectodermal tumors (PNETs) and 4 human cerebella. The raw data were originated using the Affymetrix technology and are publicly available at [93]. The processed dataset is available at

ftp://stat.ethz.ch/Manuscripts/dettling/brain.rda.

- *Colon* cancer dataset contains the expression levels of 40 tumor and 22 normal colon tissues for 6,500 human genes that are measured using the Affymetrix technology. A selection of 2,000 genes with highest minimal intensity across the samples has been made in [2]. It was further processed in [27] and the dataset is available at

  ftp://stat.ethz.ch/Manuscripts/dettling/colon.rda.

- *Leukemia* data set consists of samples from patients with either acute lymphoblastic leukemia (ALL) or acute myeloid leukemia (AML). See [37] for a complete description of the data set. You can download the data set from

  http://stat.ethz.ch/~dettling/bagboost.html.

- *Lymphoma* is a data set of the three most prevalent adult lymphoid malignancies. Which has been studied in [26]. The data set is available at

  http://stat.ethz.ch/~dettling/bagboost.html.

- *Prostate* cancer raw data are available at [93] and comprise the expression of 52 prostate tumors and 50 non-tumor prostate samples, obtained using the Affymetrix technology. It was processed in [27] and the dataset is available at

  ftp://stat.ethz.ch/Manuscripts/dettling/prostate.rda.

- *SRBCT* (Small Round Blood Cell Tumor)[59] dataset has 2308 genes and 63 experimental conditions, 8 Burkitt Lymphoma (BL), 23 Ewing Sarcoma (EWS), 12 neuroblastoma (NB), and 20 rhabdomyosarcoma (RMS). Which can be downloaded from

  http://www.stat.cmu.edu/~jiashun/Research/software/Data/SRBCT/.

Face Image:

- *AR* database consists of 4,000 color images corresponding to 126 people's faces (70 men and 56 women). Images feature frontal view faces with different facial expressions, illumination conditions, and occlusions, second sessions repeated same conditions. In our experiment, we selected pictures of 120 individuals (65 men and 55 women) in two sessions. 28 face images, 14 for each session, were used for each individual. The face portion of each image was cropped to 50×40 pixels for $AR_{50\times40}$ and 50×45 pixels for $AR_{50\times45}$, respectively. *AR* face dataset was available at

  http://cobweb.ecn.purdue.edu/~aleix/aleix_face_DB.html

- *Feret* face database contains 1564 sets of images for a total of $14,126$ images that includes 1199 individuals and 365 duplicate sets of images. A duplicate set is a second set of images of a person already in the database and was usually taken on a different day. Which is available at

  http://www.itl.nist.gov/iad/humanid/feret/feret_master.html.

  A subset of *Feret* database was used in our experiment. This subset includes 1000 sets of images of 200 individuals each of which has 5 images. It consists of the images marked with two lowercase character string: *ba*, *bj*, *bk*, *be* and *bf*. The facial portion of each original image was cropped and resized to 80×80 pixels.

- *ORL* dataset contains a set of ten different images, each of which has 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions and facial details. The database can be retrieved from

  http://www.cl.cam.ac.uk/Research/DTG/attarchive:pub/data/att_faces.tar.Z.

  In our experiment, the size of each image is resized to 64×64 pixels for $ORL_{64\times64}$ and 32×32 pixels for $ORL_{32\times32}$, respectively.

- *Palmprint* database is available at

http://www4.comp.polyu.edu.hk/ biometrics/.

We selected 100 different palms from this database. Around 6 samples from each of these palms were collected in two sessions, where 3 samples were captured in the first session and the second session, respectively. All images were compressed to 64×64 pixels.

- $Yale$ database Contains 165 grayscale images in GIF format of 15 individuals. There are 11 images per subject, one per different facial expression or configuration: center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, sleepy, surprised, and wink. Each image of data set $Yale_{32\times32}$ was resized to 32×32 pixels, while each image of data set $Yale_{64\times64}$ was resized to 64×64 pixels. $Yale$ face data was available at

  http://cvc.yale.edu/projects/yalefaces/yalefaces.html.

- $YaleB$ database includes both of the original Yale Face Database B [36] with 10 subjects and the extended Yale Face Database B [36] with 28 subjects. Each subject has 65 (64 illuminations + 1 ambient) images in a particular pose. There were 47 of them of which the corresponding strobe did not go off. All image data were manually aligned, cropped, and then re-sized to 168x192 images [66]. The database of $YaleB$ is available at

  http://vision.ucsd.edu/~leekc/ExtYaleDatabase/Yale%20Face%20Database.htm.

INCREMENTAL AND REGULARIZED LINEAR DISCRIMINANT ANALYSIS

WANG XIAOYAN

NATIONAL UNIVERSITY OF SINGAPORE

2012