Fast Implementation of Linear Discriminant Analysis

Goh Siong Thye

NATIONAL UNIVERSITY OF SINGAPORE

2009

Fast Implementation of Linear Discriminant Analysis

Goh Siong Thye

(B.Sc.(Hons.) NUS)

A THESIS SUBMITTED FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF MATHEMATICS

NATIONAL UNIVERSITY OF SINGAPORE

2009

Acknowledgements

First of all, I would like to thank my advisor, Prof Chu Delin for his guidance and patience. He has been a great mentor to me since my undergraduate study. He is a very kind, friendly and approachable advisor and he has introduced me to this area of Linear Discriminant Analysis. I have learnt priceless research skill under his mentorship. This thesis would not be possible without his valuable suggestions and creative ideas.

Furthermore, a project in Computational Mathematics would not be complete without simulations on some real life data. Collecting the data on our own would be costly and I feel very grateful to Prof Li Qi, Prof Ye Jieping, Prof Haesun Park and Prof Li Xiao Fei for their donations of data sets to make our simulation possible. Their previous works and advices have been a valuable to us.

I would also like to thank the lab assistants who have rendered a lot of help to us to manage the data as well as The NUS HPC team, especially Mr. Yeong and Mr. Wang who have been assisting us with their technical knowledge in handling large memory requirement of our project. I also feel grateful for the facility of Centre for Computational Science and Engineering that enable us to run more programmes.

I would also like to thank my family and friends for their supports for all these years. Special thanks goes to Weipin, Tao Xi, Jiale, Huey Chyi , Hark Kah, Siew Lu, Wei Biao, Anh, Wang yi, Xiaoyan and Xiaowei

Last but not least, Rome was not built in a day. I was thankful for meeting a lot of outstanding educators over these few years for nurturing my mathematical maturity.

Contents

1 Introduction **8**

1.1 Significance of Data Dimensionality Reduction 8

1.2 Applications 11

1.3 Curse of Dimensionality 16

2 An Introduction to Linear Discriminant Analysis **19**

2.1 Generalized LDA 24

2.2 Alternative Representation of Scatter Matrices 26

3 Orthogonal LDA **30**

3.1 A Review of Orthogonal LDA 30

3.2 A New and Fast Orthogonal LDA 31

3.3 Numerical Experiment 38

3.4 Simulation Output 41

3.5 Relationship between Classical LDA and Orthogonal LDA 44

4 Null Space Based LDA **49**

4.1 Review of Null Space Based LDA 49

4.2 New Implementation of Null Space Based LDA 52

4.3 Numerical Experiments 55

4.4 Simulation Output 57

4.5 Relationship between Classical LDA and Nullspace Based LDA 58

5 Conclusion **64**

Appendix **66**

Summary

Improvement of technology has been tremendously fast and we have access to varieties of data. However, the irony is that with so much information, it is very hard to manage and manipulate them. This gave birth to an area of computing science called data mining. It is the art of finding important information and from there we can make better decision, save storage cost as well as manipulate data at a more affordable price.

In this thesis, we will look at one particular area of data mining, called linear discriminant analysis. We will give a brief survey of the history as well as the varieties of the method later, including incremental approach and some other types of implementation. One common method that is used in the implementation is Singular Value Decomposition (SVD) which is very expensive. Thereafter we will review two special types of implementation called Orthogonal LDA as well as Null Space Based LDA. We will also propose improvements to the algorithm. The improvement stands apart from other implementation as it doesn't involve any inverse, SVD and it is a numerically stable. The main tool that we used is QR decomposition, which is very fast and the time saved is very significant. Numerical simulations were carried out and numerical results are reported in this thesis as well. Furthermore, we will also reveal some relationship between these variants of linear discriminant analysis.

List of Tables

Table 1.1: Silverman's Estimation

Table 3.1: Data Dimensions, Sample Size and Number of Cluster

Table 3.2: Comparison of Classification Accuracy for OLDA/new and OLDA

Table 3.3: Comparison of CPU time for OLDA/new and OLDA

Table 4.1: Comparison of Classification Accuracy for NLDA 2000, NLDA 2006 and NLDA/new

Table 4.2: Comparison of CPU time for NLDA 2000, NLDA 2006 and NLDA/new

List of Figures

Figure 1.1: Visualization of Iris Data after Feature Reduction

Figure 1.2: Classification of Handwritten Digits

Figure 1.3: Varieties of Facial Expressions

Figure 1.4: Classification of Hand Signals

Notation

1. The letter $A \in \mathbb{R}^{m \times n}$ means the data matrix given where each column a_i represents a single data point; hence our original data are data of size $m \times 1$ and n is the total number of data given to us.

2. The matrix $G \in \mathbb{R}^{m \times l}$ is the standard matrix that represents the linear transformation that we desired to use. Pre-multiplication of it to a vector in $m-$ dimensional space to $l-$ dimensional space.

3. k represents the number of classes in the data sets.

4. n_i is the total number of data in the i-th class.

5. e is the all ones vector, of which the size will be mentioned.

6. c_i represents the class centroid of the i-th class while c represent the global centroid.

7. S_b, S_w and S_t are symbols of scatter matrices in the original space, of which we will define soon.

8. S_b^L, S_w^L and S_t^L are symbols of scatter matrices in the reduced space, of which we will define soon.

9. K is the number of neighbours considered in the K- Nearest Neighbours Algorithm.

Chapter 1

Introduction

1.1 Significance of Data Dimensionality Reduction

This century is the "century of data", while traditionally, researchers might only record down a handful of features due to technology constraint, now data can be collected easily. DNA microarrays, biomedical spectroscopy, high dimensional video, and tick by tick financial data, these are just a few means to obtain high dimensional data. The collection of data might be an expensive process either economically or computational, and hence it would be a great waste to the owner of the data if their data remains not interpreted. To read the data manually and find the intrinsic relationship would be a great challenge, fortunately, computers have avoided mankind from suffering from these routine and mundane jobs, after all, as one can imagine that picking needle from the hay is not a trivial job. Dimension reduction is crucial in this manner in the sense that we have to find those factors that are contributing to the phenomenon that we observe, usually a big phenomenon might be caused by only a handful of reasons while the rest are just noise that make things fuzzy. Furthermore, it might be tough in the sense that the real factor might be a combination of a few attributes that we observe directly, making the task more and more challenging. Modeling is necessary because of this factor; the simplest one is by assuming normal distribution and the classes linearly separable.

Mathematics of dimension reduction and heuristic approaches in this aspect had been emphasized in many parts of the world especially for research community. As what John

Tukey, one of the greatest Mathematician and Computer Scientist said, it is time for us to admit the fact that there is going to be many data analysts while we only have relatively much fewer mathematicians and statisticians, hence it is crucial for us to invest resources into the research in this area, to ensure high quality of practical application that we will discuss later. [46] A scheme to solve the problem to extract the crucial information is not sufficient, for those, we already have some results over the years. More importantly, we are certainly in need of an efficient scheme that guarantees high accuracy. There are various schemes available currently. Some of them are more general while some are more application specific. In either case, we still have room for improvement for these technologies.

A typical case of dimensionality reduction is as follows:

A set of data is given to us; they might be clustered or not clustered, in the event the class labels are given to us, we said that it is supervised learning, otherwise we call it unsupervised learning, both areas are hot research areas. However, in this thesis, we will focus on the supervised case.

Suppose that the data are given to us in the form of

$$A = [A_1, \ldots, A_k]$$

where each column vector represent a data point and A_i is the collection of the i-th class, and where k is the number of classes, the whole idea is to devise a mapping $f(.)$ such that when a new data, x is given to us, $f(x)$ is a projection of x to a vector of much smaller size, maintaining the class information and help us to classify it to an existing group. Even the most trivial case, where we intend to find optimum linear projector, still has a lot of room for improvements. The question can be generalized in the sense that some classes may have more than one mode or we can even made it more complicated, some data can belong to more than one classes. This is not a purely theoretical problem as modern world requires human to multi-task and it is easy to see that a person can be

both an entertainer and a politician. It seems that existing algorithms still have room for improvement. There are many things for us to investigate in this area, for a start, what objective function should we use? Majority of the literatures have used trace or determinant to measure the dispersion of the data. However, to maximize the distance between the classes and minimize the distance within classes are impossible to be performed simultaneously, what types of trade off should we adopt? These are interesting problems that are suitable for a data driven society nowadays.

There are many other motivations for us to reduce the dimension, for example, storing every feature, say 10^6 pixels or features would cost us a lot of memory space, however, usually after feature reduction is being performed, most likely we would only be keeping a few features, such as 10^2 or 10 features, in another word, it is possible to cut down the storage cost by 10^4 times! Rather than developing tools such as thumb drives with more memory capacity to store all the information no matter how important it is, it would be wiser to extract only the most crucial information. Besides saving up the memory space, in the event that we intend to perform computations on the data that is given to us, for example, computing the SVD of a matrix of size $10^6 \times 10^6$ matrices would be much more expensive compared to computing the SVD of 10×10 matrix, the cost for the earlier case would cost us around 10^{18} flops while the latter one only cost us 10^3 flops.

Numerical simulations have also verified that by doing feature extraction, we can increase the accuracy in classification as we have removed the noise from the data. Having accurate classification is crucial as sometimes it might determine how much profit or even cost a life. For example, by doing feature reduction on Leukemia data, we can tell what are the main features that determine someone is a patient and ways to cure a patient might be designed from there. Feature reduction can in fact effectively identify the trait that is common to certain disease and push life sciences research ahead.

Another motivation of dimension reduction would be to enable visualization. At higher dimension, visualization is almost impossible as we live in a 3 dimensional world. If we can reduce the dimension to 2 or 3 dimensions, we will be able to visualize it.

For example, Iris data which consist of 4 features, 3 clusters cannot be visualized easily. Comparing pair wise every single feature need not be meaningful to distinguish the features. A feature reduction was carried out and we obtain a figure as shown below

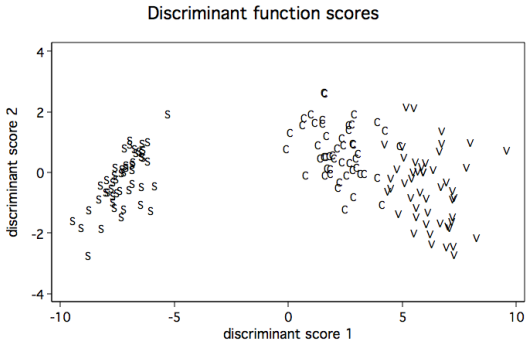

Figure 1.1: Visualization of Iris Data after Feature Reduction

We can now visualize how closed a species is related to the other and Randolf's conjecture which state how the species are related was verified by R. A. Fisher back in 1936 [44]. This shows that the applications of dimension reduction can be linked to other areas and we will show several more famous applications in the next subsection to illustrate this point.

1.2 Applications

Craniometry The importance of feature reduction can be traced back to even before the years of invention of computers. In 1936, Fisher suggested feature reduction to the area of Craniometry, namely, the study of bones. Data from bones were being reduced to identify the gender of the humans and the lifestyle when the human was alive. This area is still relevant nowadays to identify victims of crime scene or accident casualties. The only different back and now is that nowadays we have computers to help us speed up our computation and as a consequence, we can handle larger scale data.

Classification of Handwritten Digits It would be easy to ask a computer to distinguish two digits that are printed as usually two printed same digits is highly similar to each other no matter whether they are in various font types. However, asking a computer to distinguish handwritten digits would be a much greater challenge. After

all, daily experience tells us that some people's '3' resembles '5' and some people's '5' resemble '6' even for human eyes, it would be harder for us to teach the computers how to distinguish them apart. The goal here actually is to teach the computer to be even smarter than human's eye, being able to identify clearly badly written digits.

Figure 1.2: Classification of Handwritten Digits

This application is crucial if let say we want to design a machine to classify letters in post office since many still write zip code or postcode manually, this would make the operation in post office much more efficient. Also, the application can also be extended to identify alphabetical letters, other characters or distinguish signatures, hence cutting down the fraud cases.

A simple scheme to classify the data would be to consider each digit as a class and we compute the class means. From there, we take each data as a vector and when we are given a new data, we just compute the nearest mean and classify the new data to that group. Experiments have shown that by doing that the accuracy is around 75%. By using some numerical linear algebra, one can compute its SVD of the data matrix, and when a new data is given, we can compute the residuals in each basis and classify accordingly, by doing that, the accuracy has increased by a bit, the best performance is 97% but some performance can be as low as 80% only as people's handwriting can be very hard to identify. Tangent distance can be computed to solve this problem and only QR decomposition is needed.[47]

The accuracy is very high but in term of efficiency of classification, there is still room for improvement in this area. Various approaches have been adopted such as preprocessing the data and smoothing the images. In particular, in the classical implementation of Tangent distance approach, each test data is compared with every single training data, dimensional reduction might be suitable here as we can save some costs of computing the norm. For instance, if 256 pixels were taken into consideration and if there is no dimension reduction, the cost would be very high, if an algorithm called LDA is adopted, the SVD that we need to compute would cut down the cost by 10^4 times.

Text mining One researcher might like to search for relevant journal to read by using some search engine like $Google^{TM}$. The search engine must be able to identify the relevant documents given the keywords. It has been well known in the past that for a search engine to do so; the search engine must not be overly reliant on the physical appearance of the keywords, more importantly; the search engine must be able to identify the concept and return relevant materials. To do so, Google has invested a lot of research in this area. It is crucial for the algorithm to be efficient to attract more people to use their product so as to attract more advertisers and collect more data to understand consumer's interest or trend of current days.

Day by day, the database increases rapidly, new websites are being created, new documents are being uploaded and latest news is being reported, maintaining the efficiency would become tougher and tougher.

If one has very high dimensional data vector to deal with, the processing speed is going to be slow and to store all the information would be ridiculously tough. Hence we can design an incremental updating dimension reduction algorithm here to able to return the latest information to the consumers to beat the competitor.

Currently we already has incremental version of dimension reduction algorithm but there are still room for improvement in this aspect. The approximation used currently might be too crude in some aspects. Furthermore, there are rarely any theoretical results

to support the approximation. Most of the time, assumptions are just stated but these assumptions are not known to whether hold in general.

Furthermore, studies have shown that in high dimensional space, the maximum distances and minimum distances in high dimensional space are almost the same for a wide variety of distance functions and data distributions [48], this makes a proximity query such as K- nearest neighbours algorithm meaningless and unstable because there is poor discrimination between the nearest and furthest neighbour. Hence, a small relative perturbation of the target in a direction away from the nearest neighbour could easily change the nearest neighbour into the furthest and vice versa. Hence this makes the classification meaningless. Hence this provides us with another motivation to perform dimension reduction on this application. By doing dimension reduction, we are not comparing document term wise, but rather conceptual wise.

Facial Recognition The invention of digital camera and phone cameras have enabled layman to create high definition pictures easily. These are pictures with many pixels. Hence, a lot of features are captured. It is relatively easy for human to tell apart two humans, but for a computer, to tell two people apart might be tougher. Inside a picture, there are only a few features that can tell two people apart and yet to do so, due to curse of dimensionality of which we will discuss later, we will need to take a lot of pictures. The same person might be very difficult for a computer to identify once we change the environment, for example, we can change the viewing angle, different illumination, different poses, gestures, attire and many other factors. Due to this reason, facial recognition has become a very hot research topic.

It would be very slow if we use all the pixels to compare the individuals as they are high definition pictures with a lot of pixels, hence in this case, dimension reduction is necessary. One popular method to solve this problem would be null space based LDA and methods have been introduced to create artificial pictures to reduce the effect of viewing angles. It is crucial to be able distinguish a few people rapidly. [49]

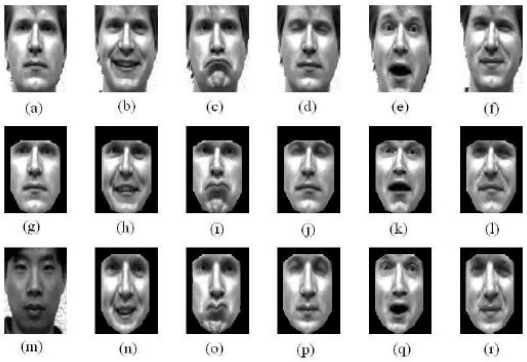

Figure 1.3: Varieties of Facial Expressions

For security purposes, some industry might think that it is too risky to create just a card for the employees to access restricted places. Hence, other human parts such as fingerprints and eyes has also been used to distinguish people nowadays, hence it would be great for us to deepen our research in this area.

Microarray Analysis Human Genomes Project should be a familiar term to many. Many scientists are interested to study the genome of human. One application of this is to tell apart those who have certain diseases from those who don't and if possible, identify what are the key things to look for to identify the diseases. As we know, the size of human genomes data is formidable. Out of such a high dimensions, to pick up what are the gene that tell us we have a certain disease is not simple. Dimension reduction would be great in this area. We have conducted a few numerical experiments and the accuracy of Leukemia diseases can be up to 95% and we believe that we can increase our accuracy and efficiency in doing this.

Another application is identifying the gene that control our alcohol tolerance, for example, currently most experiments are modeled based on simpler animal such as flies as their genetic structure is much simpler compared to humans. Dimension reduction might enable us to identify individuals who are alcohol intolerant and advise the patients accordingly. [50]

Financial Data It is well known that stock market is highly unpredictable in the sense that it can be bearish in a moment and be bullish in the very next moment. The

factor that affects the performance of a stock is hard to manage, making a lot of hedging and derivative pricing a tough job to perform. Given a set of features, we might like to identify from the set of information given what are the features that is highly responsible for stocks with high returns and stock with low returns.

Others There are various other applications, as long as the underlying problem can be converted into high dimensional data and we desire to find the intrinsic structures of the data, feature reduction is suitable. For example, we can identify potential customers by looking at consumers behaviours in the past and it is also useful in general machine learning. For instance, if we want to create a machine to identify signal sent by human positioning of hands and make the machine being able to response without human being there, we can train a machine to read the signal and count the fingers and perform corresponding task that is suitable. High accuracy is essential if this is really needed to be realized, as at certain angle, 5 fingers might overlapped be seen as one finger to human eyes, the position of each fingers are essential for this application and a good dimension reduction algorithm should pick this property up provided the raw data does report this phenomenon.

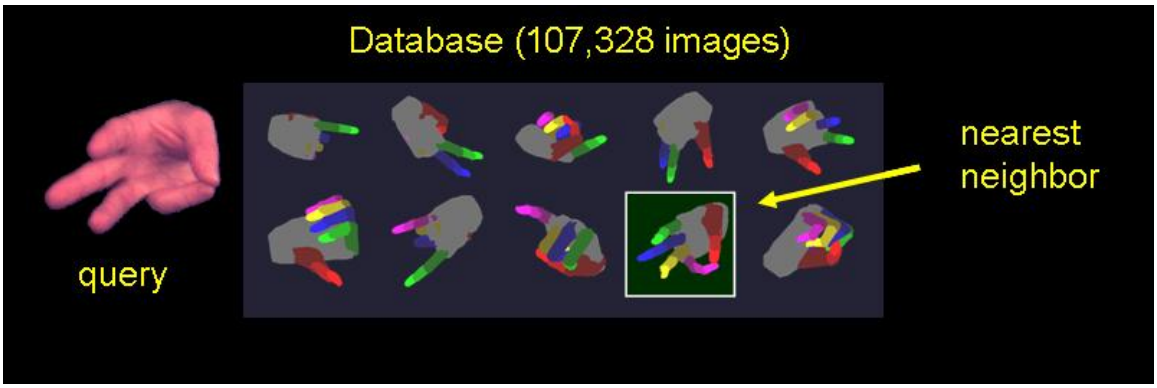

Figure 1.4: Classification of Hand Signals

1.3 Curse of Dimensionality

Coined by Richard Bellman, the curse of dimensionality is a term used to describe the problem caused by the exponential increase in volume associated with adding extra dimensions to a mathematical space.

As the dimension increases, we need more samples to describe the model well. This can be seen from the fact that as we increase the dimension, most likely we will include more noise as well and sometimes, if we collect too little data, we might be misguided by the wrong representation of the data, for example, we might accidentally keep collecting data from the tails of a distributions and it is obvious that we are not going to get a good representation of data. However, the increase of additional sample points needed would be so rapid that it is very expensive to cope with that.

Various works have been done to attempt to overcome this problem, for example Silverman [45] has provided us with a table illustrating the difficulty of kernel estimation in high dimensions. To estimate the density at 0 with a given accuracy, he reported his estimation in the table below.

Table 1.1: Silverman's estimation

Dimensionality	Required Sample Size
1	4
2	19
5	786
7	10700
10	842000

As we can see the sample size required increases tremendously, a rough idea why this is so can be modeled based on a model of a hypersphere of radius r inscribed in a hypercubes or side length $2r$.

The volume of the hypercube will be $(2r)^d$ but the volume of the hypersphere will be $\frac{2r^d\pi^{\frac{d}{2}}}{d\Gamma(\frac{d}{2})}$ where d is the dimension of the data and $\Gamma(.)$ is the Gamma function. Unfortunately, we can prove that the ratio of the volume of the hypersphere inscribed in the hypercubes will converge to zero, in other word, this implies that it is going to be very hard to obtain data that represent the central part of the data as the dimension increases.

For example, in database community, one important issue is the issue of indexing. A

number of techniques such as KDB-trees, kd-Trees and Grid Fields are discussed in the classical database literature for indexing multidimensional data. These methods generally work well for very low dimensional problems but they degrade rapidly with the increase of dimensions. Each query requires the access of almost all the data. Theoretical and empirical results have shown the negative effects of increasing dimensionality on index structures.[51]

In this research area, the phenomenon is hidden in the form of singularity of matrices, such as for the facial recognition application that we have described above, with so many pixels, to make sure that the so called "total scatter matrix" is non-singular, we have to collect more and more picture, this would be very time consuming and impractical. Mathematics to solve the problems need to be further developed to overcome or avoid this curse such as avoid computing the inverse of such matrices. There are various heuristic approaches to overcome this problem for example by taking pseudoinverse, perform a Tirkhonov inverse or perform GSVD. Which of the generalization is better theoretically and computationally, these are problems that are worth investigating.

Other application of dimensionality reduction, its applications and computational issue, in particular, linear discriminant analysis (LDA) which will be discussed later to overcome Curse of Dimensionality can be found at [1],[2],[3],[4],[5],[6],[7],[8],[9],[10],[11], [15],[16],[17],[18],[19],[20],[21],[23],[24],[25],[26], [28],[30],[31],[32],[33],[34], [35],[36],[39],[40].

Chapter 2

An Introduction to Linear Discriminant Analysis

Given a data matrix $A \in \mathbb{R}^{m \times n}$, where n columns of A represent n data items in a m dimensional space. Any linear transformation $G^T \in \mathbb{R}^{l \times m}$ can map a vector x in the m dimensional space to a vector y in the l dimensional space,

$$G^T : \ x \in \mathbb{R}^{m \times 1} \ \rightarrow \ y \in \mathbb{R}^{l \times 1},$$

where l is an integer with $l << m$. Our goal is to find an optimal linear transformation $G^T \in \mathbb{R}^{l \times m}$ such that the cluster structure in the original data is preserved in the reduced l-dimensional space, assuming that the given data is already clustered. For this purpose, a measure of cluster quality has to be established first. Of course, to have high cluster quality, a specific clustering result must have a tight within-cluster relationship while the between-cluster relationship has to be remote. To quantify this, within-cluster, between-cluster and mixture scatter matrices are defined in discriminant analysis.

Let the data matrix A be partitioned into k clusters as

$$A = \left[\begin{array}{cccc} a_1 & a_2 & \cdots & a_n \end{array} \right] = \left[\begin{array}{cccc} A_1 & A_2 & \ldots & A_k \end{array} \right]$$

where

$$A_i \in \mathbb{R}^{m \times n_i}, \quad i = 1, \cdots, k, \qquad \text{and} \qquad \sum_{i=1}^{k} n_i = n.$$

Further, let

$$e = \begin{bmatrix} 1 & \dots & 1 \end{bmatrix}^T \in \mathbb{R}^{n\times 1}, e_i = \begin{bmatrix} 1 & \dots & 1 \end{bmatrix}^T \in \mathbb{R}^{n_i\times 1}, \quad i = 1, \dots, k,$$

and denote the set of column indices that belong to the cluster i by N_i. The centroid $c^{(i)}$ and the global centroid are given by

$$c^{(i)} = \frac{1}{n_i} A_i e_i, \quad i = 1, \dots, k,$$

and

$$c = \frac{1}{n} Ae,$$

respectively. Then the between-cluster scatter matrix S_b, the within-cluster scatter matrix S_w and the total scatter matrix are defined as

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

It is well-known [16] that $S_t = S_b + S_w$. Let

$$\begin{aligned} H_b &= \begin{bmatrix} \sqrt{n_1}(c^{(1)} - c) & \dots & \sqrt{n_k}(c^{(k)} - c) \end{bmatrix} \in \mathbb{R}^{m\times k}, \\ H_w &= \begin{bmatrix} A_1 - c^{(1)} e_1^T & \dots & A_k - c^{(k)} e_k^T \end{bmatrix} \in \mathbb{R}^{m\times n}, \\ H_t &= \begin{bmatrix} a_1 - c & \cdots & a_n - c \end{bmatrix} = A - ce^T \in \mathbb{R}^{m\times n}. \end{aligned}$$

The scatter matrices S_b, S_w and S_t can be expressed as

$$S_b = H_b H_b^T, \quad S_w = H_w H_w^T, \quad S_t = H_t H_t^T.$$

Since

$$\text{Trace}(S_b) = \sum_{i=1}^{k} \sum_{j \in N_i} \|c^{(i)} - c\|_2^2,$$

and

$$\text{Trace}(S_w) = \sum_{i=1}^{k} \sum_{j \in N_i} \|a_j - c^{(i)}\|_2^2,$$

it is clear that $\text{Trace}(S_b)$ measures the distance between clusters while $\text{Trace}(S_w)$ measures the closeness of the columns within the clusters over all k clusters. Note that when the between-cluster relationship is remote, i.e., the centroids of the clusters are remote, $\text{Trace}(S_b)$ will have a large value, but, when within each cluster are located tightly around their own cluster centroid, $\text{Trace}(S_w)$ will have a small value. Hence, the cluster quality can be measured using $\text{Trace}(S_b)$ and $\text{Trace}(S_w)$.

In the lower dimensional space mapped from the linear transformation $G^T \in \mathbb{R}^{l \times m}$, the between-scatter, within-scatter 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^T would maximize $\text{Trace}(S_b^L)$ and minimize $\text{Trace}(S_w^L)$ simultaneously, equivalently, maximize $\text{Trace}(S_b^L)$ and minimize $\text{Trace}(S_t^L)$ simultaneously, which results that the commonly used optimization in classical LDA for determining the optimal linear transformation G^T is

$$G = \arg\max_G \{\text{Trace}((S_t^L)^{-1} S_b^L)\}. \tag{2.1}$$

In the classical LDA [33], the optimization problem above is solved by computing all the eigen-pairs

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

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

The classical LDA does not work when S_t is singular, which is the case for undersampled data. To deal with the singularity of S_t, several generalized optimization criterions

for determining the transformation G have been proposed. These extensions include

$$G = \arg\max_{G}\{\text{Trace}((S_t^L)^{(+)}S_b^L)\} \tag{2.2}$$

$$G = \arg\min_{G}\{\text{Trace}((S_b^L)^{(+)}S_w^L)\}, \qquad rank(G^T H_b) = rank(H_b), \tag{2.3}$$

and

$$\arg \max_{G^T S_w G=0,\ G^T G=I} \text{Trace}(S_b^L). \tag{2.4}$$

They are generalizations of the optimization problem (2.1) in the sense that they are equivalent to the problem (2.1) if S_t is nonsingular. The optimization problems (2.2), (2.3) and (2.4) are the optimization criterions for establishing the OLDA in [8], LDA/QR-GSVD [2],[10],[11],[16] and NLDA in [20],[26], respectively. One remark is that these objective functions do not have a unique solution. One can always add vector that are inside the common nullspace of H_b and H_w and the objective value will still be preserved. In [8] and [2],[10],[11],[16], the optimal transformation G is obtained by computing an eigen-decomposition associated with the generalized eigenvalue problems

$$S_b x = \lambda S_t x, \quad \lambda \neq 0, \tag{2.5}$$

and

$$S_w x = \lambda S_b x, \quad \lambda \neq 0, \tag{2.6}$$

through the simultaneous diagonalization of the scatter matrices S_b and S_t and the QR and GSVD [37] of the pair (H_b, H_w), respectively. However, any eigen-decomposition including the simultaneous diagonalization of the scatter matrices S_b and S_t for the generalized eigenvalue problem (2.5) and the GSVD of the pair (H_b, H_w) for the generalized eigenvalue problem (2.6) is very expensive and may not be numerically stable if the some matrix inversions are involved and the inversed matrices are highly ill-conditioned [29]. Motivated by these observations, we will show that a column orthogonal solution of both

optimization problems (2.2) and (2.3) can be computed easily by using only orthogonal transformations without involving any eigen-decomposition and matrix inverse. As a direct result, a fast and stable orthogonal LDA algorithm is developed in the next section.

Numerous schemes have been proposed in the past to handle the problem of dimensionality reduction. These methods include Principal Component Analysis (PCA) [21] and Linear Discriminant Analysis (LDA) [33]. When the problem involves classification and the underlying distribution of the data follow normal distributions, LDA has been known to be one of the most optimal dimensionality reduction methods , for it attempts to seek an optimal linear transformation by which the original data in high-dimensional space is transformed to a much lower dimensional space, preferably the reduced dimension is as small as possible and yet retaining as much information as possible. This algorithm is more efficient compared to PCA for it makes use of class information while the latter doesn't. Relative to the famous Support Vector Machine, LDA take advantage of the normal distribution assumption and makes the classification more accurate. A good data structure would be one that has data of the same class being close to each other and data from different classes are far away from each other. To balance these two goals, LDA achieves maximum class discrimination by minimizing the within-class distance and maximizing the between-class distance simultaneously via classical Fisher Criteria which we will discuss in the next section later.

LDA has been applied successfully for decades in many important applications including pattern recognition [4],[23],[33], information retrieval [28],[32], face recognition [24],[30], micro-array data analysis [18],[19], and text classification [35] of which we have also discussed in the previous section. It is usually formulated as an optimization problem and involved linear transformation which is classically computed by applying eigen-decompositions on scatter matrices. As a result, a main disadvantage of LDA is that the so-called "total scatter matrix" must be nonsingular. However, in many applications such as those mentioned above, all scatter matrices can be singular since 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. This is known as the undersampled problem [33] and it is also commonly called small sampled size problem. As a result, we

cannot apply the classical LDA because of the singularity of the scatter matrices caused by high dimensionality and we need some generalizations to overcome this technical issue.

2.1 Generalized LDA

In order to overcome the singularity problem and make the LDA applicable in a wider range of applications, many extensions of the classical LDA have been proposed in literature. According to [11], such extensions can be categorized into three groups. The first approach, known as two-stage LDA [9],[10],[30], is to apply an intermediate dimensionality reduction stage to reduce the data dimensionality by applying the classical LDA [33]. Although this approach is simple, the intermediate dimensionality reduction stage may remove some important information, furthermore, there are some technicalities issue such as to what extent should we reduce the dimension before we apply our LDA algorithm. The second approach is to do a regularization by adding a perturbation term to the scatter matrix, usually the perturbation term is a positive multiples of identity matrix, the resulting algorithm is known as the regularized LDA (RLDA) [15],[34]. The main disadvantage of RLDA is that the optimal amount of the perturbation to be used is difficult to determine [11],[15], since if perturbation is large then we lose information on the scatter matrix, while if it is very small the regularization may not be sufficiently effective. Usually cross-validation is used in this implementation and it might be very costly. The third approach applies the pseudoinverse [29] to avoid the singularity problem. The methods based on this approach include Null space LDA (NLDA) [20],[26], Uncorrelated LDA (ULDA) [8], Orthogonal LDA (OLDA) [8], QR/GSVD-based LDA (LDA/QR-GSVD) [2],[10],[11],[16]. It seems that the pseudoinverse-based methods are different when dealing with the singularity problem, in fact, they are closely related, for instance, it has been shown in [5] that NLDA and OLDA are equivalent under a mild condition which holds in many applications involving high-dimensional data. It is now well-realized that the pseudoinverse-based methods achieve comparable performance with two-stage LDA and RLDA. Note that both two-stage LDA and RLDA involve the estimation of some parameters which can be very expensive, but, the pseudoinverse-

based methods do not involve any parameter and hence might be attractive. There is a commonality for many generalized LDA algorithms, that is, they compute the optimal linear transformations by some eigen-decompositions and involve some matrix inversions. However, the eigen-decomposition is computationally expensive [29], at least much more expensive than a QR decomposition especially when the data size is very large, and the involvement of matrix inverses may lead to that the methods being not numerically stable if the related matrices are ill-conditioned [29]. Hence, many LDA algorithms may have high computational cost and potential numerical instability problems. This problem also occurs in the computation of null space based LDA as well in which inverse, eigenvalue decomposition or singular value decomposition are being computed.

There are many other variants of LDA to handle different types of problems, for example, semi-supervised LDA was proposed so that when incomplete information about classes are given to us, we can still classify them and reduce the dimension, trying our best with the information given [52]. Furthermore, the assumption of linearity is very demanding as in real world, we clearly observe that our world is highly non-linear. A good thing is that it has been shown that most data can be handled by using kernel that satisfy Mercer's condition, and transform it to a linear problem to handle, of course the challenge is to find a nice kernel, and there seems to be a need to study the optimal parameter of the kernel [53]. Other application such as when the data is real time in nature, i.e. the data are created from time to time, we should also enable an algorithm that enable updating. An algorithm was proposed by Prof Li Qi et al earlier on [39], however the approximation is quite crude in the sense that no error bound is given and there seems to involve a few heuristic approaches. Another variant of LDA include the fact that some data comes in 2 dimensions in nature, for example pictures. Vectorization would increase the computational cost significantly; hence it would be ideal if we can handle the problem without vectorization. An algorithm called 2DLDA was proposed and it is shown that there is a good improvement in terms of speed [54] though again, heuristic approaches are taken, however this motivated a lot more variants of 2DLDA algorithms as the time saved is really significant; a potential extension would be to come out with a method to handle video. It seems that this area is full of potential and it is interesting, relating many real life situations to mathematics.

It would not be possible to summarize the whole development of LDA in a single thesis as the field has invited the attention of various computer scientists, engineers and statisticians to work on them for centuries and various heuristic approaches have been developed over the time. It is about time for Mathematicians to join in this area to justify and check on these heuristic approaches and explore new implementations of them, for instance, up to date, there is no sparse implementation of LDA being developed yet.

2.2 Alternative Representation of Scatter Matrices

It is well known that computing SVD of the scatter matrices are highly expensive if we do not adopt any trick. For in stance, knowing that $S_t = H_t H_t^T$, rather than computing SVD of an m by m matrix, we can actually just compute $H_t^T H_t$ to cut down the cost and by doing that we can cut down the cost from $\mathbf{O}(m^3)$ to $\mathbf{O}(n^3)$ when $m >> n$. But at the same time, they transfer the cost to matrix multiplication which can take up to $\mathbf{O}(mn^2)$, however, overall, we are still saving up some costs significantly.

We will propose another method to cut down the cost further.

Lemma 2.2.1. *Let*

$$E = \frac{1}{n} e e^T,\ E_i = \frac{1}{n_i} e_i e_i^T,\ i = 1 \ldots, k$$

then

$$\begin{cases} S_b = A(\begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_k \end{bmatrix} - E)A^T \\ S_w = A(I - \begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_k \end{bmatrix})A^T \\ S_t = A(I - E)A^T \end{cases}$$

Proof. First we have

$$
\begin{aligned}
S_b &= \sum_{i=1}^{k} n_i (c^{(i)} - c)(c^{(i)} - c)^T \\
&= \sum_{i=1}^{k} n_i (\frac{1}{n_i} A_i e_i - \frac{1}{n} Ae)(\frac{1}{n_i} A_i e_i - \frac{1}{n} Ae)^T \\
&= \sum_{i=1}^{k} [\frac{1}{n_i} A_i e_i e_i^T A_i^T - \frac{1}{n} A_i e_i e^T A^T - \frac{1}{n} Ae e_i^T A_i^T + \frac{n_i}{n^2} Aee^T A^T] \\
&= \sum_{i=1}^{k} A_i E_i A_i^T - \frac{1}{n} (\sum_{i=1}^{k} A_i e_i) e^T A^T - \frac{1}{n} Ae (\sum_{i=1}^{k} e_i^T A_i^T) + \frac{\sum_{i=1}^{k} n_i}{n} AEA^T \\
&= A \begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_k \end{bmatrix} A^T - \frac{1}{n}(Ae)e^T A^T - \frac{1}{n} Ae(e^T A^T) - \frac{1}{n} Ae(e^T A^T) + \frac{n}{n} AEA^T \\
&= A \begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_k \end{bmatrix} A^T - AEA^T - AEA^T + AEA^T \\
&= A(\begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_k \end{bmatrix} - E) A^T
\end{aligned}
$$

Similarly, we can also prove that

$$
\begin{aligned}
S_t &= \sum_{i=1}^{n} (a_i - c)(a_i - c)^T \\
&= \sum_{i=1}^{n} (a_i - \frac{1}{n} Ae)(a_i^T - \frac{1}{n} e^T A^T) \\
&= \sum_{i=1}^{n} a_i a_i^T - \frac{1}{n} (\sum_{i=1}^{n} a_i) e^T A^T - \frac{1}{n} Ae \sum_{i=1}^{n} a_i^T + \sum_{i=1}^{n} \frac{1}{n^2} Aee^T A^T \\
&= AA^T - \frac{1}{n}(Ae)e^T A^T - \frac{1}{n} Ae(e^T A^T) + \frac{1}{n} Aee^T A^T \\
&= AA^T - \frac{1}{n} Aee^T A^T \\
&= A(I - E)A^T
\end{aligned}
$$

Last but not least, the last equality is trivial, we know that $S_t = S_b + S_w$, so

$$S_w = S_t - S_b = A(I - \begin{bmatrix} E_1 & & \\ & \ddots & \\ & & E_k \end{bmatrix})A^T$$

and the proof is now complete. □

We will now develop an alternative representation of the factor of scatter matrices which will cut down the cost when the number of classes is big by using Householder transformation.

Let P be the permutation matrix which is obtained by exchanging the i-th column and the $(\sum_{j=1}^{i-1} n_j + 1)$-th column of the $n \times n$ identity matrix, $i = 2, \ldots, k$. Denote

$$W_i = I - \left(\frac{1}{\sqrt{n_i - \sqrt{n_i}}} \begin{bmatrix} 1 - \sqrt{n_i} \\ 1 \\ \vdots \\ 1 \end{bmatrix} \right) \left(\frac{1}{\sqrt{n_i - \sqrt{n_i}}} \begin{bmatrix} 1 - \sqrt{n_i} \\ 1 \\ \vdots \\ 1 \end{bmatrix} \right)^T, i = 1, \ldots, k,$$

and

$$W = I - \left(\frac{1}{\sqrt{n - \sqrt{n n_1}}} \begin{bmatrix} \sqrt{n_1} - \sqrt{n} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} \right) \left(\frac{1}{\sqrt{n - \sqrt{n n_1}}} \begin{bmatrix} \sqrt{n_1} - \sqrt{n} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix} \right)^T, i = 1, \ldots, k,$$

The matrices W and $W_i (i = 1, \ldots, k)$ are nothing but the Householder transfoma-tions of vectors $\begin{bmatrix} \sqrt{n_1} \\ \sqrt{n_2} \\ \vdots \\ \sqrt{n_k} \end{bmatrix}$ and $e_i, (i = 1, \ldots, k)$ respectively, they are all orthogonal and

as Householder transformation, they satisfy the properties:

$$W(\frac{1}{\sqrt{n}}\begin{bmatrix}\sqrt{n_1}\\ \sqrt{n_2}\\ \vdots\\ \sqrt{n_k}\end{bmatrix}) = \begin{bmatrix}1\\ 0\\ \vdots\\ 0\end{bmatrix}, W_i(\frac{1}{n_i}e_i) = \begin{bmatrix}1\\ 0\\ \vdots\\ 0\end{bmatrix}, i = 1, \ldots, k.$$

Lemma 2.2.2. *Denote*

$$[A_1, A_2, A_3] = A\begin{bmatrix}W_1 & & \\ & \ddots & \\ & & W_k\end{bmatrix}P\begin{bmatrix}W & \\ & I\end{bmatrix}, A_2 \in \mathbb{R}^{m\times(k-1)}, A_3 \in \mathbb{R}^{m\times(n-k)}$$

Then we have

$$S_b = A_2A_2^T, S_w = A_3A_3^T, S_t = [A_2, A_3][A_2, A_3]^T$$

Proof. we can prove the theorem by direct verification.

□

The computation of A_2 and A_3 only requires $\mathbf{O}(mn)$ flops, the complexity is almost the same with computation of H_w and H_t. However, $H_w \in \mathbb{R}^{m\times n}$ but $A_3 \in \mathbb{R}^{m\times(n-k)}$, thus the structure of S_w and S_t can be cut down and the impact will be great when the number of classes are big. Furthermore, as a by product of the lemma, it is clear that S_w is nonsingular only if $m \leq n - k$, this fact seems to be not revealed in earlier literatures as it is commonly addressed that S_w is non-singular only if $m \leq n$, hence we have obtained an even sharper bound.

The significance of the result is that we can now replace H_b with A_2 and H_w with A_3 and obtain a faster implementation and the effect would be clear when the number of classes is big.

Chapter 3

Orthogonal LDA

3.1 A Review of Orthogonal LDA

As mentioned earlier, orthogonal LDA is a type of LDA implementation of which we insist that the projection axes are orthogonal to each other. Namely, Let G be the projection matrix, then we insist that $G^T G = I$.

The objective function considered in earlier literatures can be written in this form:

$$\begin{cases} G = argmin_G \left\{ Trace((S_t^L)^{(+)} S_w^L) \right\} \\ \qquad rank(G^T H_b) = rank(H_b) \end{cases} \tag{3.1}$$

Notice that the rank constraint imposed, served to prevent lost of information, in particular, when this constraint is imposed, we are avoiding the case of $G = 0$, of which if it is chosen, we are going to lose all information. Common way to attain the orthogonal condition is simple, after obtaining some projection direction; we can simply either perform SVD decomposition or perform Gram Schmidt process or QR decomposition to obtain orthogonal directions. The rank preserving condition is more meaningful as it carries the physical meaning of preserving class separation condition.

Two famous Orthogonal LDA algorithms have been proposed in the past, namely OLDA which was proposed by Prof Ye Jieping et al as well as Foley Sammon LDA (FSLDA)[42]. FSLDA algorithm is very expensive for large and high dimensional data. FSLDA algorithm is very expensive as some matrix inversions are involved, which may

cause numerical instability problems, the two methods are derived from different perspective. Compared to FSLDA, the OLDA algorithm provides a simpler and more efficient way for computing orthogonal transformation of LDA.

The whole idea of OLDA is simply to perform GSVD first, followed by an orthogonlization process which can be obtained by an economic QR decomposition. Hence the process, strictly speaking is even more expensive than a regular GSVD as it requires the orthogonalization process. The algorithm is given as Algorithm 2 in the **numerical experiment** subsection later.

3.2 A New and Fast Orthogonal LDA

Recall from the review in the earlier chapter, one way to overcome the singularity issue is to perform a preprocessing step such as PCA or QR decomposition. We would prefer using economic QR as a preprocessing step as it is cheaper than computation of SVD and it really cuts down the dimension significantly, in particular, rather than dealing with matrices of size $m \times n$, we will be working with matrices of size $n \times n$ which is much smaller in size especially when we deal with other more expensive operation besides matrix multiplication. Hence there will be a significant saving.

Next we will develop some tools to cut down the cost of performing OLDA.

First of all, we shall define trace of a pair of matrix as follows:

For any matrix pair (A, B), the nonzero finite generalized eigenvalues of the eigenvalue problem

$$Bx = \lambda Ax, \lambda \neq 0$$

be $\lambda_1, \ldots, \lambda_p$, then we define

$$Trace_{eig}(A, B) = \sum_{i=1}^{p} \lambda_i$$

We have the following results which can be found in [8] and [11]

Lemma 3.2.1.

$$trace(S_t^{(+)} S_b) = max_G Trace((S_t^L)^{(+)} S_b^L)$$

$$Trace_{eig}(S_b, S_w) = min_{G \ s.t. \ rank(G^T H_b) = rank(H_b)} Trace((S_b^L)^{(+)} S_w^L)$$

By using this lemma, we will be able to obtain the following theorem which will enable us to have a fast algorithm.

Theorem 3.2.2. *Let the economic QR decomposition of* $[A_2, A_3]$ *be*

$$[A_2, A_3] = Q_1 \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \\ 0 & 0 \end{bmatrix}, \ R_{1,1} \in \mathbb{R}^{q \times (k-1)}, \ R_{2,2} \in \mathbb{R}^{\gamma - q \times (n-k)}, \ Q_1 \in \mathbb{R}^{m \times (n-1)}$$

where Q_1 *is column orthogonal,* $R_{1,1}$ *and* $R_{2,2}$ *are of full row rank,*

$$q = rank(A_2), \ \gamma = rank\,[A_2 \ A_3] - rank(A_2)$$

Next, let the QR decomposition of $\begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} R_{2,2}^T$ *be*

$$\begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} R_{2,2}^T = Q_2 \begin{bmatrix} \Pi \\ 0 \end{bmatrix} \ \Pi \in \mathbb{R}^{\gamma \times \gamma}, \ Q_2 \in \mathbb{R}^{(q+\gamma) \times (q+\gamma)}$$

where Q_2 *is orthogonal. Define*

$$G = Q_1(:, 1 : q + \gamma) Q_2(:, \gamma + 1 : \gamma + q) \in \mathbb{R}^{m \times q}$$

Then G satisfies

1. $G^T G = I$

2. *G solves the optimization problems (2.2) and (2.3)*

Proof. First, it is clear that both Q_1 and Q_2 are column orthogonal, so, it is obvious that $G^T G = I$.

Next, since $R_{2,2}$ is of full row rank, there exists an orthogonal matrix V such that

$$R_{2,2} = [R_{2,2}\ 0]\, V,\ R_{2,2} \in \mathbb{R}^{\gamma\times\gamma},\ rank(R_{2,2}) = \gamma$$

Denote

$$R_{1,2}V^T = [R_{1,2}\ R_{1,3}],\ R_{1,2} \in \mathbb{R}^{q\times\gamma}$$

Let $\left[Q_1\ \widetilde{Q_1}\right]$ and $\begin{bmatrix} V_1 \\ \widetilde{V_1} \end{bmatrix}$ be orthogonal. We have

$$\left[Q_1\ \widetilde{Q_1}\right]^T A_2 = \begin{bmatrix} R_{1,1} \\ 0 \\ 0 \end{bmatrix}, \left[Q_1\ \widetilde{Q_1}\right]^T A_3 V^T = \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \\ 0 & 0 \end{bmatrix}$$

Hence

$$\left[Q_1\ \widetilde{Q}_1\right]^T S_b \left[Q_1\widetilde{Q}_1\right] = (\left[Q_1\ \widetilde{Q}_1\right]^T A_2)(A_2^T \left[Q_1\widetilde{Q}_1\right]) = \begin{bmatrix} R_{1,1}R_{1,1}^T & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$\begin{aligned}\left[Q_1\ \widetilde{Q}_1\right]^T S_w \left[Q_1\widetilde{Q}_1\right] &= (\left[Q_1\ \widetilde{Q}_1\right]^T A_3 V^T)(V A_3^T \left[Q_1\widetilde{Q}_1\right]) \\ &= \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \\ 0 & 0 \end{bmatrix}^T\end{aligned}$$

and

$$\begin{aligned}
&\left[Q_1\ \widetilde{Q}_1\right]^T S_t \left[Q_1\ \widetilde{Q}_1\right] \\
&= (\left[Q_1\ \widetilde{Q}_1\right]^T [A_2\ A_3] \begin{bmatrix} I & 0 \\ 0 & V^T \end{bmatrix})(\begin{bmatrix} I & 0 \\ 0 & V \end{bmatrix} [A_2\ A_3]^T \left[Q_1\ \widetilde{Q}_1\right]) \\
&= \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & 0 \\ 0 & 0 & 0 \end{bmatrix}^T
\end{aligned}$$

Note that $\begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & 0 \end{bmatrix}$ is of full rank and $R_{2,2}$ is nonsingular, we obtain

$$\begin{aligned}
Trace(S_t^{(+)} S_b) &= Trace\left[\left(\begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & 0 \end{bmatrix} \begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & 0 \end{bmatrix}^T\right)^{-1} \begin{bmatrix} R_{1,1}R_{1,1}^T & 0 \\ 0 & 0 \end{bmatrix}\right] \\
&= Trace\left(([R_{1,1}\ R_{1,3}]\,[R_{1,1}\ R_{1,3}]^T)^{-1}(R_{1,1}R_{1,1}^T)\right) \qquad (3.2)
\end{aligned}$$

by the matrix inversion formula and

$$Trace_{eig}(S_b, S_w) = Trace((R_{1,1}R_{1,1}^T)^{-1} R_{1,3}R_{1,3}^T) \qquad (3.3)$$

We partition Q_2 as follows:

$$Q_2 = \begin{bmatrix} Q_{1,1} & Q_{1,2} \\ Q_{2,1} & Q_{2,2} \end{bmatrix},\ Q_{2,1} \in \mathbb{R}^{\gamma\times\gamma},\ Q_{1,2} \in \mathbb{R}^{q\times q} \qquad (3.4)$$

Since $\begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} = Q_2 \begin{bmatrix} \Pi \\ 0 \end{bmatrix}$ is equivalent to

$$Q_2^T \begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} V^T V R_{2,2}^T = Q_2^T \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix} \begin{bmatrix} R_{2,2}^T \\ 0 \end{bmatrix} = Q_2^T \begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} R_{2,2}^T = \begin{bmatrix} \Pi \\ 0 \end{bmatrix}$$

which is basically equivalent to

$$\begin{bmatrix} Q_{1,2}^T & Q_{2,2}^T \\ Q_{1,1}^T & Q_{2,1}^T \end{bmatrix} \begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} = \begin{bmatrix} 0 \\ \Pi R_{2,2}^{-T} \end{bmatrix}$$

Hence $Q_{1,2}$ is also non-singular because $R_{2,2}$ is nonsingular, so now we have

$$\begin{bmatrix} Q_{1,2}^T & Q_{2,2}^T & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \left[Q_1 \ \widetilde{Q}_1\right]^T A_2 = \begin{bmatrix} Q_{1,2}^T R_{1,1} \\ 0 \\ 0 \end{bmatrix} \tag{3.5}$$

and

$$\begin{bmatrix} Q_{1,2}^T & Q_{2,2}^T & 0 \\ 0 & I & \\ 0 & 0 & I \end{bmatrix} \left[Q_1 \widetilde{Q}_1\right]^T A_3 V^T = \begin{bmatrix} 0 & Q_{1,2}^T R_{1,3} \\ R_{3,3} & 0 \\ 0 & 0 \end{bmatrix} \tag{3.6}$$

which together with $G = Q_1(:, 1:q+\gamma)Q_2(:, \gamma+1:\gamma+q) = Q_1(:, 1:q+\gamma)\begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix}$ give us

$$G^T A_2 = Q_{1,2}^T R_{1,1},\ G^T A_3 V^T = \left[0 \ Q_{1,2}^T R_{1,3}\right],$$

$$\begin{aligned} S_b^L &= (G^T A_2)(G^T A_2)^T = Q_{1,2}^T (R_{1,1} R_{1,1}^T) Q_{1,2}, \\ S_w^L &= (G^T A_3 V^T)(G^T A_3 V^T)^T = Q_{1,2}^T (R_{1,3} R_{1,3}^T) Q_{1,2} \\ S_t^L &= (G^T [A_2 \ A_3] \begin{bmatrix} I & 0 \\ 0 & V^T \end{bmatrix})(G^T [A_2 \ A_3] \begin{bmatrix} I & 0 \\ 0 & V^T \end{bmatrix})^T \\ &= Q_{1,2}^T ([R_{1,1} \ R_{1,3}] [R_{1,1} \ R_{1,3}]^T) Q_{1,2} \end{aligned} \tag{3.7}$$

which together with (3.2) and (3.3), we have

$$\begin{aligned} Trace((S_t^L)^{(+)}S_b^L) &= Trace((Q_{1,2}^T[R_{1,1}\ R_{1,3}][R_{1,1}\ R_{1,3}]^T Q_{1,2})^{-1}(Q_{1,2}^T R_{1,1}R_{1,1}^T Q_{1,2})) \\ &= Trace\left(([R_{1,1}\ R_{1,3}]\,[R_{1,1}\ R_{1,3}])^{-1}(R_{1,1}R_{1,1}^T)\right) \\ &= Trace(S_t^{(+)}S_b) \end{aligned}$$

$$\begin{aligned} Trace((S_b^L)^{(+)}S_w^L) &= Trace((Q_{1,2}^T R_{1,1}R_{1,1}^T Q_{1,2})^{(-1)}(Q_{1,2}^T R_{1,3}Q_{1,2})) \\ &= Trace((R_{1,1}R_{1,1}^T)^{-1}(R_{1,3}R_{1,3})) \\ &= Trace(S_t^{(+)}S_b) \end{aligned}$$

and

$$\begin{aligned} Trace((S_b^L)^{(+)}S_w^L) &= Trace((Q_{1,2}^T R_{1,1}R_{1,1}^T Q_{1,2})^{-1}(Q_{1,2}^T R_{1,3}R_{1,3}^T)Q_{1,2}) \\ &= Trace((R_{1,1}R_{1,1}^T)(R_{1,3}R_{1,3^T})) \\ &= Trace_{eig}(S_b, S_w) \end{aligned}$$

Hence now it remains to check that the transformation preserves the rank.

$$\begin{aligned} rank(G^T H_b) &= rank(G^T H_b H_b^T G) = rank(G^T S_b G) = rank(G^T A_2 A_2^T G) = rank(G^T A_2) \\ &= rank(Q_{1,2}^T R_{1,1}) = rank(R_{1,1}R_{1,1}^T) = rank(A_2 A_2^T) \\ &= rank(S_b) = rank(H_b H_b^T) = rank(H_b). \end{aligned}$$

Therefore we conclude using Lemma 3.2.1 that G solves both the optimization problems (2.2) and (2.3). □

Theorem 3.2.2 leads us to the following new orthogonal LDA, called OLDA/new for short:

Notice that we perform QR decomposition on the data matrix at the very first step,

Algorithm 1: OLDA/new

Input: data matrix A, cluster number k, $[m, n] = size(A)$
and Index vector Ind=$[n_1 \ \sum_{i=1}^{2} n_i \ \ldots, \sum_{i=1}^{k} n_i]^T$
Output: Transformation matrix G

1. Perform an economic QR decomposition on data matrix A, $[\widetilde{Q}, A] = qr(A, 0)$
2. Set $n1 = Ind(1)$, $w(1,1) = \sqrt{n1}$,
$v = \left[1 - \sqrt{n1} \ 1 \ \ldots \ 1\right]^T / \sqrt{n1 - \sqrt{n1}} \in \mathbb{R}^{n1 \times 1}$,
Compute
$A(:, 1:n1) = A(:, 1:n1) - (A(:, 1:n1) * v) * v^T$;
For $i = 2:k$
set $ni = Ind(i) - Ind(i-1)$,, $w(i,1) = \sqrt{ni}$,
$v = \left[1 - \sqrt{ni} \ 1 \ \ldots, \ 1\right]^T / \sqrt{ni - \sqrt{ni}} \in \mathbb{R}^{ni \times 1}$,
compute
$A(:, Ind(i-1)+1 : Ind(i))$
$= A(:, Ind(i-1)+1 : Ind(i)) - (A(:, Ind(i-1)+1 : Ind(i)) * v) * v^T$
$v = A(:, i)$, $A(:, i) = A(:, Ind(i-1)+1)$, $A(:, Ind(i-1)+1) = v$;
3. Set $w(1,1) = w(1,1) - \sqrt{n}$, $w = w/\sqrt{n - \sqrt{n * n1}}$,
Compute $A(:, 1:k) = A(:, 1:k) - (A(:, 1:k) * w) * w^T$;
4. Compute $[Q1, R] = qr(A(:, 2:n), 0)$, $A(1:n-1, 2:n) = R$
$q = rank(A(1:k-1, 2:k))$, $\gamma = rank(A(q+1:n-1, k+1:n))$;
5. $[Q2, R] = qr(A(1:q+\gamma, k+1:n) * A(q+1:q+\gamma, k+1:n)')$
6. Compute $G = \widetilde{Q} * Q1(:, 1:q+\gamma) Q2(:, \gamma+1, \gamma+q)$.

that is to cut down the cost effectively, and we also multiply $\widetilde{Q}$ to form the G at the very last step to match things back. This is due to the rationale that frequently, in practical world, the number of features is going to be much greater than the number of sample points, namely, $m >> n$. Such examples include gene expression data, text document data as well as facial recognition data. By performing a QR decomposition, we will cut down subsequent computation of data matrix including something of magnitude $\mathbf{O}(mn^2)$ to $\mathbf{O}(n^3)$ which is so much cheaper.

Algorithm 1 for OLDA/new is implemented by some QR factorizations without computing any eigen-decomposition and matrix inversion. Hence it is inverse-free and numerically stable. Moreover, its cost is about the total cost of a QR factorization for the data matrix A and a QR factorization of a $(q+\gamma) \times (n-k)$ matrix. Note that $q + \gamma = rank\,[A_2 \ A_3] = rank(S_t) \leq n - 1$, therefore, Algorithm 1 is very fast.

3.3 Numerical Experiment

Based on the optimization problem (2.3), a a generalized LDA, called orthogonal LDA (OLDA) was derived by Prof Ye Jieping et al in [42]. The feature that distinguishes the method from other algorithms is that it insists that the transformation matrix G must be column orthogonal, it was computed through the simultaneous diagonalization of the scatter matrices S_b and S_t and thus the singularity difficulty is overcome implicitly. OLDA algorithm that was proposed consists of three main steps:

1. The null space of the total scatter matrix S_t is removed
2. Classical uncorrelated LDA (ULDA) was performed. (This is basically step 3 to step 4 of the algorithm).
3. Apply an orthogonalization step to the transformation

Remark: ULDA is a version of LDA whereby the features produced are uncorrelated from each other.[25]

The pseudocode of the method is given in Algorithm 2 as such:

Algorithm 2: OLDA

Input: data matrix A, cluster number k, $[m, n] = size(A)$
and Index vector Ind=$[n_1 \ \sum_{i=1}^{2} n_i \ \ldots, \sum_{i=1}^{k} n_i]^T$
Output: Transformation matrix G

1. Form matrices H_b, H_w, H_t;
2. Compute the reduced SVD of H_t as $H_t = U_1 \Sigma_t V_1^T$;
3. Compute the SVD of $\Sigma_t^{-1} U_1^T H_b$ as $\Sigma_t^{-1} U_1^T H_b = P \Sigma Q^T$ and let $q = rank(\Sigma)$;
4. $X := U_1 \Sigma_t^{-1} P$;
5. Compute the economic QR factorization of $X(:, 1:q)$ as $X(:, 1:q) = \widetilde{Q}\widetilde{R}$;
6. $G := \widetilde{Q}$.

We can compare the algorithms and provide the following comments:

- Algorithm 1 is inverse-free and numerically stable
- $\Sigma_t^{-1} U_1^T H_b$ and $U_1 \Sigma_t^{-1} P$ are involved in Algorithm 2, if Σ_t is highly ill-conditioned, Algorithm 2 may have numerical instability problem;

Computational Complexity of Algorithm 1 (OLDA/new)
Steps 1 : $4mn^2 - 2n^3$,
Step 2 : $\mathbf{O}(n^2)$,
Step 3 : $2n(n-1)^2 - \frac{4}{3}(n-1)^3 + 4n(n-1)\gamma - 4\gamma^2 n + \frac{4}{3}\gamma^3$,
Step 4 : $2\gamma(n-k)(\gamma-q) + 4\gamma^2(\gamma-q) + \frac{2}{3}(\gamma-q)^3 - 2\gamma(\gamma-q)^2$,
Step 5 : $\mathbf{O}(m\gamma q)$.
Computational Complexity of Algorithm 2 (OLDA)
Step 1 : $\mathbf{O}(mn)$,
Step 2 : $14mn^2 - 2n^3$,
Step 3 : $2m\gamma k + 14\gamma k^2 - 2k^3$,
Step 4 : $2m\gamma^2$,
Step 5 : $4mq^2 - 2q^3$.

Since $q \leq k-1, \quad \gamma \leq n-1$, it can be seen that the computational complexity of Algorithm 1 is much lower than that of Algorithm 2, and is also lower than that of Algorithm 3 when m and n are huge, k is large and m is much larger than n.

As we can see from above that computational complexity of Algorithm 1 is much lower than that of Algorithm 2, In this section we demonstrated the efficiency of our OLDA/new algorithm by comparing the classification accuracy with Algorithm 2 for OLDA. The K-Nearest-Neighbor (K-NN) algorithm with different K is used as the accuracy classifier for Algorithms 1 and Algorithm 2.

In our numerical experiments, various high-dimensional data sets from various data sources are used, such data sets include text documents, face images and gene expression data. Our data sets are described as follows:

Leukemia, **Lymphoma** and **Srbct** are gene expression data. **Leukemia** is a data set coming from a study of gene expression (http://www.genome.wi.mit.edu/MPR) in two types of acute leukemia, acute lymphoblastic leukemia(ALL) and acute myeloblas-

tic leukemia (AML) The sample size is 72 while the dimension is 7129. **Lymphoma** is a data set of the three most prevalent adult lymphoid malignancies (http://genome-www.stanford.edu/lymphoma). The sample size is 62 while the dimension is 4026. **Srbct** is the data set of small, round tumors of childhood cancer (http://research.nhgri.nih.gov/microarray/Supplement). This data consists of 83 samples spanning four classes (excluding 5 samples as done in Ye et al: Using Uncorrelated Discriminant Analysis for Tissue Classification with Gene Expression Data), the dimension of the data is 2308.

Some data sets are related to facial recognition problem. The following two samples are from
http://www.cs.uiuc.edu/homes/dengcai2/Data/FaceData.html: **Yale** contains 165 grayscale images in GIF format of 15 individuals, and **ORL** contains ten different images of each of 40 distinct subjects. The subscripts in the table of experiment output denotes the number of pixels involved.

Yale B data set has 38 individuals and around 64 near frontal images under different illuminations per individual [58].

We also studied some text document data **Tr41**, **K1a**, **K1b**, **wap**,**cranmed**, **classic**, **review** and **sports** from CLUTO
(http:/glaros.dtc.umn.edu/gkhome/cluto/cluto/download).

3.4 Simulation Output

The following table summarizes the data structures in our experiments.

Data	m	n	k	Test data
Leukemia	3571	58	2	14
Lymphoma	4026	51	3	11
Srbct	2308	52	4	11
Tr41	7454	707	10	134
wap	8460	1254	20	306
$\text{Yale}_{32\times32}$	1024	135	15	30
$\text{Yale}_{64\times64}$	4096	135	15	30
Yale B	32256	1959	38	465

Data	m	n	k	Test data
$\text{ORL}_{32\times32}$	1024	320	40	80
$\text{ORL}_{64\times64}$	4096	320	40	80
classic	41618	5677	4	1417
sports	126373	6866	7	1714
Kla	21839	1882	20	458
Klb	21839	1875	6	465
cranmed	41618	1946	2	485
review	126373	3258	5	811

Table 3.1 : Data dimension (m), sample size(n), number of clusters (k) and the number of test data

For For the data, we performed our study by repeated random splitting into training and test sets ourselves. The following algorithm is used: within each class, we randomly reorder the data and then for each class size n_i, we compute $\lceil 0.8n_i \rceil$ whereby $\lceil . \rceil$ is the ceiling function. The splitting was repeated 10 times and the resulting average accuracies of different algorithms are summarized thereafter.

Recall from earlier discussion that without performing dimension reduction, usually the accuracy is going to be lower due to the fact that data might contain some noise. This observation has been reported for example in [59]. K-NN accuracies are reported in Table 3.2 below. For the K-NN algorithm, $K = 1, 3, 5$ are used for all data sets.

	OLDA/new			OLDA		
data	$1-NN$	$3-NN$	$5-NN$	$1-NN$	$3-NN$	$5-NN$
Leukemia	98.79	98.79	98.79	98.79	98.79	98.79
Lymphoma	100	100	100	100	100	100
Srbct	99.55	99.55	99.55	99.55	99.55	99.55
Tr41	87.95	87.95	87.95	87.95	87.95	87.95
wap	79.51	79.51	79.51	79.51	79.51	79.51
K1a	83.24	83.24	83.24	83.24	83.24	83.24
$\text{Yale}_{32\times32}$	84.83	84.83	84.83	84.83	84.83	84.83
Yale B	95.82	95.82	95.82	95.82	95.82	95.82
$\text{ORL}_{32\times32}$	98.81	98.81	98.81	98.81	98.81	98.81

Table 3.2 : Comparison of classification accuracy
for OLDA/new (Algorithms 1) and OLDA (Algorithm 2)

Table 3.2 indicates clearly that the classification accuracy of OLDA/new is competitive with OLDA. Notice that for most of the data, the accuracy is the same for these small values of K's in the K-NN algorithm. This is most likely due to the fact that the K value that we pick are close to each other and hence no significant differences are observed. Similar observation can be seen in [59]

	OLDA/new	OLDA
Leukemia	0.1329	0.3649
Lymphoma	0.1194	0.3468
Srbct	0.0601	0.1852
Tr41	8.0985	26.2900
wap	58.5860	121.1310
K1a	0.6860	2.1710
$\text{Yale}_{32\times32}$	0.2305	0.7685
$\text{ORL}_{32\times32}$	1.0475	3.1360
Yale B	100.8755	366.4930

Table 3.3 : Comparison of CPU time
for OLDA/new (Algorithms 1) and OLDA (Algorithm 2)

From the comparison of the CPU time, it is clear that our algorithm is much faster than the original implementation of the Orthogonal LDA while retaining similar accuracy.

In this section, we developed a new generalized LDA method for undersampled problem. The computed optimal transformation matrix by our method is column orthogonal, thus, our method is a new orthogonal LDA method. Furthermore, our method is implemented by using only orthogonal transformations without computing any eigendecomposition and matrix inverse, consequently, our method is inverse-free and numerically stable. In addition, our LDA method has an acceptable computational complexity, that is about the cost of a economic QR factorization of the data matrix $A \in \mathbb{R}^{m\times n}$ with column pivoting and a QR factorization of a $\gamma \times (\gamma - q)$ matrix, here k is the cluster number of data, γ is the rank of the total scatter matrix, q is the rank of the between-cluster scatter matrix, $\gamma \leq n-1$, and $q \leq k-1$, thus, our LDA method is a fast one. The effectiveness of our new method has also been illustrated by some real-world data sets.

3.5 Relationship between Classical LDA and Orthogonal LDA

OLDA and LDA/GSVD are based on the optimization (2.3) and (2.4) respectively. Experiments have demonstrated that these two methods are very competitive with other existing algorithms in terms of classification accuracy. Theorem 3.2.2 implies that the optimization problems (2.4) admit a same column orthogonal solution. Motivated by this fact, we study the relationship between the solutions of the two optimization problems

Theorem 3.5.1. *: Any solution of the optimization problem (2.4) is also a solution of the optimization problem (2.3), but the converse is not necessarily true.*

Proof. First, let G be a solution of the optimization (2.4), with the notation in the previous theorem and its proof, for any $G \in \mathbb{R}^{m\times l}$, denote

$$\begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix} = \begin{bmatrix} Q_{1,2} & 0 & 0 \\ Q_{2,2} & I & 0 \\ 0 & 0 & I \end{bmatrix}^{-1} \begin{bmatrix} Q_1^T \\ \widetilde{Q_1^T} \end{bmatrix} G,\ G_1 \in \mathbb{R}^{q\times l}, G_2 \in \mathbb{R}^{\gamma\times l}, G_3 \in \mathbb{R}^{(m-q-\gamma)\times l},$$

then we have

$S_b^L = G_1^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_1$

$S_w^L = G_2^T R_{2,2} R_{2,2}^T G_2 + G_1^T Q_{1,2}^T R_{1,2} R_{1,3}^T Q_{1,2} G_1,$

$S_t^L = G_2^T R_{2,2} R_{2,2}^T G_2 + G_1^T Q_{1,2}^T (R_{1,1} R_{1,1}^T + R_{1,3} R_{1,3}^T) Q_{1,2} G_1$

Note that $rank(G^T H_b) = rank(H_b)$ implies

$$\begin{aligned} rank(S_b^L) &= rank(G^T H_b H_b^T G) = rank(G^T H_b) = rank(H_b) \\ &= rank(H_b H_b^T) = rank(S_b) = rank(R_{1,1} R_{1,1}^T) = q \end{aligned}$$

so,

$$rank(G_1^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_1) = q$$

which yields

$$rank(G_1) = q$$

i.e., G_1 is of full row rank. Consequently, there exists an orthogonal matrix $W \in \mathbb{R}^{l\times l}$ such that

$$G_1 W = [G_{1,1}\ 0]\,, G_{1,1} \in \mathbb{R}^{q\times q}, rank(G_{1,1}) = q.$$

Denote

$$G_2 W = [G_{2,1}\ G_{2,2}], G_{2,1} \in \mathbb{R}^{\gamma\times q}$$

then we have

$$W^T S_b^L W = \begin{bmatrix} G_{1,1}^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_{1,1} & 0 \\ 0 & 0 \end{bmatrix}$$

$$W^T S_w^L W = \begin{bmatrix} G_{2,1}^T \\ G_{2,2}^T \end{bmatrix} R_{2,2} R_{2,2}^T [G_{2,1}\ G_{2,2}] + \begin{bmatrix} G_{1,1}^T Q_{1,2}^T R_{1,3} R_{1,3}^T Q_{1,2} G_{1,1} & 0 \\ 0 & 0 \end{bmatrix}$$

$$W^T S_t^L W = \begin{bmatrix} G_{2,1}^T \\ G_{2,2}^T \end{bmatrix} R_{2,2} R_{2,2}^T [G_{2,1}\ G_{2,2}] + \begin{bmatrix} G_{1,1}^T Q_{1,2}^T (R_{1,1} R_{1,1}^T + R_{1,3} R_{1,3}^T) Q_{1,2} G_{1,1} & 0 \\ 0 & 0 \end{bmatrix}$$

According to Lemma 3.2.1 and (3.3), we have

$$Trace((S_b^L)^{(+)} S_w^L) = Trace_{eig}(S_b, S_w) = Trace((R_{1,1} R_{1,1}^T)^{-1} (R_{1,3} R_{1,3}^T)).$$

We obtain

$$
\begin{aligned}
&Trace((G_{1,1}^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_{1,1})^{-1}(G_{2,1}^T R_{2,2} R_{2,2}^T G_2 + G_{1,1}^T Q_{1,2}^T R_{1,3} R_{1,3}^T Q_{1,2} G_{1,1})) \\
&= Trace((R_{1,1} R_{1,1}^T)^{-1}(R_{1,3} R_{1,3}^T)) \\
&= Trace((G_{1,1}^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_{1,1})^{-1}(G_{1,1}^T Q_{1,2}^T R_{1,3} R_{1,3}^T Q_{1,2} G_{1,1}))
\end{aligned}
$$

which gives

$$G_{2,1}^T R_{2,2} R_{2,2}^T G_{2,1} = 0$$

i.e.,

$$G_{2,1} = 0$$

In return, we have

$$
W^T S_b^L W = \begin{bmatrix} G_{1,1}^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_{1,1} & 0 \\ 0 & 0 \end{bmatrix}
$$

$$
W^T S_b^L W = \begin{bmatrix} G_{1,1}^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_{1,1} & 0 \\ 0 & 0 \end{bmatrix}
$$

Furthermore,

$$
\begin{aligned}
&Trace((S_t^L)^{(+)} S_b^L) \\
&= Trace((G_{1,1}^T Q_{1,2}^T (R_{1,1} R_{1,1}^T + R_{1,3} R_{1,3}^T) Q_{1,2} G_{1,1})^{-1}(G_{1,1}^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2} G_{1,1})) \\
&= Trace((R_{1,1} R_{1,1}^T + R_{1,3} R_{1,3}^T)^{-1}(R_{1,1} R_{1,1}^T)) \\
&= Trace(S_t^{(+)} S_b)
\end{aligned}
$$

by (3.2). Hence we know by using Lemma 3.2.1 that G also solves the optimization problem (2.3).

Conversely, take

$$G = Q_1(:, 1 : q + \gamma),$$

Then

$$S_b^L = G^T A_2 A_2^T G = (Q_1(:, 1 : q + \gamma)A_2)(Q_1(:, 1 : q + \gamma)A_2)^T = \begin{bmatrix} R_{1,1}R_{1,1}^T & 0 \\ 0 & 0 \end{bmatrix}$$

$$S_w^L = G^T A_3 A_3^T G = (Q_1(:, 1 : q + \gamma)A_3 V^T)(Q_1(:, 1 : q + \gamma)A_3 V^T)^T$$

$$= \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix} \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix}^T$$

$$S_t^L = S_b^L + S_w^L = \begin{bmatrix} R_{1,1}R_{1,1}^T & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix} \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix}^T$$

It is obvious that

$$Trace((S_t^L)^{(+)} S_b^L)$$

$$= Trace((\begin{bmatrix} R_{1,1}R_{1,1}^T & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix} \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix}^T)^{-1} \begin{bmatrix} R_{1,1}R_{1,1}^T & 0 \\ 0 & 0 \end{bmatrix})$$

$$= Trace((R_{1,1}R_{1,1}^T + R_{1,3}R_{1,3}^T)^{-1} R_{1,1}R_{1,1}^T)$$

$$= Trace(S_t^{(+)} S_b)$$

by (3.2).

Thus, Lemma (3.2.1) implies that G solves the optimization problem (2.3). However, since

$$(S_b^L)^{(+)} S_w^L = \begin{bmatrix} R_{1,1}R_{1,1}^T & 0 \\ 0 & 0 \end{bmatrix}^{(+)} \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix} \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix}^T$$

and so if $R \neq 0$, we have

$$\begin{aligned} Trace((S_b^L)^{(+)} S_w^L) &= Trace((R_{1,1}R_{1,1}^T)^{-1}(R_{1,2}R_{1,2}^T + R_{1,3}R_{1,3}^T)) \quad > Trace((R_{1,1}R_{1,1}^T)^{-1}(R_{1,3}R_{1,3}^T)) \\ &= Trace_{eig}(S_b, S_w) \end{aligned}$$

by (3.3).

Hence by using Lemma 3.2.1, G is not a solution of (2.4) if $R_{1,2} \neq 0$. □

As a result according to Theorem 5, any LDA method based on (2.4) can be regarded as a special LDA method based on the criterion (2.3) but the converse need not be true.

Chapter 4

Null Space Based LDA

4.1 Review of Null Space Based LDA

Null Space Based LDA was introduced in [26]. The main idea is simply to project the between class scatter matrices, S_b to the nullspace of S_w. In another words, all the data from the same class are going to be mapped to the same point. This is clear because if

$$G^T(a_j - c_i)(a_j - c_i)^T G = 0$$

then we have

$$G^T(a_j - c_i) = 0$$

and hence in the reduced space, we are going to be left with a maximum of k points. After which, we want to further maximize the distances of the points, which imply that we want to maximize the new between class scatter.

The assumption of the model is that the null space of within class scatter matrices contains sufficient information to discriminate the distinct classes apart provided the projection of the between class scatter matrices is not zero in that direction, in another words, there is a possibility that when we first project the data to the nullspace of S_w, we might even get exactly 1 point, in that case nullspace LDA cannot be applied. However, for a lot of real life data, this rarely occur.

By maximizing the between class scatter after projection to the null space of the within class scatter, the singularity problem that has been bothering us from the beginning would have been overcome easily. Real life data has shown that this method is especially useful for facial recognition.

The objective to be considered for NLDA can be stated as such

$$G = argmax_{G^T S_w G=0} trace(G^T S_b G) \tag{4.1}$$

The algorithm presented in [26] is stated as Algorithm 4. Their approach is to compute the nullspace of S_w using SVD, then perform SVD to maximize the scatter again. They work on the full scatter matrices directly and did not exploit the structure of the scatter matrices as the representation of $S_i = H_i H_i^T, i = t, b, w$ were not known to them back then, as a result, the computation of the optimal G was very expensive back then. It involves the computation of the null space of S_w by using SVD. After all, the dimension of null space of S_w is at least $m + k - n$, so for a problem that is highly undersampled, the computational cost is going to be very high.

Relative to the algorithm that was originally proposed in [26],Huang et al. [42] improved the algorithm by first removing the null space of the total scatter matrices of S_t, this can be done due to the observation that $null(S_t) = null(S_b) \cap null(S_w)$ which is true due to the fact that the total scatter matrices are symmetric positive semidefinite as well as $S_t = S_b + S_w$, whereby $null(A)$ denotes the nullspace of A. Hence by removing the common null space, the value of $trace(S_b)$ and $trace(S_w)$ won't be affected at all, details can be found at [42] and [55], this approach is actually just an application of PCA (principal component analysis). This approach was adopted in Algorithm 5. In his approach, the null space of S_t was removed by projecting the scatter matrices to the range space of S_t, of which its basis can easily be obtained from the economic SVD of H_t. We will now go through the main idea of the algorithm in [42].

Let $H_t = U\Sigma V^T$, then

$$S_t = H_t H_t^T = U\Sigma\Sigma^T U^T = U \begin{pmatrix} \Sigma^2 & 0 \\ 0 & 0 \end{pmatrix} U^T$$

Let $U = (U_1\ U_2)$ be a partition of U such that $U_1 \in \mathbb{R}^{m\times t}$ and $U_2 \in \mathbb{R}^{m\times(m-t)}$, then U_1 would provide us with the range space of S_t

Notice that by removing the common null space of the between class scatter matrices and within class scatter matrices, it won't affect the quality of classification as $trace(S_t^L)$ and $trace(S_b^L)$ would remain the same.

Let's denote the projected scatter matrices as follows:

$$\widetilde{S_b} = U_1^T S_b U_1 \quad \widetilde{S_w} = U_1^T S_w U_1 \quad \widetilde{S_t} = U_1^T S_t U_1$$

Notice that the computation involved only requires us to compute reduced SVD of H_t to obtain U_1 and full SVD is not necessary and avoided the computation of full SVD of S_t. Furthermore, in computation, we do not even form the reduced scatter matrices explicitly, but rather we just form their factors such as H_b and H_w.

Hence we have simplified our problem in terms of the size of the matrices has been greatly reduced and we just have to find N such that

$$N = argmax_{N^T\widetilde{S_w}N=0} trace(N^T\widetilde{S_b}N) \tag{4.2}$$

The above optimization criteria implies that columns of N lie inside the nullspace of $\widetilde{S_w}$ and maximizes $trace(N^T\widetilde{S_b}N)$

To satisfy the nullspace constraint, we just have to find a matrix M such that $N = WM$ where the columns of W span the nullspace of $\widetilde{S_w}$.

Next, we have to consider an issue; the constraint above certainly is unbounded and

can make the objective value tends to infinity. This is clear, because if N is a solution, then $2N$ is a better solution unless the objective function is zero which implies that our goal has failed. To make it bounded and meaningful, we can impose the orthogonality condition on M, of which we know that the optimal value can be obtained from the SVD of $W^T \widetilde{S}_b W$ as we are just taking the principal components and we can choose our optimal transformation matrix as

$$G = U_1 W M$$

The theory seems very easy to understand but computational wise, as we have seen above, we need to compute 3 SVD which is very expensive. We shall consider an approach which is SVD free and inverse free.

4.2 New Implementation of Null Space Based LDA

Theorem 4.2.1. *Denote the column orthogonal matrix* $[P_1, P_2, P_3] \in \mathbb{R}^{m\times(n+k)}$ *be*

$$[A_3\ A_2] = [P_1\ P_2\ P_3] \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \\ 0 & 0 \end{bmatrix}$$

$$R_{1,1} \in \mathbb{R}^{q\times(n-k)}\ R_{2,2} \in \mathbb{R}^{(\gamma-q)\times(k-1)}$$

$$P_1 \in \mathbb{R}^{m\times q}\ P_2 \in \mathbb{R}^{m\times(\gamma-q)}$$

where $R_{1,1}$ *and* $R_{2,2}$ *are of full rank and*

$$q = rank(A_3)\ \gamma = rank\,[A_3\ A_2]$$

$G = P_2$ *solves the optimization problem of null space LDA and it is column orthogonal.*

Proof. From the QR decomposition of $[A_3\ A_2]$ above, we can see that $A_3 = P_1 R_{1,1}$. Since we know that $S_w = A_3 A_3^T$, hence we have $null(S_w) = null(A_3^T)$, to find a basis of

null space of A_3^T, we can choose them to the columns of $[P_2\ P_3\ P_4]$ where $[P_1\ P_2\ P_3\ P_4]$ is an orthogonal matrix. Hence if we let$N = [P_2\ P_3\ P_4]$, its columns would span the null space of the within class scatter matrix. We will project the between class scatter matrix to the null space and consider the eigenvalue decomposition.

$$[P_2\ P_3\ P_4]^T S_b [P_2\ P_3\ P_4] = [U_1\ U_2] \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} = U_1 \Sigma U_1^T \tag{4.3}$$

According to the discussion of the previous section, we know that the maximum value is $trace(U_1^T [P_2\ P_3\ P_4]^T S_b [P_2\ P_3\ P_4] U_1)$ but observe that

$$\begin{aligned}
&trace(U_1^T [P_2\ P_3\ P_4]^T S_b [P_2\ P_3\ P_4] U_1) \\
&\leq trace([P_2\ P_3\ P_4]^T S_b [P_2\ P_3\ P_4]) \\
&= trace([P_2\ P_3\ P_4]^T (P_1 R_{1,2} + P_2 R_{2,2})(P_1 R_{1,2} + P_2 R_{2,2})^T [P_2\ P_3\ P_4]) \\
&= trace\left(\begin{bmatrix} R_{2,2} \\ 0 \end{bmatrix} \begin{bmatrix} R_{2,2}^T\ 0 \end{bmatrix} \right) \\
&= trace \begin{pmatrix} R_{2,2} R_{2,2}^T & 0 \\ 0 & 0 \end{pmatrix} \\
&= trace(R_{2,2} R_{2,2}^T)
\end{aligned}$$

whereby the inequality is due the result in the appendix of [39]. After we obtain this upper bound, what we need to show is that by letting $G = P_2$, this upper bound can be attained.

The fact that it is column orthogonal is obvious by the definition of P_2 as it is defined by parts of columns of Q in the QR decomposition of $[A_3\ A_2]$. We will now check whether it satisfies the constraint.

$$\begin{aligned} G^T S_w G &= P_2^T A_3 A_3^T P_2 \\ &= P_2^T (P_1 R_{1,1})(R_{1,1}^T P_1^T) P_2 \\ &= 0 \end{aligned}$$

The last equality is due to $[P_1\ P_2]$ is a column orthogonal matrix. Hence the constraint is satisfied. To check whether it in fact attains the maximum value, observe that

$$\begin{aligned} trace(P_2^T S_b P_2) &= trace(P_2^T (P_1 R_{1,2} + P_2 R_{2,2})(P_1 R_{1,2} + P_2 R_{2,2})^T P_2) \\ &= trace(R_{2,2} R_{2,2}^T) \end{aligned}$$

Hence the maximum value is attained and the proof is now complete. □

Of course, similar to the fast implementation of Orthogonal LDA, we can adopt a pre-processing step such as taking QR decomposition first would further cut down the workload. This trick and the theorem above leads to the following algorithm, called NLDA/new for short:

The main features of Algorithm 3 are as follows:

1. The transformation matrix G in NLDA/new is column orthogonal

2. Algorithm 3 is implemented by only orthogonal transformations without computing any eigenvalue decomposition and matrix inversion, hence it is inverse free and numerically stable.

3. Step 4 is carried out by the economic QR factorization of $[A_3\ A_2]$ with column pivoting, as a result, the cost of Algorithm 3 is about the cost of an economic QR factorization of the data matrix A with column pivoting, therefore, Algorithm 3 is a fast one.

Algorithm 3: NLDA/new

Input: data matrix A, cluster number k, $[m,n] = size(A)$
and Index vector Ind=$[n_1 \ \sum_{i=1}^{2} n_i \ \ldots, \sum_{i=1}^{k} n_i]^T$
Output: Transformation matrix G

1. Compute the QR decomposition of $A = QR$, thereafter let $A = R$
2. Set $n1 = Ind(1),\ w(1,1) = \sqrt{n1}$
$v = \left[1 - \sqrt{n1}\ 1\ \ldots\ 1\right]^T / \sqrt{n1 - \sqrt{n1}} \in \mathbb{R}^{n1 \times n1}$
Compute
$A(:, 1:n1) = A(:, 1:n1) - (A(:, 1:n1) * v) * v^T$; For $i = 2:k$
Set $ni = Ind(i) - Ind(i-1),\ w(i,1) = \sqrt{ni}\ v = \left[1 - \sqrt{ni}\ 1 \ldots\ 1\right]^T / \sqrt{ni - \sqrt{ni}}$
Compute
$A(:, Ind(i-1)+1 : Ind(i))$
$= A(:, Ind(i-1)+1 : Ind(i)) - (A(:, Ind(i-1)+1 : Ind(i)) * v) * v^T$
$v = A(:,i), \quad A(:,i) = A(:, Ind(i-1)+1),\ A(:, Ind(i-1)+1) = v;$
3. Set $w(1,1) = w(1,1) - \sqrt{n}$, $w = w/\sqrt{n - \sqrt{n} * n1}$ and compute
$A(:, 1:k) = A(:, 1:k) - (A(:, 1:k) * w) * w^T$
4. Form A_2 and A_3 and compute the decomposition of $[A_3\ A_2]$ via economic QR decomposition with column pivoting;
5. Let $G := Q * P_2$.

Notice that a remark is that we can also replace A_2 and A_3 with H_b and H_w respectively in the Algorithm above. However, this would actually be slower than our algorithm as we deal with bigger matrices instead.

4.3 Numerical Experiments

There are a few variants of implementation of the null space based LDA, including the one in [26] in 2000 and also [5] in 2006. In order to compare the efficiency and accuracy of the numerical methods, we implemented them and compare the numerical result.

The pseudocode of [26] is stated in Algorithm 4 as follows:

Algorithm 4: NLDA 2000

Input: data matrix A, cluster number k, $[m,n] = size(A)$
and Index vector Ind=$[n_1 \ \sum_{i=1}^{2} n_i \ \ldots, \sum_{i=1}^{k} n_i]^T$
Output: Transformation matrix G

1. Form matrices H_b and H_w;
2. Compute the SVD of $H_w = U_w \begin{bmatrix} \Sigma_w & 0 \\ 0 & 0 \end{bmatrix} V_w^T$, $\Sigma_w \in \mathbb{R}^{q \times q}, q = rank(H_w)$
3. Compute $\widetilde{H_b} = U_{w2}(U_{w2}^T H_b)$
4. Compute the economic SVD of $\widetilde{H_b}$ to get column orthogonal matrix $\widetilde{U_{b1}}$ such that its columns are eigenvectors of $\widetilde{H_b}\widetilde{H_b^T}$ corresponding to the nonzero eigenvalues;
5. $G = U_{w2}U_{w2}^T\widetilde{U_{b1}}$

Comparing to the new algorithm, we need to compute SVD of matrices, hence it is going to be expensive and we might need to compute inverse in step 2, which will be problematic if the matrix that we are taking inverse is ill-conditioned.

We can also take a look at the pseudocode provided in [5] as stated below

Algorithm 5: NLDA 2006
Input: data matrix A, cluster number k, $[m,n] = size(A)$ and Index vector Ind=$[n_1 \ \sum_{i=1}^{2} n_i \ \ldots, \sum_{i=1}^{k} n_i]^T$
Output: Transformation matrix G
1. Form matrices H_t, H_b and H_w;
2. Compute the SVD of $H_t = U_1 \Sigma_t V_1^T$;
3. Form the matrices $\widetilde{H_b} = U_1 H_b$ and $\widetilde{H_w} = U_1^T H_w$;
4. Compute the SVD of $\widetilde{H_w^T}$ to get the null space W of $\widetilde{H_w^T}$;
5. Compute the economic SVD of $W^T \widetilde{H_b}$ to get matrix M consisting of the top eigenvectors of $W^T \widetilde{H_b} \widetilde{H_b} W$;
6. Compute $G = U_1 W M$

In the following we list the main computational cost of the algorithms. We only consider the undersampled case, that is $m > n$.

Computational Complexity of Algorithm 3 (NLDA/new)
Step 1: $4mn^2 - 2n^3$,
Step 2: $\mathbf{O}\ (n^2)$,
Step 3: $2n(n-1)^2 - \frac{4}{3}(n-1)^3 + 4n(n-1)\gamma - 4\gamma^2 n + \frac{4}{3}\gamma^3$,
Step 4: $\frac{2}{3}n^3 + 4n^2 p - 4np^2 + \frac{4}{3}p^3$,
Step 5: $\mathbf{O}\ (mn(\gamma - q))$.
Computational Complexity of Algorithm 4 (NLDA 2000)
Step 1: $\mathbf{O}\ (mn)$,
Step 2: $4m^2 n - 8mn^2$,
Step 3: $4m(n-q)k$,
Step 4: $14mk^2 - 2k^3$,
Step 5: $4m(n-q)(\gamma - q)$.
Computational Complexity of Algorithm 5 (NLDA 2006)
Step 1: $\mathbf{O}\ (mn)$,
Step 2: $14m^2 n - 2n^3$,
Step 3: $2mn(n+k)$,
Step 4: $12n^3$,
Step 5: $2n(n-q)k + 14(n-q)k^2 - 2k^3$,
Step 6: $2mn(\gamma - q) + 2n(n-q)\gamma - q$.

4.4 Simulation Output

	NLDA 2000			NLDA 2006			NLDA/new		
data	$1-NN$	$3-NN$	$5-NN$	$1-NN$	$3-NN$	$5-NN$	$1-NN$	$3-NN$	$5-NN$
Leukemia	97.05	97.05	97.05	97.05	97.05	97.05	97.05	97.05	97.05
Lymphoma	98.61	98.61	98.61	98.61	98.61	98.61	98.61	98.61	98.61
$Yale_{32\times32}$	77.88	77.88	77.88	77.88	77.88	77.88	77.88	77.88	77.88
$Yale_{64\times64}$	83.13	83.13	83.13	83.13	83.13	83.13	83.13	83.13	83.13
$ORL_{32\times32}$	90.82	90.82	90.82	90.82	90.82	90.82	90.82	90.82	90.82
$ORL_{64\times64}$	91.45	91.45	91.45	91.45	91.45	91.45	91.45	91.45	91.45
classic	86.87	86.87	86.88	86.77	86.77	86.77	86.77	86.77	86.77
K1a	82.58	82.58	82.58	82.58	82.58	82.58	82.58	82.58	82.58
K1b	96.94	96.94	96.94	96.94	96.94	96.94	96.94	96.94	96.94
cranmed	98.37	98.37	98.37	98.37	98.37	98.37	98.37	98.37	98.37
review	–	–	–	84.42	84.42	84.42	84.42	84.42	84.42
sports	–	–	–	69.65	69.65	69.65	69.65	69.65	69.65

Table 4.1 : Comparison of classification accuracy
for NLDA 2000 (Algorithms 4), NLDA 2006 (Algorithm 5), andNLDA/new (Algorithm 3)

Table 4.1 indicates clearly that the classification accuracy of NLDA/new is competitive with NLDA 2000 and NLDA 2006.

	NLDA 2000	NLDA 2006	NLDA/new
Leukemia	82.04	0.32	0.18
Lymphoma	93.92	0.63	0.19
$Yale_{32\times32}$	1.25	0.20	0.05
$Yale_{64\times64}$	28.68	0.61	0.38
$ORL_{32\times32}$	2.23	2.01	0.29
$ORL_{64\times64}$	44.83	3.38	1.25
classic	24011.00	12613.00	3075.60
K1a	3058.90	536.57	171.91
K1b	2495.30	465.78	125.32
cranmed	13314.00	694.02	235.64
review	–	4538.70	2178.10
sports	–	26703.83	7980.12

Table 4.2 : Comparison of CPU time
for NLDA 2000 (Algorithms 4), NLDA 2006 (Algorithm 5), andNLDA/new (Algorithm 3)

Table 4.2 indicates clearly the CPU time of the NLDA/new is much faster than the NLDA 2000 and NLDA 2006.

Notice that some values are not printed in the table above, this is due to some data size is too big for NLDA 2000 to be implemented.

4.5 Relationship between Classical LDA and Nullspace Based LDA

The NLDA [20],[26] aims to maximize the between-cluster distance in the null space of the within-cluster scatter matrix, the singularity problem of the total scatter matrix S_t is thus implicitly avoided. The basic idea behind NLDA is that the null space of S_w may contain significant discriminant information if the projection of S_b is not zero in that direction. Both the NLDA method in [20, 26] and the OLDA in [8] result in column orthogonal optimal transformations. Many numerical results show they often lead to similar performance for high-dimensional data. Due to this observation, it was proved in [5] that under the condition

$$rank(S_t) = rank(S_b) + rank(S_w), \tag{4.4}$$

the optimal transformation matrix obtained by its OLDA algorithm is also a solution of the optimization problem (2.4). The following result reveals more intrinsic relationship between these two optimization problems.

Lemma 4.5.1. *Any solution G of the optimization problem (2.4) is also a solution of the optimization problem (2.2) if and only if the condition (4.4) holds true.*

Proof. With notation in Theorem 3.2.2 and its proof, it is easy to know by using Lemma 2.2.2 and the properties that $R_{1,1}$ and $R_{2,2}$ are of full row rank and $R_{2,2}$ is nonsingular

that

$$
\begin{aligned}
& \text{the condition (4.4) holds} \\
\Leftrightarrow \quad & rank([A_2\ A_3]\begin{bmatrix} A_2 & A_3 \end{bmatrix}^T) = rank(A_2A_2^T) + rank(A_3A_3^T) \\
\Leftrightarrow \quad & rank([A_2\ A_3]) = rank(A_2) + rank(A_3) \\
\Leftrightarrow \quad & rank(\begin{bmatrix} Q_1\ \widetilde{Q}_1 \end{bmatrix}^T [A_2\ A_3V^T] = rank(\begin{bmatrix} Q_1\ \widetilde{Q}_1 \end{bmatrix}^T A_2) \\
& \qquad\qquad\qquad\qquad\qquad\qquad + rank(\begin{bmatrix} Q_1 & \widetilde{Q}_1 \end{bmatrix}^T [A_3V^T] \\
\Leftrightarrow \quad & rank\begin{bmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ 0 & R_{2,2} & 0 \end{bmatrix} = rank\begin{bmatrix} R_{1,1} \\ 0 \end{bmatrix} + rank\begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix} \\
\Leftrightarrow \quad & \gamma = q + (\gamma - q) + rank(R_{1,3}) \\
\Leftrightarrow \quad & R_{1,3} = 0. \qquad (4.5)
\end{aligned}
$$

We will work with this condition instead.

For any solution $G \in \mathbb{R}^{m\times l}$ of the optimization problem (2.4), denote

$$
\begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix} = \begin{bmatrix} Q_1^T \\ \widetilde{Q}_1^T \end{bmatrix} G, G_1 \in \mathbb{R}^{q\times l},\ G_2 \in \mathbb{R}^{(\gamma-q)\times l}, G_3 \in \mathbb{R}^{(m-\gamma)\times l}.
$$

Then, the following is a direct consequence of Lemma 2.2.2,

$$
\begin{aligned}
G^TS_wG = 0 \quad \Leftrightarrow \quad & G^TA_3 = 0 \\
\Leftrightarrow \quad & G^TA_3V^T = 0 \\
\Leftrightarrow \quad & [G_1^T\ G_2^T] \begin{bmatrix} R_{1,2} & R_{1,3} \\ R_{2,2} & 0 \end{bmatrix} = 0 \\
\Leftrightarrow \quad & \begin{bmatrix} G_1^T & G_2^T \end{bmatrix} \begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} = 0, \quad G_1^TR_{1,3} = 0, \\
\Rightarrow \quad & rank(G_1) \le q - rank(R_{1,3}). \qquad (4.6)
\end{aligned}
$$

Since $G^T A_3 = 0$ is equivalent to

$$G^T A_3 = \begin{bmatrix} G_1^T & G_2^T \end{bmatrix} \begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} = 0,$$

so,

$$\begin{cases} S_t^L = G^T \begin{bmatrix} A_2 & A_3 \end{bmatrix} \begin{bmatrix} A_2 & A_3 \end{bmatrix}^T G \\ \quad = \begin{bmatrix} G_1^T & G_2^T e \end{bmatrix} \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \end{bmatrix} \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \end{bmatrix}^T \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} \\ \quad = G_1^T R_{1,1} R_{1,1}^T G_1, \\ S_b^L = G^T A_2 A_2^T G = \begin{bmatrix} G_1^T & G_2^T \end{bmatrix} \begin{bmatrix} R_{1,1} \\ 0n \end{bmatrix} \begin{bmatrix} R_{1,1} \\ 0 \end{bmatrix}^T \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} \\ \quad = G_1^T R_{1,1} R_{1,1}^T G_1. \end{cases} \tag{4.7}$$

Hence, according to Lemma 3.2.1 and (3.4), we have

$$\begin{aligned} & G \text{ is a solution of the optimization problem (2.2)} \\ \Leftrightarrow \quad & \text{Trace}((S_t^L)^{(+)} S_b^L) = \text{Trace}((R_{1,1}R_{1,1}^T + R_{1,3}R_{1,3}^T)^{-1}(R_{1,1}R_{1,1}^T)) \\ \Leftrightarrow \quad & \text{Trace}((G_1^T R_{1,1} R_{1,1}^T G_1)^{(+)}(G_1^T R_{1,1} R_{1,1}^T G_1)) = \text{Trace}((R_{1,1}R_{1,1}^T + R_{1,3}R_{1,3}^T)^{-1}(R_{1,1}R_{1,1}^T)). \end{aligned}$$

Note that

$$\begin{aligned} q - rank(R_{1,3}) &\geq rank(G_1) \\ &\geq rank(G_1^T R_{1,1} R_{1,1}^T G_1) \\ &= \text{Trace}((G_1^T R_{1,1} R_{1,1}^T G_1)^{(+)}(G_1^T R_{1,1} R_{1,1}^T G_1)), \end{aligned}$$

and Lemma 4.5.2 below gives

$$\text{Trace}((R_{1,1}R_{1,1}^T + R_{1,3}R_{1,3}^T)^{-1}(R_{1,1}R_{1,1}^T)) \geq q - rank(R_{1,3}),$$

so, we obtain

$$\begin{aligned}
& G \text{ is a solution of the optimization problem (2.2)} \\
\Leftrightarrow & \begin{cases} q - rank(R_{1,3}) = \text{Trace}((G_1^T R_{1,1} R_{1,1}^T G_1)^{(+)}(G_1^T R_{1,1} R_{1,1}^T G_1)), \\ q - rank(R_{1,3}) = \text{Trace}((R_{1,1}R_{1,1}^T + R_{1,3}R_{1,3}^T)^{-1}(R_{1,1}R_{1,1}^T)), \end{cases} \qquad (4.8) \\
\Leftrightarrow & \begin{cases} q - rank(R_{1,3}) = \text{Trace}((G_1^T R_{1,1} R_{1,1}^T G_1)^{(+)}(G_1^T R_{1,1} R_{1,1}^T G_1)), \\ R_{1,3} = 0, \text{(by Lemma 4.5.2)} \end{cases} \qquad (4.9) \\
\Leftrightarrow & \quad R_{1,3} = 0, \quad rank(G_1) = q, \\
\Leftrightarrow & \quad \text{Condition (4.4) holds and } rank(G_1) = q. \qquad \text{(by (4.5))} \qquad (4.10)
\end{aligned}$$

Hence it suffices to prove that $rank(G_1) = q$ when $R_{1,3} = 0$

On the other hand, G is a solution of the optimization problem (2.4) and $G^T A_3 = 0$, so,

$$\begin{bmatrix} G_1^T & G_2^T \end{bmatrix} \begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} R_{2,2}^T = G^T A_3 R_{2,2}^T = 0,$$

which gives that

$$\begin{bmatrix} G_1 \\ G_2 \end{bmatrix} = \begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix} U$$

for some $U \in \mathbb{R}^{q \times l}$ with

$$\begin{aligned}
(Q_{1,2}^{-1}G_1)^T(Q_{1,2}^{-1}G_1) &= U^T U = U^T \begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix}^T \begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix} U \\
&= \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}^T \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} \leq \begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix}^T \begin{bmatrix} G_1 \\ G_2 \\ G_3 \end{bmatrix} \\
&= I, \qquad (4.11)
\end{aligned}$$

because the columns of $\begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix}$ forms a basis of the null space of $(\begin{bmatrix} R_{1,2} \\ R_{2,2} \end{bmatrix} R_{2,2}^T)^T$.

Note that under the condition (4.4), $R_{1,3} = 0$, and consequently,

$$(Q_1(:,\ 1:\gamma)\begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix})^T S_w (Q_1(:,\ 1:\gamma)\begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix}) = Q_{1,2}^T R_{1,3} R_{1,3}^T Q_{1,2} = 0.$$

Since $Q_1(:,\ 1:\gamma)\begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix}$ is column orthogonal and G is a solution of the optimization problem (2.4), we must have using (3.7) and (4.7) that

$$\begin{aligned}
& \text{Trace}((Q_{1,2}^{-1}G_1)^T Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2}(Q_{1,2}^{-1}G_1)) \\
= \ & \text{Trace}(G_1^T R_{1,1} R_{1,1}^T G_1) \\
= \ & \text{Trace}(S_b^L) \\
\geq \ & \text{Trace}(Q_1(:,\ 1:\gamma)\begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix}^T S_b (Q_1(:,\ 1:\gamma)\begin{bmatrix} Q_{1,2} \\ Q_{2,2} \end{bmatrix}) \\
= \ & \text{Trace}(Q_{1,2}^T R_{1,1} R_{1,1}^T Q_{1,2}),
\end{aligned}$$

which with (4.11) imply

$$(Q_{1,2}^{-1}G_1)(Q_{1,2}^{-1}G_1)^T = I,$$

thus,

$$rank(G_1) = q. \tag{4.12}$$

Hence, by (4.10) and (4.12) we conclude that G is a solution of the optimization problem (2.2) if and only if the condition (4.4) holds true. □

In the proof above, the following support lemma is used.

Lemma 4.5.2. *Let $C \in \mathbb{R}^{q \times l_1}, D \in \mathbb{R}^{q \times l_2}$ and $rank(D) = q$. Then*

$$\text{Trace}((DD^T + CC^T)^{-1} DD^T) \geq q - rank(C).$$

The equality holds if and only if $C = 0$.

Proof. We can assume without loss of generality that

$$DD^T = \begin{bmatrix} \Theta_{1,1} & \Theta_{1,2} \\ \Theta_{1,2}^T & \Theta_{2,2} \end{bmatrix}, \quad CC^T = \begin{bmatrix} \Phi_{1,1} & 0 \\ 0 & 0 \end{bmatrix},$$

with

$$\Theta_{1,1}, \Phi_{1,1} \in \mathbb{R}^{p\times p}, \quad rank(C) = rank(CC^T) = rank(\Phi_{1,1}) = p.$$

Now, $rank(D) = q$, so, $\Theta_{2,2}$, $\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T$ and $\Phi_{1,1}$ are all symmetric and positive definite. Hence, by using Schur complement, we have

$$\begin{aligned} & \text{Trace}((DD^T + CC^T)^{-1}DD^T) \\ = \ & \text{Trace}\left(\begin{bmatrix} (\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T) + \Phi_{1,1} & 0 \\ 0 & \Theta_{2,2} \end{bmatrix}^{-1} \begin{bmatrix} \Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T & 0 \\ 0 & \Theta_{2,2} \end{bmatrix} \right) \\ = \ & q - p + \text{Trace}((\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T + \Phi_{1,1})^{-1}(\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T)) \\ \geq \ & q - p, \end{aligned}$$

and the equality holds if and only if

$$\text{Trace}((\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T + \Phi_{1,1})^{-1}(\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T)) = 0,$$

i.e., $p = 0$, equivalently, $rank(C) = 0$, because, otherwise,

$$\text{Trace}((\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T + \Phi_{1,1})^{-1}(\Theta_{1,1} - \Theta_{1,2}\Theta_{2,2}^{-1}\Theta_{1,2}^T)) > 0.$$

Therefore, Lemma 4.5.2 follows. □

Theorem 4.5.1 implies that NLDA is a special OLDA provided the condition (4.4) holds. This confirms the numerical results in [5].

Chapter 5

Conclusion

Due to the singularities problems, variations of generalized Linear Discriminant Analysis have been proposed. In this thesis, we have provided a survey of the history, the developments and the current state of art of various implementations of Linear Discriminant Analysis. Various real life applications of Linear Discriminant Analysis are discussed as well.

Though it seems that some models of Linear Discriminant Analysis have been solved theoretically, whereby for example, analysis based on GSVD has been proposed, there are still many issues to address. In particular, when it comes to practical implementation, many existing methods can only solve small scale problems. In the past, there have been various implementations that require methods such as computing Singular Value Decomposition which is very expensive. Some of the existing algorithms even involve computations of inverse which will be a big problem if the initial condition is large. Some versions of Linear Discriminant Analysis require the selection of parameter of whereby selection of parameter is solely based on heuristic approaches.

In this thesis, we proposed two new implementations of Orthogonal Linear Discriminant Analysis as well as Null Space Based LDA. These algorithms are inverse free and numerical stable. Furthermore, they do not involved any weaknesses that we mentioned above. In stead of SVD, only QR algorithms are invoked. Numerical simulations are implemented and they are consistent with our theoretical expectation that they produce similar accuracy with a much shorter CPU time needed.

The future of this area is full of potentials; there are still many open problems in this

area. For example, so far no algorithm which takes advantages of the sparse nature of text documents data have been invented. Furthermore, there are other questions such as does the objective functions used in the models reflect the classification accuracy? or can we come out with an algorithm which is very robust? What are the relationships among all the variants of models of Linear Discriminant Analysis? Hopefully all these questions can be answered in the near future. There is no doubt that this area is going to be filled with excitement and surprises.

Appendix

In this appendix, we include the computational complexities of various algorithms:

Computational Complexity for QR decomposition of $\Theta \in \mathbb{R}^{m\times n}$ with $m \geq n$.
Full QR factorization: $4mn^2 + \frac{2}{3}n^3 - 2mn^2$;
Economic QR factorization: $4mn^2 - 2n^3$
Full QR factorization with column pivoting:
$(4mn^2 - 4mn^2 + \frac{4}{3}n^3) + (4mnp - 2p^2(m+n) + \frac{4}{3}p^3),\ p = rank(\Theta)$
Economic QR factorization with column pivoting: $2mn^2 - \frac{4}{3}n^3 + (4mnp - 2p^2(m+n) + \frac{4}{3}p^3)$

And for the computation of SVD

Computational Complexity for SVD
$(\Theta = U\Sigma V^T, U_1 = U(:, 1:n))$ of $\Theta \in \mathbb{R}^{m\times n}$ with $m \geq n$
$\Sigma : 4mn^2 - \frac{4}{3}n^3$;
$\Sigma, V : 4mn^2 + 8n^3$;
$U, \Sigma : 4m^2n - 8mn^2$;
$U_1, \Sigma : 14mn^2 - 2n^3$;
$U, \Sigma, V : 4m^2n + 8mn^2 + 9n^3$;
$U_1, \Sigma, V : 14mn^2 + 8m^3$

These show that it would clearly improve an algorithm if we can replace SVD by economic QR decomposition.

Bibliography

[1] Y. Guo, T. Hastie, and R. Tibshirani, Regularized Linear Discriminant Analysis and Its Application in Microarray, *Biostatistics*, 8:86–100, 2007.

[2] 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.

[3] L. Wang, and X. Shen, On L_1-norm Multiclass Support Vector Machines: Methodology and Theory, *Journal of the American Statistical Association*, 102:583–594, 2007.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006

[5] 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.

[6] 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.

[7] 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.

[8] J. Ye, Characterization of a Family of Algorithms for Generalized Discriminant Analysis on Undersampled Problems, *J. Mach. Learn. Res.*, 6:483–502, 2005.

[9] J. Ye, and Q. Li, A Two-stage Linear Discriminant Analysis via QR-Decomposition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:929–941, 2005.

[10] P. Howland, and H. Park, Generalizing Discriminant Analysis Using the Generalized Singular Value Decomposition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:995–1006, 2004.

[11] J. Ye, R. Janardan, C. H. Park, and H. Park, An Optimization Criterion for Generalized Discriminant Analysis on Undersampled Problems, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:982–994, 2004.

[12] Delin 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.

[13] Delin Chu, De Lathauwer, Lieven; De Moor, Bart On the computation of the restricted singular value decomposition via the cosine-sine decomposition. SIAM J. Matrix Anal. Appl. 22 (2000), no. 2, 580–601

[14] Delin Chu, De Moor, Bart On the nonuniqueness of the factorization factors in the product singular value decomposition. Linear Algebra Appl. 314 (2000), no. 1-3, 191–203

[15] D. Q. Dai, and P. C. Yuen, Regularized Discriminant Analysis and its Application to Face Recognition, *Pattern Recognition*, 36: 845–847, 2003.

[16] 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.

[17] 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.

[18] P. Baldi, and G. W. Hatfield, *DNA Microarrays and Gene Expression: From Experiments to Data Analysis and Modeling*, Cambridge, 2002.

[19] 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 to 87, 2002.

[20] R. Huang, Q. Liu, H. Lu, and S. Ma, Solving the Small Sample Size Problem of LDA, In *Proc. International Conference on Pattern Recognition*, pp.29–32, 2002.

[21] I.T. Jolliffe, *Principal Component Analysis*, 2nd edition, Springer-Verlag, New York, 2002.

[22] P. Benner, and R. Byers, Evaluating Products of Matrix Pencils and Collapsing Matrix Products, *Numerical Linear Algebra with Applications*, 8:357–380, 2001.

[23] R. Q. Duda, P. E., Hart, and D. G. Stork, *Pattern Classification*, second edition, John Wiley and Sons, Inc., 2001.

[24] Z. Jin, J. Y. Yang, Z. S. Hu, and Z. Lou, Face Recognition based on the Uncorrelated Discriminant Transformation, *Pattern Recognition*, 34:1405 to 1416, 2001.

[25] Z. Jin, J. Y. Yang, Z. M. Tang, and Z. S. Hu, A Theorem on the Uncorrelated Optimal Discriminant vectors, *Pattern Recognition*, 34:2041 to 2047, 2001.

[26] L. Chen, H. M. Liao, M. Ko, J. Lin, and G. Yu, A New LDA-based Face Recognition System Which Can Solve the Small Sample Size Problem, *Pattern Recognition*, 33:1713–1726, 2000.

[27] 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.

[28] G. Kowalski, *Information Retrieval Systems: Theory and Implementation*, Kluwer Academic Publishers, 1997.

[29] G. H. Golub, and C. F. Van Loan, *Matrix Computations*, 3rd edition, The Johns Hopkins University Press, Baltimore, Maryland, 1996.

[30] D. L. Swets, and J. Weng, Using Discriminant Eigenfeatures for Image Retrieval, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:831–836, 1996.

[31] M. W. Berry, S. T. Dumais, and G. W. O' Brie, Using Linear Algebra for Intelligent Information retrieval, *SIAM Review*, 37, 573 to 595, 1995.

[32] W. B. Frakes, and R. Baeza-Yates, *Information Retrieval: Data Structures and Algorithms*, Prentice Hall PTR, 1992.

[33] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, second edition, Academic Press, Inc., 1990.

[34] J. H. Friedman, Regularized Discriminant Analysis, *Journal of the American statistical association*, 84, 165–175, 1989.

[35] A. K. Jain, and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.

[36] L. Duchene, and S. Leclerq, An Optimal Transformation for Discriminant and Principal Component analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:978 to 983, 1988.

[37] C. C. Paige, and M. A. Saunders, "Towards a Generalized Singular Value Decomposition," *SIAM J. Numer. Anal.*, 18:398–405, 1981.

[38] C. F. Van Loan, Generalizing the Singular Value Decomposition, *SIAM J. Numer. Anal.*, 13:76–83, 1976.

[39] J.Ye, Q. Li, H.Xiong, H. Park, R. JanardanV. Kumar IDR/QR: An incremental Dimension Reduction ALgorithm via QR decomposition IEEE transaction on Knowledge and Data Engineering Vol 17, No. 9, 1208– 1221, 2005

[40] W. Zuo, K. Wang, D. ZHang Improvement on Null Space LDA for Face Recognition: A Symmetry Consideration Springer-Verlag Berlin Heidelberg ICB 2006, LNCS 3832, 78– 74, 2005

[41] Yang, J. and Yang, J.Y. Why can LDA be performed in PCA transformed space. Pattern Recognition. v36 i2. 563-566

[42] J. Ye, T. Xiong. Null Space versus Orthogonal Linear Discriminant Analysis Proceedings of the 23rd International Conference on Machine Learning, Puttsburgh, PA, 2006

[43] Ye, J. , Ji, S. A unified framework for generalized Linear Discriminant Analysis Computer Vision and Pattern Recognition, 2008, CVPR 2008 IRRR, Conf 1–7, 2008

[44] R.A, Fisher The use of multiple measurements in taxonomic problems Annals of Eugenics 7, 179-188

[45] B. W. Silverman Density Estimation for Statistics and Data Analysis Chapman and Hall, London , 1986

[46] D. L. Donoho Aide Memoire. High Dimensional Data Analysis: The Curses and Blessings of Dimensionality, 2000

[47] L. Elden Matrix Methods in Data Mining and Pattern Recognition Fundamentals of Algorithms, SIAM

[48] S. Zhu, T. Li A Non-distance Based Clustering Algorithm Neural Networks, 2002. IJCNN apos;02. Proceedings of the 2002 International Joint Conference on Volume 3, Issue , 2002 Page(s):2357 - 2362

[49] W. Zuo, K. Wang, D. Zhang Improvement on Null Space LDA for Face Recognition: A Symmetry Consideration international conference on biometrics, Hong Kong , CHINE (05/01/2006) 2006 , vol. 3832, pp. 78-84

[50] C.D. Sloan, V. Sayarath, J.H. Moore Systems Genetics of Alcoholism NIAAA Publications

[51] C.C. Aggarwal On the Effects of Dimensionaliy Reduction on High Dimensional Similarity Search IBM T.J. Watson Research Center

[52] Y. Song, F. Nie, C. Zhang, S. Xiang A unified framework for semi-supervised dimensionality reduction Pattern Recognition 41 (2008) 2789-2799

[53] H. Park, C. Park Nonlinear Discriminant Analysis using Kernel Functions and the Generalized Singular Value Decomposition SIAM Journal on Matrix Analysis and Applications, 27-1, pp. 98-102, 2005

[54] S. Kongsontana, Y. Rangsanseri Face Recognition using 2DLDA Algorithm Signal Processing and Its Applications, 2005. Proceedings of the Eighth International Symposium Volume 2, August 28-31, 2005 Page(s):675 - 678

[55] H. Park Two-stage Methods for Linear Discriminant Analysis: Equivalent Results at a Lower Cost Technical Report GT-CSE, 2009

[56] Delin Chu, S.T. Goh A New and Fast Orthogonal Linear Discriminant Analysis on Undersampled Problems, submitted

[57] Delin Chu , S.T. Goh A New and Fast Implementation for Null Space Base Linear Discriminant Analysis Pattern Recognition, 2009

[58] 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, Vol. 23, no. 6, 643-660, 2001

[59] H. Park http://dimacs.rutgers.edu/Workshops/Advances/Slides/park.ppt ♯ 307,7,LDA based on GSVD (LDA/GSVD) (Howland, Jeon, Park, SIMAX03, Howland and Park, IEEE TPAMI 04)