Neurocomputing 575 (2024) 127261

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Domain decomposed classification algorithms based on linear discriminant analysis: An optimality theory and applications

Jingwei Li [b,c], Xiao-Chuan Cai [a,*]

[a] Department of Mathematics, University of Macau, Macau, 999078, China
[b] Department of Psychology, University of South Carolina, Columbia, SC 29208, USA
[c] Department of Computer Science, University of Colorado Boulder, Boulder, CO 80309, USA

ARTICLE INFO

Communicated by L. Zhang

MSC:
15A18
01-08

Keywords:
Linear discriminant analysis
Domain decomposition
Optimality theory
Iterative methods
Deflation

ABSTRACT

Linear discriminant analysis (LDA) is a popular technique for supervised classification problems, and it works quite well when the number of classes is small, but the accuracy deteriorates when the number of classes becomes large. In this paper, we propose a domain decomposed method and an iteratively deflated method to improve the classification accuracy. In the domain decomposed LDA, we decompose the given dataset into subsets and apply LDA separately to each subset for the training part of the algorithm. In the testing step, we project the samples into multiple subspaces, contrary to the full space as in the traditional LDA. From the multiple low-dimensional projections we determine the class or classes that the sample belongs to. An optimality theory is developed to show why the new method offers better classification under a technical assumption. In the iteratively deflated method, the traditional LDA method serves as the initial iteration from which we select separable classes to be deflated from the training set, and the remaining samples in the dataset are used for the next iteration. As the process goes on we generate a sequence of projection matrices that are used to determine which class or classes a sample belongs to using certain classification criteria. With the proper choices of the quantile radii in the separable criteria for the training and testing phases, we show that the proposed method is much more accurate than the traditional LDA. To test and compare the two proposed methods, we consider the popular datasets CIFAR-10/100 and a gene expression dataset of cancer patients, and show that the new approaches outperform the traditional LDA by a large margin.

1. Introduction

In machine learning, dimensionality reduction plays an important role, and linear discriminant analysis (LDA) [1] is one of the popularly used techniques to extract the most discriminative features of a dataset. LDA finds the optimal discriminative direction by maximizing the between-class variance while simultaneously minimizing the within-class variance in the projected space. In the reduced low-dimensional feature space, the dataset can be classified without high computational costs.

LDA was originally introduced in taxonomy for the purpose of assigning an individual specimen to a proper group [2,3]. More recently the method and its variants are applied to many problems in data sciences. For example, Huang et al. [4] applied LDA to the classification of cancer patients and compared the classification performance of LDA and its modified variants by applying these methods to six public cancer gene expression datasets. Ricciardi et al. [5] reported the use of data mining techniques to analyze a population of 10,265 people who were evaluated and 22 features were extracted, and linear discriminant analysis was used to classify both normal and pathological patients. The classification accuracies were 84.5 and 86.0 percent, respectively. Le et al. [6] defined the discriminant capacity for each connected component of a graph, and the variables of the most discriminant components are kept. The adapted LDA and the variable selection procedure were initially evaluated with synthetic data, and then applied to real data from PET images of the human brain for the classification of patients with Alzheimer's disease. Sengur et al. [7] investigated the usage of LDA and adaptive neuro-fuzzy inference system to determine the normal and abnormal heart valves from the Doppler heart sounds. The heart valve disorder detection system was composed of three stages, from which a comparative study was realized with a data set containing 215 samples. In the study [8], the classification of invasive ductal carcinoma breast cancer is performed by using the ridge regression method and LDA.

Generally speaking, the traditional LDA performs well when the problem has a small number of classes. The performance deteriorates

* Corresponding author.
E-mail address: xccai@um.edu.mo (X.-C. Cai).

https://doi.org/10.1016/j.neucom.2024.127261
Received 22 December 2022; Received in revised form 5 November 2023; Accepted 14 January 2024
Available online 17 January 2024

the number of classes is large. In the present paper we introduce two techniques to solve classification problems with a relatively large number of classes. In the new algorithms, the traditional LDA is used for subsets of the problem with a smaller number of classes, and the global classification solution is obtained by combining the subspace results with proper scaling. The first technique is referred to as domain decomposition which is a family of methods designed originally for solving partial differential equations [9–12] and recently be used for learning problems [13–16]. In each subdomain, we select a dimension of the reduced space and then apply the traditional LDA in this subspace. The testing is carried out for all subdomains and the classification is determined by a distance vector computed in all subdomains. The second technique is referred to as iterative deflation which is motivated by linear algebra algorithms for eigenvalue calculations [17]. We iteratively remove the well-separated classes during the training phase of the algorithm and several projection matrices are generated to determine if a sample belongs to a certain class or classes.

The rest of the paper is organized as follows. The related works are reviewed in Section 2. In Section 3, we discuss the traditional LDA, then we introduce a domain decomposed linear discriminant analysis in Section 4 and an iteratively deflated linear discriminant analysis in Section 5. To demonstrate the classification accuracy of these two methods, in Section 6, we consider three applications including the CIFAR-10/100 datasets and a gene expression dataset. Finally, we make some concluding remarks in Section 7.

2. Related work

LDA has been extensively used in industry and economics. The applications include, for example, bankruptcy prediction, face recognition, marketing, and earth science. In bankruptcy prediction, Edward et al. [18] introduced an LDA based statistical method to systematically explain which companies entered bankruptcy. In face recognition, each face is represented by some ordered pixel values, LDA is then used to reduce the number of features to a more manageable number before classification. In marketing, LDA was often used to determine the factors that distinguish different types of customers and/or products on the basis of surveys or other collected data. Panayides et al. [19] examined the marketing practices and investigated the marketing strategy-business performance relationship across logistics companies in the Asia-Pacific region. Further discriminant analysis of the significant predictor variables suggested that two variables, market segmentation and positioning, and cross-functional customer focus are useful in differentiating between high and low performers. Sunny et al. [20] used linear models of classification including logistic regression classification, LDA, partial least-square discriminant analysis, and penalized discriminant analysis, and nearest Shrunken discriminant analysis were considered in the study to predict the stock prices of top six banks of Bangladesh. In earth science, Tahmasebi et al. [21] employed LDA and 24 electron microprobe element analysis related to the rock samples were used as the input data to classify different rocks and separate the variety of alterations. In AI, Gorban et al. [22] presented the probabilistic basis for fast non-destructive correction of AI systems. It showed that simple linear Fisher discriminant analysis can separate the situations with errors from correctly solved tasks even for very large samples. In biology, Mngadi et al. [23] sought to determine certain characteristical values in discriminating and mapping commercial forest species. Based on LDA, the multi-spectral imagery showed an overall classification accuracy of 84%, with bands such as the red-edge, narrow near infrared, and short wave infrared particularly influential in discriminating individual forest species stands.

There are several advanced studies to improve different aspects of LDA. Khoder et al. [24] extended and improved a scheme for linear feature extraction that can be used in supervised multi-class classification problems. The paper introduced an iterative alternating minimization scheme to estimate and update the linear transformation and the orthogonal matrix via the steepest descent gradient technique. Kim et al. [25] provided a novel method to define sample similarity in deep metric learning using LDA to analyze the characteristics of the embedding space. The technique is applied to an existing deep metric learning scheme and proved to provide better similarity than previous works. Sifaou et al. [26] proposed an improved LDA classifier based on the assumption that the covariance matrix follows a spiked covariance model. The main principle of the proposed technique was the design of a parametrized inverse covariance matrix estimator, the parameters of which are shown to be easily optimized. Ortega-Martinez et al. [27] used single stimulus Kalman filter regression to estimate the hemodynamic response function produced by subjects performing different tasks. It trained an LDA classifier with a subset of the data and performed cross-validation to estimate the mean classification accuracy. Huang et al. [28] proposed multi-subspace ratio-trace LDA (ms-LDA) and trace-ratio LDA (ms-LDA-tr) methods. It showed that they outperformed their ratio-trace and single-subspace counterparts. The methods can be used to solve some rather challenging classification problems with a large number of classes. The main difference with the proposed algorithm is that [28] introduced the centroid-based multi-subspace scatter matrices and ddLDA introduced the within-class scatter matrix and the between-class scatter matrix in the subdomains. Both methods improve the traditional LDA.

3. A brief review of linear discriminant analysis

LDA [29] is an algorithm that projects a set of vectors linearly to a lower-dimensional space such that vectors with different features are best separated into different classes. We first introduce some notations. Let X be the dataset of vectors in $\mathbb{R}^m$ to be classified into c classes and n_j is the number of samples in the jth class. $x^{ij} \in \mathbb{R}^m$ denotes the ith sample in the jth class. $n^c = \sum_{j=1}^{c} n_j$ is the total number of samples in the dataset. $X^j = [x^{1j}, \dots, x^{n_j j}] \in \mathbb{R}^{m\times n_j}$ is the matrix consisting of all samples in the jth class. In other words, $X = [X^1, X^2, \dots, X^c] \in \mathbb{R}^{m\times n^c}$ is a matrix representation of the dataset.

Let $\mu \in \mathbb{R}^m$ be the mean of the dataset,

$$\mu = \frac{1}{n^c}\sum_{j=1}^{c}\sum_{i=1}^{n_j} x^{ij}$$

and $\mu^j \in \mathbb{R}^m$ the mean of the jth class,

$$\mu^j = \frac{1}{n_j}\sum_{i=1}^{n_j} x^{ij}.$$

Then we define the **within-class scatter** matrix S_W and the **between-class scatter** matrix S_B as follows

$$S_W = \sum_{j=1}^{c}\sum_{i=1}^{n_j}(x^{ij}-\mu^j)(x^{ij}-\mu^j)^T, \tag{1}$$

$$S_B = \sum_{j=1}^{c} n_j(\mu^j-\mu)(\mu^j-\mu)^T. \tag{2}$$

LDA computes an orthogonal matrix $V \in \mathbb{R}^{m\times d}$, where d is the desired dimension of the reduced space often chosen to be much smaller than m, to maximize the following ratio:

$$\max_V J(V) = \frac{Tr(V^T S_B V)}{Tr(V^T S_W V)}, \tag{3}$$

where Tr is the trace of a matrix. We denote the optimizer as V^* and the optimal value of the objective function as

$$J^* = J(V^*). \tag{4}$$

The optimizer consists of the set of eigenvectors $v_i \in \mathbb{R}^m$ of the generalized eigenvalue problem

$$S_B v_i = \lambda_i S_W v_i, \tag{5}$$

associated with the largest d eigenvalues. More precisely, we have

$$V^* = [v_1, v_2, \dots, v_d] \in \mathbb{R}^{m\times d}.$$

Let $N = [n_1, n_2, \dots, n_c] \in \mathbb{Z}^c$ be a vector of integers denoting the sizes of all classes. We describe the LDA algorithm in Algorithm 1.

Algorithm 1 The LDA algorithm.

```
Input $X \in \mathbb{R}^{m\times n^c}$, $N \in \mathbb{Z}^c$, $c$, $d$
    1. Compute the means $\mu$ and $\mu_j$, $j = 1, \dots, c$
    2. Form the within-class matrix $S_W$
    3. Form the between-class matrix $S_B$
    4. Solve the generalized eigenvalue problem for $d$ largest eigenvalues and the corresponding eigenvectors $v_1, \dots, v_d$
    5. Project the samples to the lower dimensional space $\mathbb{R}^d$
Output $V = LDA(X, N, c, d) \in \mathbb{R}^{m\times d}$, $Y = V^T X \in \mathbb{R}^{d\times n^c}$
```

4. Domain decomposed linear discriminant analysis and an optimality theory

As mentioned earlier in the paper, LDA works well when the number of classes is small. When the number of classes is large, a natural idea is to divide the training set into smaller subsets and conduct the training for the subsets separately. The question is how to combine the projection operators and use them to obtain more accurate testing results. In this section, we first introduce a domain decomposed linear discriminant analysis (ddLDA). Recall that X is the original dataset with c classes. In ddLDA, we define p as the total number of partitions and $k = 1, 2, \dots, p$ as the index for the kth subdomain. For each k, define s_k as the subclasses in domain k. To decompose the dataset into p subdomains ($p \le c$), we denote an index set by

$$s = \{1, 2, \dots, c\} = \{s_1, \dots, s_p\}$$

where each s_k is a subset of s such that $\cup_{k=1}^p s_k = s$, and $|s_k|$ denotes the number of classes of the kth subdomain. For example, if $c = 10$ and $p = 3$, then $s = \{1, 2, \dots, 10\}$ and one possible decomposition is $s_1 = \{1, 2, 3\}$, $s_2 = \{4, 5, 6\}$, $s_3 = \{7, 8, 9, 10\}$. Note that there are multiple ways to decompose the same dataset. A subdomain is a union of the classes defined by s_k and

$$\mathcal{X}_k = \{X^i \mid i \in s_k\} \in \mathbb{R}^{m\times \sum_{i\in s_k} n_i}.$$

The subdomains do not overlap. Denote d_k as the desired dimension of the kth reduced space that can be chosen differently for different subdomains, in practice, one often uses some small values, such as $d_k = 2$ or 3.

In the training part of ddLDA, we apply the traditional LDA algorithm (Algorithm 1) to produce a projection matrix $\mathcal{V}_k$ in each subdomain defined by s_k with selected d_k. Since the subdomains are independent of each other, this part of the computation can be carried out in parallel. Note that compared with the traditional LDA, ddLDA offers a tremendously important option of additional parallelism which is needed for large problems running on large scale parallel computers. For each reduced space, we define

$$\mathcal{Y}_k = \mathcal{V}_k^T \mathcal{X}_k \in \mathbb{R}^{d_k\times \sum_{i\in s_k} n_i}$$

as the projection of the training samples in the subdomain defined by s_k. The training part of ddLDA is summarized in Algorithm 2.

Algorithm 2 The ddLDA training algorithm.

```
Input: $X \in \mathbb{R}^{m\times n^c}$, $N \in \mathbb{Z}^c$, $c$ , and a decomposition $s = \{s_1, \dots, s_p\}$
    1. Decompose $X$ into $\mathcal{X}_1, \dots, \mathcal{X}_p$
    2. Select $d_k$ for each $\mathcal{X}_k$
    3. Apply $LDA(\mathcal{X}_k, s_k, |s_k|, d_k)$ to produce $\mathcal{V}_k$
Output: $\mathcal{V}_k$ and $\mathcal{Y}_k = \mathcal{V}_k^T \mathcal{X}_k$, $k = 1, \dots, p$
```

In the testing part of ddLDA, we denote $T = \{z^1, \dots, z^L\}$ as a test dataset, and for $z^i \in T$, define

$$C(z^i) = \{\text{the class(es) it belongs to}\}.$$

For each $\mathcal{V}_k$, we define

$$z_k^i = \mathcal{V}_k^T z^i, \text{ where } i = 1, \dots, L \text{ and } k = 1, \dots p.$$

Recall μ^j ($j = 1, \dots c$) is the mean of X^j. For each z_k^i, we define a scaled distance to the center of all c classes

$$\eta_{k,j}^i = \frac{1}{\sqrt{d_k}} \|z_k^i - \mu^j\|_2, \quad j \in s_k \tag{6}$$

and for each i, we define

$$\eta^i = \{\eta_{k,j}^i, \ k = 1, \dots, p \text{ and } j \in s_k\} \tag{7}$$

as the *distance vector* of z^i. Note that the scaling factor $1/\sqrt{d_k}$ in (6) is important; without the factor, the distances between different subdomains calculated with different d_k would not be comparable. To decide the class that the test sample belongs to, we solve the following minimization problem

$$arg \min_{k,j} \ \eta_{k,j}^i. \tag{8}$$

If η_{k^*,j^*}^i is the minimal value, then j^* is the class that z^i belongs to, then we put j^* in $C(z^i)$. The testing part of ddLDA is summarized in Algorithm 3. The dimension of the optimization problem is the same as the number of classes which is also the dimension of the distance vector (7). The number of minimal solutions may or may not be unique.

After the testing step, we compute the success rate defined as the ratio of the number of correctly classified samples to the total number of samples in the testing dataset.

We remark that the ddLDA is different from the traditional LDA in several aspects. In LDA, a single projection matrix is produced and each sample in the training set is then projected to the low dimension space with this projection. In the ddLDA, depending on the number of subdomains we choose, we generate p projections matrices, and the samples in the training set are projected into low dimensional spaces subdomain-by-subdomain. The total number of samples are the same and the total number of low dimensional projected vectors are also the same, but the ways they are computed are different. The two important variables in ddLDA are the decomposition of the training dataset and the selection of the dimension of the reduced subspaces. We will show their impact with numerical experiments.

Algorithm 3 The ddLDA testing algorithm.

```
Input: The testing set $T$
for $i = 1, \dots L$ do
    for $k = 1, \dots, p$ do
        $z_k^i = \mathcal{V}_k^T z^i$
    end for
    Calculate the distance vector $\eta^i$
    Solve the minimization problem (8)
    if $\eta_{k^*,j^*}^i$ is the minimal value, then put $j^*$ in $C(z^i)$
end for
Output: $C(z^1), \dots, C(z^L)$
```

Next, we present some analysis of the ddLDA algorithm for the situation that

$$d_1 = \cdots = d_p.$$

Similar to the traditional LDA, the **within-class** scatter matrix in the kth subdomain is defined as

$$S_W^k = \sum_{j\in s_k} \sum_{i=1}^{n_j} (x^{ij} - \mu^j)(x^{ij} - \mu^j)^T, \tag{9}$$

and the **between-class** scatter matrix in the kth subdomain is defined as

$$S_B^k = \sum_{j\in s_k} n_j(\mu^j - \mu^{(k)})(\mu^j - \mu^{(k)})^T, \tag{10}$$

where $\mu^{(k)}$ is the mean of all samples in the kth subdomain in the ddLDA. Define $n^{(k)}$ as the number of samples in the kth subdomain. We have the following properties:

$$n^c\mu = \sum_{k=1}^{p} n^{(k)}\mu^{(k)} = \sum_{j=1}^{c} n_j\mu^j. \tag{11}$$

ddLDA computes an orthogonal matrix $V_k \in \mathbb{R}^{m\times d_k}$ to maximize the following ratio:

$$J^k(V_k) = \frac{Tr(V_k^T S_B^k V_k)}{Tr(V_k^T S_W^k V_k)}. \tag{12}$$

The optimizer consists of the set of eigenvectors $v_i^k \in \mathbb{R}^m$ of the generalized eigenvalue problem in the kth subdomain

$$S_B^k v_i^k = \lambda_i S_W^k v_i^k, \tag{13}$$

associated with the largest d_k eigenvalues.

For each k, the optimizer V_k^* satisfies

$$J^k(V_k^*) \geq J^k(V), \tag{14}$$

for any $V \in \mathbb{R}^{m\times d}$. We denote the value of the scaled sum of the subdomain objective functions

$$J_{dd}^* = \sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\max_{V_k^T V_k = I} J^k(V_k) = \sum_{k=1}^{p}\frac{n^c}{n^{(k)}}J^k(V_k^*).$$

We remark that the factor $\frac{n^c}{n^{(k)}}$ is important since the number of samples in each subdomain is different.

To compare the optimal value J_{dd}^* from the ddLDA and the optimal value J^* from the traditional LDA, we need to find the connections between the ddLDA and the LDA. By definition, we can rewrite S_W as

$$S_W = \sum_{k=1}^{p} S_W^k,$$

and S_B as

$$S_B = \sum_{j=1}^{c} n_j(\mu^j - \mu)(\mu^j - \mu)^T = \sum_{k=1}^{p}\sum_{j\in s_k} n_j(\mu^j - \mu)(\mu^j - \mu)^T.$$

Then, we show that the maximum of the objective function J^* in the traditional LDA is smaller than J_{dd}^* obtained from the ddLDA. In the remainder of this section, we first recall some mathematical facts, then we show some lemmas useful in the proof of the main theorem.

Titu's Lemma:

$$\sum_i \frac{a_i}{b_i} \geq \frac{\sum_i a_i}{\sum_i b_i}. \tag{15}$$

for $a_i \geq 0$ and $b_i > 0$.

There are several names commonly associated with this inequality and its proof is elementary.

Lemma 1. *For any function $f_k(x) : \mathbb{R}^m \to \mathbb{R}$, $1 \leq k \leq p$,*

$$\sum_{k=1}^{p}\max_{x_k} f_k(x_k) \geq \max_{x_1,\ldots,x_p}\sum_{k=1}^{p} f_k(x_k). \tag{16}$$

The proof is trivial.

Because for any $x, y \in \mathbb{R}^m$, $Tr(xy^T) = Tr(yx^T) = x^T y$, we have

$$Tr(\mu^j\mu^{(k)T}) = Tr(\mu^{(k)}\mu^{jT}) = \mu^{jT}\mu^{(k)}, \tag{17}$$

$$Tr(\mu^j\mu^T) = Tr(\mu\mu^{jT}) = \mu^{jT}\mu. \tag{18}$$

With these facts, we present several technical lemmas.

Lemma 2. *For any d_k, $k = 1, \ldots, p$, we have*

$$\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\max_{V_k}\frac{Tr(V_k^T S_B^k V_k)}{Tr(V_k^T S_W^k V_k)} \geq \max_{V_1,\ldots,V_p}\frac{Tr(\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}V_k^T S_B^k V_k)}{Tr(\sum_{k=1}^{p}V_k^T S_W^k V_k)}.$$

Proof. Let $f_k(V_k) = \frac{n^c}{n^{(k)}}\frac{Tr(V_k^T S_B^k V_k)}{Tr(V_k^T S_W^k V_k)}$, then by (16), we have

$$\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\max_{V_k}\frac{Tr(V_k^T S_B^k V_k)}{Tr(V_k^T S_W^k V_k)} \geq \max_{V_1,\ldots,V_p}\frac{Tr(\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}V_k^T S_B^k V_k)}{Tr(\sum_{k=1}^{p}V_k^T S_W^k V_k)}.$$

Denote $a_k = Tr\left(\frac{n^c}{n^{(k)}}V_k^T S_B^k V_k\right)$, $b_k = Tr(V_k^T S_W^k V_k)$, then following **Titu's Lemma**, we have the desired estimate.

Lemma 3. *Assume $d_1 = \cdots = d_p = d$, for any $V \in \mathbb{R}^{m\times d}$ such that $V^T V = I$,*

$$Tr(V^T S_B V) \leq Tr\left(\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}V^T S_B^k V\right).$$

Proof. By definition,

$$\begin{aligned}Tr(V^T S_B V) &= Tr\left(\sum_{j=1}^{c} n_j V^T(\mu^j - \mu)(\mu^j - \mu)^T V\right)\\ &= \sum_{j=1}^{c} n_j Tr(V^T\mu^j\mu^{jT}V) - \sum_{j=1}^{c} n_j Tr(V^T\mu^j\mu^T V)\\ &\quad - \sum_{j=1}^{c} n_j Tr(V^T\mu\mu^{jT}V) + \sum_{j=1}^{c} n_j Tr(V^T\mu\mu^T V).\end{aligned}$$

To simplify the notations, we denote $\tilde{\mu} = V^T\mu$ and $\tilde{\mu}^j = V^T\mu^j$, then the above expression takes the following form

$$\begin{aligned}&\sum_{j=1}^{c} n_j\|\tilde{\mu}^j\|_2^2 - \sum_{j=1}^{c} n_j\tilde{\mu}^{jT}\tilde{\mu} - \sum_{j=1}^{c} n_j\tilde{\mu}^T\tilde{\mu}^j + \sum_{j=1}^{c} n_j\|\tilde{\mu}\|_2^2\\ &= \sum_{j=1}^{c} n_j\|\tilde{\mu}^j\|_2^2 - 2\sum_{j=1}^{c} n_j\tilde{\mu}^{jT}\tilde{\mu} + \sum_{j=1}^{c} n_j\|\tilde{\mu}\|_2^2\\ &= \sum_{k=1}^{p}\sum_{j\in s_k} n_j\|\tilde{\mu}^j\|_2^2 - n^c\|\tilde{\mu}\|_2^2.\end{aligned}$$

The last step is obtained using the fact that $\sum_{j=1}^{c} n_j = n^c$ and $\sum_{j=1}^{c} n_j\tilde{\mu}^j = n^c\tilde{\mu}$.

Similarly, by definition,

$$\begin{aligned}Tr\left(\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}V^T S_B^k V\right) &= Tr\left(\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\sum_{j\in s_k} n_j V^T(\mu^j - \mu^{(k)})(\mu^j - \mu^{(k)})^T V\right)\\ &= \sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\sum_{j\in s_k} n_j Tr(V^T(\mu^j - \mu^{(k)})(\mu^j - \mu^{(k)})^T V).\end{aligned}$$

Denote $\tilde{\mu}^{(k)} = V^T\mu^{(k)}$, by (17) and (18) and using the fact that $\sum_{j\in s_k} n_j = n^{(k)}$ and $\sum_{k=1}^{p} n^{(k)}\tilde{\mu}^{(k)} = \sum_{j=1}^{c} n_j\tilde{\mu}^j$, the above expression can be written as

$$\begin{aligned}&\sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\sum_{j\in s_k} n_j Tr(\tilde{\mu}^j\tilde{\mu}^{jT}) - \sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\sum_{j\in s_k} n_j Tr(\tilde{\mu}^j\tilde{\mu}^{(k)T})\\ &\quad - \sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\sum_{j\in s_k} n_j Tr(\tilde{\mu}^{(k)}\tilde{\mu}^{jT}) + \sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\sum_{j\in s_k} n_j Tr(\tilde{\mu}^{(k)}\tilde{\mu}^{(k)T})\\ &= \sum_{k=1}^{p}\frac{n^c}{n^{(k)}}\sum_{j\in s_k} n_j\|\tilde{\mu}_j\|_2^2 - 2\sum_{k=1}^{p}\sum_{j\in s_k}\frac{n^c}{n^{(k)}}n_j\tilde{\mu}^{jT}\tilde{\mu}^{(k)}\\ &\quad + \sum_{k=1}^{p}\sum_{j\in s_k}\frac{n^c}{n^{(k)}}n_j\|\tilde{\mu}^{(k)}\|_2^2\end{aligned}$$

$$= \sum_{k=1}^{p} \frac{n^c}{n^{(k)}} \sum_{j\in s_k} n_j \|\tilde{\mu}_j\|_2^2 + \sum_{k=1}^{p} n^{(k)} \|\tilde{\mu}^{(k)}\|_2^2 - \sum_{k=1}^{p} \left(1 - \frac{n^c}{n^{(k)}}\right) n^{(k)} \|\tilde{\mu}^{(k)}\|_2^2.$$

Thus,

$$Tr(V^T S_B V) - Tr\left(\sum_{k=1}^{p} \frac{n^c}{n^{(k)}} V^T S_B^k V\right) = \sum_{k=1}^{p} \left(1 - \frac{n^c}{n^{(k)}}\right) \sum_{j\in s_k} n_j \|\tilde{\mu}_j\|_2^2$$
$$+ \sum_{k=1}^{p} \left(1 - \frac{n^c}{n^{(k)}}\right) n^{(k)} \|\tilde{\mu}^{(k)}\|_2^2 - \sum_{k=1}^{p} n^{(k)} \|\tilde{\mu}^{(k)}\|_2^2 - n^c \|\tilde{\mu}\|_2^2. \tag{19}$$

Because $1 - n^c/n^{(k)} < 0$, all the coefficients of the right-hand side in (19) are negative. Therefore, we have the proof.

Lemma 4. *Assume $d_1 = \cdots = d_p = d$, for any $V \in \mathbb{R}^{m\times d}$ such that $V^T V = I$,*

$$\sum_{k=1}^{p} \frac{n^c}{n^{(k)}} \frac{Tr(V^T S_B^k V)}{Tr(V^T S_W^k V)} \geq \frac{Tr(V^T S_B V)}{Tr(V^T S_W V)}.$$

Proof. Let $a_k = Tr\left(\frac{n^c}{n^{(k)}} V^T S_B^k V\right)$, $b_k = Tr(V^T S_W^k V)$, by **Titu's Lemma**,

$$\sum_{k=1}^{p} \frac{Tr\left(\frac{n^c}{n^{(k)}} V^T S_B^k V\right)}{Tr(V^T S_W^k V)} \geq \frac{\sum_{k=1}^{p} Tr\left(\frac{n^c}{n^{(k)}} V^T S_B^k V\right)}{\sum_{k=1}^{p} Tr(V^T S_W^k V)}$$
$$= \frac{\sum_{k=1}^{p} Tr\left(\frac{n^c}{n^{(k)}} V^T S_B^k V\right)}{Tr(V^T S_W V)} \geq \frac{Tr(V^T S_B V)}{Tr(V^T S_W V)}.$$

The last inequality follows from Lemma 3.

With the technical lemmas mentioned above, we have the following theorem.

Theorem 1. *When $d_1 = \cdots = d_p = d$,*

$$J_{dd}^* \geq J^*. \tag{20}$$

Proof. For any $V \in \mathbb{R}^{m\times d}$, by Lemma 4,

$$\sum_{k=1}^{p} \frac{n^c}{n^{(k)}} \frac{Tr(V^T S_B^k V)}{Tr(V^T S_W^k V)} \geq \frac{Tr(V^T S_B V)}{Tr(V^T S_W V)}.$$

Especially, for V^* defined in (4), that is

$$\sum_{k=1}^{p} \frac{n^c}{n^{(k)}} \frac{Tr(V^{*T} S_B^k V^*)}{Tr(V^{*T} S_W^k V^*)} \geq \frac{Tr(V^{*T} S_B V^*)}{Tr(V^{*T} S_W V^*)}. \tag{21}$$

By the definition of V_k^*, $k = 1, \ldots, p$, in (14), and taking V as V^*, we have

$$\frac{Tr(V_k^{*T} S_B^k V_k^*)}{Tr(V_k^{*T} S_W^k V_k^*)} \geq \frac{Tr(V^{*T} S_B^k V^*)}{Tr(V^{*T} S_W^k V^*)}. \tag{22}$$

Multiplying both sides of (22) by $n^c/n^{(k)}$ and taking the sum for $k = 1, \ldots, p$, and combining with estimate (21), we arrive at the desired estimate (20).

We remark that the theorem indicates that the ddLDA is better than the traditional LDA for $p > 1$. When $p = 1$, they are obliviously the same. Note that the proof is valid only when $d_1 = \cdots = d_p = d$, even though we think the conclusion holds for any d_i. In the numerical experiments, we use different values of d_i for different subdomains.

The other important feature of ddLDA is that it is well-suited for distributed computing. In other words, on a parallel computing system, we can map different subdomains to different processors and they can all be computed in parallel. In such an implementation, ddLDA is not only more accurate but also faster.

The computational complexity of ddLDA in the kth subdomain is $O(|s_k|m^2)$. Since $\frac{1}{p} \approx \frac{|s_k|}{c}$ in the practical applications, the complexity in the kth subdomain is $\frac{1}{p}$ of the complexity of the traditional LDA.

In short, ddLDA is capable of dealing with datasets with a large number of classes, and it is naturally parallel so that it can be implemented on computers with many processors. However, we note that the optimal partition is not always easy to determine.

5. Iteratively deflated linear discriminant analysis

Motivated by the fact that the traditional LDA works quite well for problems with a small number of classes, in this section, we introduce an iteratively deflated LDA (idLDA) method, formulated by the deflation techniques for eigenvalue problems [17], in which we iteratively remove the well-separated classes during the training phase of the algorithm.

We start with some notations. Let $D^{(0)}$ be the original dataset with $c^{(0)}$ classes. Denote $n_j^{(0)}$ as the number of samples in the jth class in $D^{(0)}$ and $n^{c^{(0)}} = \sum_{j=1}^{c^{(0)}} n_j^{(0)}$ is the total number of samples in $D^{(0)}$. Let $N^{(0)} = [n_1^{(0)}, n_2^{(0)}, \ldots, n_c^{(0)}] \in \mathbb{Z}^{c^{(0)}}$ be a vector of integers indicating the sizes of all classes in $D^{(0)}$. Let m be the dimension of the samples in the dataset and $x^{ij(0)} \in \mathbb{R}^m$ denote the ith sample from the jth class in the $D^{(0)}$, then $X^{j(0)} = [x^{11(0)}, \ldots, x^{n_j^{(0)} j(0)}] \in \mathbb{R}^{m\times n_j^{(0)}}$ is the set of all samples in the jth class in $D^{(0)}$. In other words, $X^{(0)} = [X^{1(0)}, X^{2(0)}, \ldots, X^{c^{(0)}(0)}] \in \mathbb{R}^{m\times n^{c^{(0)}}}$ is $D^{(0)}$.

The idLDA algorithm takes several iterates. At the kth iterate, suppose $D^{(k)}$ is the resulting dataset after the first $k-1$ steps of deflation, let $c^{(k)}$ be the total number of classes in $D^{(k)}$ and $n_j^{(k)}$ be the number of samples in the jth class, we define $n^{c^{(k)}} = \sum_{j=1}^{c^{(k)}} n_j^{(k)}$ as the total number of samples in $D^{(k)}$. We assume that

$$D^{(k)} \subset D^{(k-1)} \subset \cdots \subset D^{(0)},$$

and we denote by $X^{j(k)} = [x^{1j(k)}, \ldots, x^{n_j^{(k)} j(k)}] \in \mathbb{R}^{m\times n^{c^{(k)}}}$ as the jth class of samples in $D^{(k)}$ and $X^{(k)} = [X^{1(k)}, X^{2(k)}, \ldots, X^{c^{(k)}(k)}] \in \mathbb{R}^{m\times n^{c^{(k)}}}$ in $D^{(k)}$. In the reduced space, let us denote $y^{ij} = V^T x^{ij}$ as the projection of the sample into the reduced space and $Y^j = [y^{1j}, \ldots, y^{n_j j}] \in \mathbb{R}^{d\times n_j}$ is the jth class of samples in the reduced space. Also we define $Y = [Y^1, \ldots, Y^c] \in \mathbb{R}^{d\times n^c}$ as the sample matrix in the reduced space. We define $Y^{j(k)} = [y^{1j(k)}, \ldots, y^{n_j^{(k)} j(k)}] \in \mathbb{R}^{d\times n^{c^{(k)}}}$ is the jth class of samples and $Y^{(k)} = [Y^{1(k)}, Y^{2(k)}, \ldots, Y^{c^{(k)}(k)}] \in \mathbb{R}^{d\times n^{c^{(k)}}}$ is the sample matrix in the reduced space.

Next, we discuss the classification in the reduced space $\mathbb{R}^{d,n^c}$. To determine whether the classes are well-separated, we introduce several metrics. Let $v^j \in \mathbb{R}^d$ denote the mean of the jth class in the reduced space,

$$v^j = \frac{1}{n_j} \sum_{i=1}^{n_j} y^{ij}.$$

We define the radius of the jth class as

$$R^j = \max_{y^{ij}\in Y^j} \{\|v^j - y^{ij}\|_2\}.$$

To remove the impact of the outliers in the training phase, for each Y^j, we define, for example, the 75% training quantile set

$$Q_{TR}^j = \{y^{ij} \in Y^j \;\;|\;\; \|v^j - y^{ij}\|_2 \leq 75\% R^j\}$$

and its radius as

$$R_{TR}^j = \max_{y^{ij}\in Q_{TR}^j} \{\|v^j - y^{ij}\|_2\}.$$

Similarly, for the testing phase, we define, for example, the 95% testing quantile set

$$Q_{TE}^j = \{y^{ij} \in Y^j \;\;|\;\; \|v^j - y^{ij}\|_2 \leq 95\% R^j\}$$

and the corresponding radius

$$R_{TE}^j = \max_{y^{ij}\in Q_{TE}^j} \{\|v^j - y^{ij}\|_2\}.$$

With these definitions, we introduce the class-separation criteria as follows. If the distance between v^j and v^i is larger than the sum of their radii, in other words,

$$\|v^j - v^i\|_2 > R^j_{TR} + R^i_{TR},$$

we say that the two classes j and k are well-separated. If

$$\|v^j - v^i\|_2 \leq R^j_{TR} + R^i_{TR},$$

we say that the two classes are not well-separated. If a class is well-separated from all other classes, then we say this class is separable.

In the above definitions, we specify the quantile values to be 75% for training and 95% for testing. The best choices of these values are application dependent and should be selected carefully. In the numerical experiment section of the paper, we examine different quantile values for both training and testing and compare the classification results.

For the class-separation criteria, we define a distance matrix $M^{(k)} = \{M^{(k)}_{ij}\} \in \mathbb{R}^{c^{(k)} \times c^{(k)}}$ calculated at the kth iteration in the idLDA. Let $v^{i(k)}$ be the mean of the ith class in the reduced space at the kth iteration. Let

$$M^{(k)}_{ij} = \|v^{i(k)} - v^{j(k)}\|_2 \quad \text{if} \quad i \neq j;$$

$$M^{(k)}_{ii} = R^i_{TR} \quad \text{if} \quad i = j,$$

where R^i_{TR} is the training quantile radius of the ith class. If $M^{(k)}_{ii} < M^{(k)}_{ij}$, for all $j \neq i$, then we say that the ith class is well-separated from the other classes in $D^{(k)}$. Otherwise, the two classes are not separable.

The training part of idLDA is summarized in Algorithm 4.

Algorithm 4 The idLDA training algorithm.

```
Input: $X^{(0)} \in \mathbb{R}^{m \times n^{c^{(0)}}}$, $N^{(0)} \in \mathbb{Z}^{c^{(0)}}$, $c^{(0)}$, $d^{(0)}$
for $k = 0 \ldots$ do
  1. Compute $V^{(k)} = LDA(D^{(k)}, N^{(k)}, c^{(k)}, d^{(k)})$
  2. Compute $Y^{(k)} = V^{(k)T} X^{(k)} \in \mathbb{R}^{d \times n^{c^{(k)}}}$
  3. Compute the distance matrix $M^{(k)}$
  4. Deflate the well-separated class(es); if nothing to deflate, return

  5. Define $D^{(k+1)}, N^{(k+1)}, c^{(k+1)}, d^{(k+1)}$; $k = k + 1$, go to Step 1
end for
Output: $V^{(0)}, \cdots \in \mathbb{R}^{m \times d^{(k)}}$
```

We remark that the dimension of the reduced space can be different at different iterations, but in practice, one often uses the same value, such as $d^{(k)} = 2$. For the testing procedure, we denote

$$T = \{z^1, \ldots, z^L\} \in D^{(0)}$$

as a test set. For $z^i \in T$, define

$$C(z^i) = \{\text{the class(es) it belongs to}\}.$$

For each $V^{(k)}$, $z^i_k = (V^{(k)})^T z^i$, where $i = 1, \ldots, L$ and $k = 1, \ldots K$, where K is the number of deflation steps in the training phase of the algorithm and $v^{(k)}$ is the mean of $Y^{(k)}$. The testing part of idLDA is summarized in Algorithm 5. In the last step, we compare the predicted label to the true label. The success rate is defined as the ratio of the number of correctly predicted samples to the total number of samples in the testing dataset.

Because idLDA does not perform well when the dataset has a large number of classes that are not separable, it is often better to use ddLDA first to reduce the problem and then apply idLDA.

6. Numerical experiments

In this section, we present some numerical experiments for the proposed ddLDA and idLDA algorithms. Three datasets are used in the testing, the CIFAR-10/100 datasets with 60,000 images are used for the study of ddLDA, and a gene expression dataset with 5,629 cancer patient data is used for the study of idLDA.

Algorithm 5 The idLDA testing algorithm.

```
Input: The testing dataset $T$
for $i = 1, \ldots, L$ do
  for $k = 1, \ldots K$ do
    for $j = 1, \ldots, c^{(k)}$ do
      $z^i_k = (V^{(k)})^T z^i$
      if $||z^i_k - v^j_k||_2 < R^j_{TE}$ then
        include $j$ in $C(z^i)$
      end if
    end for
  end for
end for
Output: $C(z^1), \ldots, C(z^L)$
```

6.1. Experiments with ddLDA

6.1.1. Experiments with the CIFAR-10 dataset

In this subsection, we test the proposed ddLDA method using the CIFAR-10 dataset [30] consisting of 60,000 images in 10 classes including airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The color images are of size 32 × 32. The dataset is separated into a training dataset with 50,000 samples and a testing dataset with 10,000 samples.

In this experiment, we consider a decomposition of the dataset with $c = 10$ classes into $p = 3$ subdomains. More precisely, we choose subdomain 1 as {airplane, cat, ship}, subdomain 2 as {automobile, deer, truck}, and subdomain 3 as {bird, Dog, Frog, Horse}.

In the first test, we select $d_1 = d_2 = d_3 = 2$ and the quantile radius is set to be 95%. We project all training images into the two-dimensional space, subdomain-by-subdomain. In Fig. 1, each picture corresponds to a subdomain, and the circles are the 95% quantile radii. It is clear that the classes are well separated for subdomains 1 and 2, but not for subdomain 3. To further understand the situation, we take a sample image (a ship) represented by a green circle. It is correctly classified in subdomain 1, and when it is placed in subdomains 2 and 3, the image does not belong to any of the classes.

Following the ddLDA testing algorithm, we compute the distance vector for this sample, as shown below in (23), which has a minimum value of 0.0131 indicating that this particular sample belongs to the class of ships.

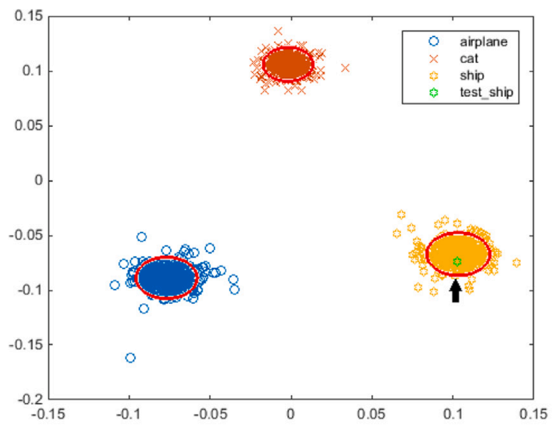
$$\begin{array}{c c c c c c c c c c c} & Airplane & Cat & Ship & Automobile & Deer & Ttruck & Bird & Dog & Frog & Horse \\ Ship & (\;0.1615 & 0.1923 & 0.0131 & 1.8267 & 2.0488 & 1.2713 & 1.0271 & 0.9493 & 2.2207 & 1.6399\;)_{1\times 10}. \end{array} \tag{23}$$

6.1.2. A wrongly classified case

In this subsection, we show a wrongly classified sample. (24) is the computed distance vector of a bird. The true label of the test sample is a bird, but the minimum value is 0.0740 which put it in the class of airplanes. Fig. 2 shows the figure of this test sample and the right figure is a sample of an airplane in the training dataset. The distance vector says that this bird sample is close to the classes of airplane, frog, and bird.

$$\begin{array}{c c c c c c c c c c c} & Airplane & Cat & Ship & Automobile & Deer & Ttruck & Bird & Dog & Frog & Horse \\ Bird & (\;0.0740 & 0.0964 & 0.1078 & 0.2876 & 0.1667 & 0.5540 & 0.0896 & 0.1573 & 0.0839 & 0.5314\;)_{1\times 10}. \end{array} \tag{24}$$

We now consider the whole testing dataset and compare the traditional LDA, ddLDA with different values of quantile radius. First of all, we find that ddLDA significantly improves the classification results when compared with LDA which has an accuracy of 47.4%. In Table 2, we show the classification results obtained with ddLDA with three different values of the training and testing QR vary from 85% to 95%, and the accuracies range from 84.6% to 93.9%. When the dimensions of



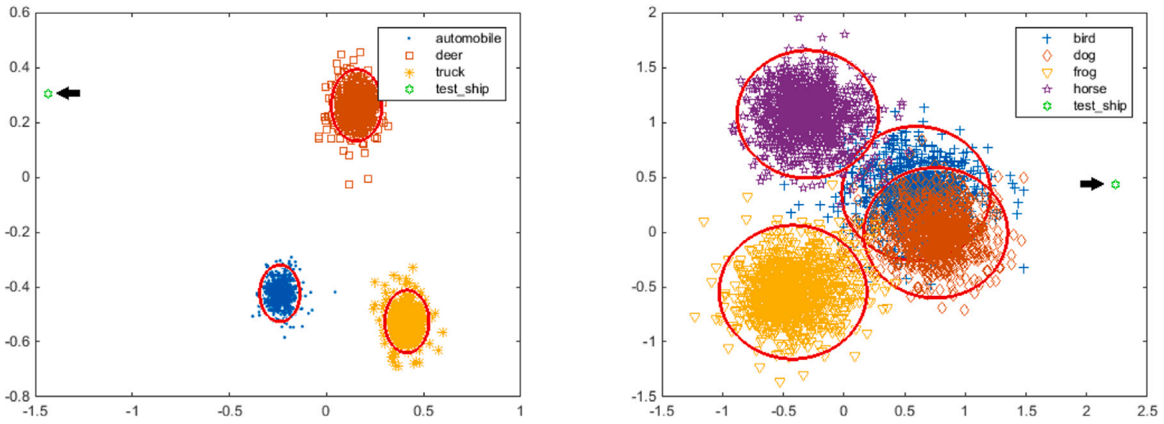


Fig. 1. The testing results using ddLDA. The number of subdomains is 3, and the corresponding dimensions of the reduced spaces are $d_1 = d_2 = d_3 = 2$. The quantile radius is 95%. The top figure is for subdomain 1 with 3 classes {airplane, cat, ship}. The lower left figure is for subdomain 2 with 3 classes {automobile, deer, truck}. The lower right figure is for subdomain 3 with 4 classes {bird, dog, frog, horse}. The green circle is a sample of the ship in the testing dataset. The horizontal axis corresponds to the largest singular value, and the vertical axis corresponds to the second largest singular value.

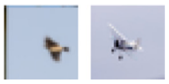


Fig. 2. The left figure is a test sample of a bird that is wrongly classified in the class of airplanes; the right figure is a sample of an airplane in the training dataset.

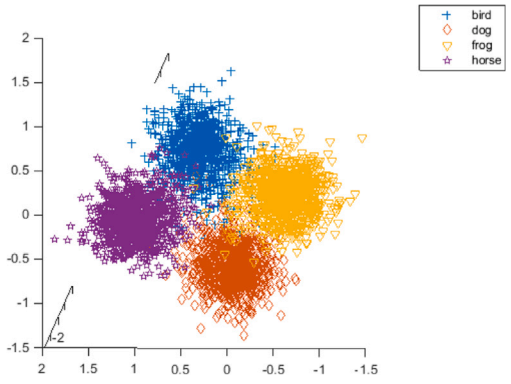


Fig. 3. This figure corresponds to the lower-right figure of Fig. 4 where $d_3 = 2$. For this figure, we increase d_3 to 3. The non-separable classes in two-dimensional space are now separable in three-dimensional space. The horizontal axis corresponds to the largest singular value, the vertical axis corresponds to the second largest singular value, and the third axis corresponds to the third largest singular value.

the reduced subspaces are all set to 2, the best result is 90.4% when the training and testing quantile radii are both set to 90%. As we mentioned before, we can choose different values of d_i in the ddLDA algorithm. As shown in the bottom right figure in Fig. 1, we find that the classes of bird and dog overlap when $d_3 = 2$. If we increase $d_3 = 3$, then the four classes in subdomain 3 are separately nicely, as shown in Fig. 3. This suggests that classes that cannot be separated cleanly in a low dimensional space ($d_3 = 2$), can be separated in a higher dimensional space ($d_3 = 3$). However, for subdomains 1 and 2, if we increase d_1 or d_2 to 3, the results do not improve. We conclude that d_i should be larger in subdomains with a larger number of classes. In the right panel of Table 2, we show the results when different subdomains use different values of d_i, the results are indeed better for all values of training and testing quantile radii. The best result is 93.9% when the quantile radii are 90% for both.

We also compare the results of ddLDA with PCA and ddPCA. Based on the traditional PCA, the testing accuracy on CIFAR-10 is 35.8% with 2 eigenmodes. ddPCA [16] does a little better at 39.3% with 2 eigenmodes with a 2 × 2 partition.

In [15], the same dataset was used for the study of several deep convolutional neural network approaches, and a slightly better accuracy was achieved at 94.76%. For practical applications, we feel that ddLDA and DCNN offer similar accuracy, and ddLDA has the advantage that it is a deterministic method and also has fewer parameters to adjust.

6.1.3. *Experiments with the CIFAR-100 dataset*

In this subsection, we test the proposed ddLDA method using the CIFAR-100 dataset [30] including 60,000 images in 100 classes for bed, bee, beetle, boy, girl, and oak tree, etc. The dataset is divided into a

Table 1
The table includes testing results for the traditional LDA, the ddLDA with $d_1 = d_2 = d_3 = 2$, and the ddLDA with $d_1 = d_2 = 2$, $d_3 = 3$. The dataset is CIFAR-10. QR_{TR} is the training quantile radius and QR_{TE} is the testing quantile radius.

LDA	ddLDA ($d_1 = d_2 = d_3 = 2$)				ddLDA ($d_1 = d_2 = 2, d_3 = 3$)		
	$QR_{TE}\backslash QR_{TR}$	85%	90%	95%	85%	90%	95%
47.4%	85%	84.6%	87.4%	86.3%	87.7%	91.6%	89.2%
	90%	87.8%	90.4%	88.2%	90.6%	93.9%	91.1%
	95%	85.5%	89.7%	87.7%	89.4%	91.4%	90.5%

Table 2
The table shows the testing accuracy for the ddLDA with $d_1 = \cdots = d_p = 2$. The dataset is CIFAR-100 and the number of partitions is p. QR_{TR} is the training quantile radius and QR_{TE} is the testing quantile radius.

p	ddLDA ($d_1 = \cdots = d_p = 2$)			
	$QR_{TE}\backslash QR_{TR}$	85%	90%	95%
2	85%	31.3%	34.3%	32.1%
	90%	32.5%	36.5%	33.2%
	95%	32.1%	36.3%	31.8%
4	85%	37.5%	39.7%	36.4%
	90%	38.5%	38.9%	36.2%
	95%	36.6%	36.8%	35.8%
10	85%	58.4%	61.2%	61.1%
	90%	62.5%	65.4%	63.2%
	95%	62.8%	62.9%	61.3%
25	85%	67.6%	69.4%	69.1%
	90%	69.2%	70.8%	69.8%
	95%	68.8%	70.9%	69.4%
33	85%	72.1%	73.4%	71.9%
	90%	74.2%	75.3%	72.8%
	95%	72.4%	74.2%	73.3%

Table 3
The table shows the testing accuracy for the data-specific ddLDA with $d_1 = \cdots = d_p = 2$. The dataset is CIFAR-100 and the number of partitions is p. QR_{TR} is the training quantile radius and QR_{TE} is the testing quantile radius.

p	data-specific ddLDA ($d_1 = \cdots = d_p = 2$)			
	$QR_{TE}\backslash QR_{TR}$	85%	90%	95%
2	85%	34.8%	36.3%	33.9%
	90%	35.7%	38.4%	34.2%
	95%	34.1%	35.2%	32.7%
4	85%	37.5%	39.7%	36.4%
	90%	38.5%	38.9%	36.2%
	95%	36.6%	36.8%	35.8%
10	85%	62.9%	63.8%	63.1%
	90%	67.3%	70.8%	68.7%
	95%	63.2%	64.1%	62.4%
25	85%	73.4%	75.2%	72.8%
	90%	75.1%	76.9%	73.4%
	95%	74.8%	75.9%	74.1%
33	85%	83.2%	84.5%	82.3%
	90%	87.6%	90.1%	87.7%
	95%	84.1%	85.3%	83.7%

training dataset with 50,000 samples and a testing dataset with 10,000 samples. In each class, there are 500 samples in the training dataset and 100 samples in the testing dataset.

In this experiment, we consider several decompositions of the dataset with $c = 100$ classes into $p = 2, 4, 10, 25, 33$ subdomains respectively. The subdomains are picked randomly in the ddLDA method. To evaluate the accuracy, we repeat the experiment 25 times and calculate the average accuracy. The testing accuracy is 21.7% in the traditional LDA. Table 2 shows the testing accuracy of ddLDA. Notably, the testing accuracy increases as more partitions are used in the ddLDA from 31.3% to 75.3%.

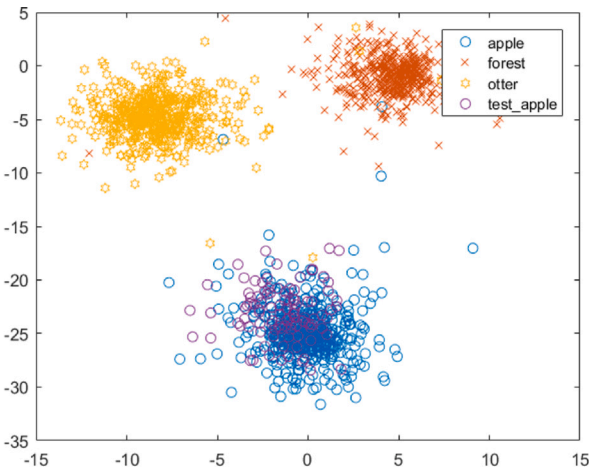


Fig. 4. The testing results using ddLDA. The purple circles are the test samples of apples projected onto the class of apples in the training dataset.

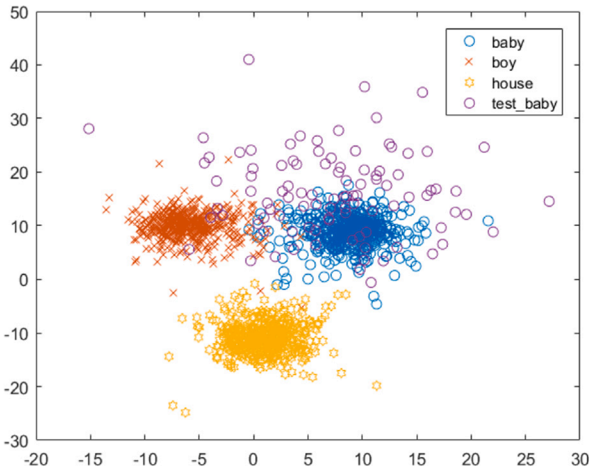


Fig. 5. The testing results using ddLDA. The purple circles are the test samples of babies projected onto the class of babies, the class of boys, and the class of houses in the training dataset. There are some wrongly classified cases for babies and boys.

We observe that the testing accuracy varies among classes. For instance, Fig. 4 shows the testing accuracy is 100% for the class of apples with a $p = 33$ partition. However, for some classes, the testing accuracy is low. For instance, Fig. 5 shows the testing accuracy is only 72% for the class of babies.

We also observe that the classification error increases when images in some of the classes look similar, for example, the classes of babies and boys share common features that are difficult to distinguish in the testing. To avoid this issue, we introduce the **data-specific** ddLDA method. Instead of selecting the classes randomly in each subdomain, we put the classes with common features in different subdomains in the training phase of ddLDA. For example, if we separate the classes of babies and boys into different subdomains {baby, bicycle, house} and {boy, cloud, lizard}, the testing accuracy can be improved significantly; see Figs. 6–7. Table 3 shows the testing accuracy of the data-specific ddLDA method when the dimensions of the reduced subspaces are all set to 2. The best result is 90.1% when the training and testing quantile radii are both set to 90%.

In [15], the CIFAR-100 dataset was also used to study some deep convolutional neural network methods, and the accuracy was achieved at 74.5%. ddLDA offers similar accuracy to DCNN when using $p = 33$ partitions. Moreover, when the specific-data ddLDA is applied, higher accuracy is achieved.

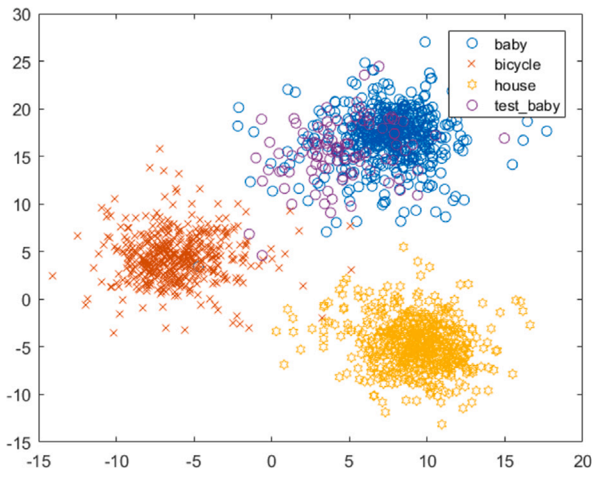


Fig. 6. The testing results using data-specific ddLDA. The purple circles are the test samples of babies projected onto the class of babies in the training dataset.

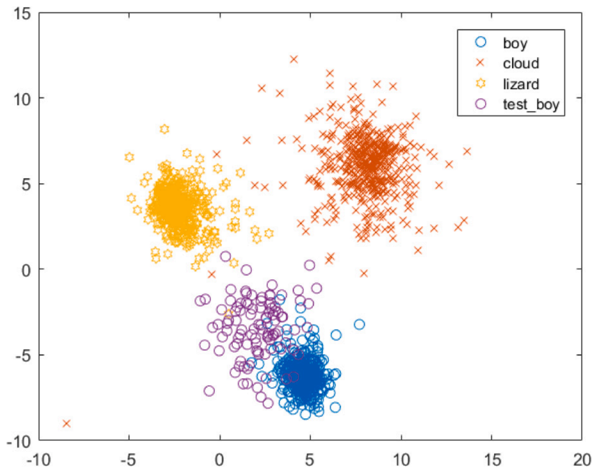


Fig. 7. The testing results using data-specific ddLDA. The purple circles are the test samples of boys projected onto the class of boys in the training dataset.

Finally, we show the compute time for the traditional LDA and the ddLDA with different sizes of partitions, and each subdomain is computed on a different processor. Table 4 shows the compute time spent on the training phase of the algorithm using different partitions for the CIFAR-100 dataset. Let t_p be the compute time using p processors. If the algorithm and the implementation satisfy the relation:

$$\frac{t_p}{t_q} \approx \frac{q}{p}, \text{ for any } p \text{ and } q,$$

where t_p and t_q are the compute times using p and q processors, respectively, then we claim that the algorithm is linearly scalable. Table 4 shows that ddLDA is close to be linearly scalable. We also record the compute time for the CIFAR-10 dataset. LDA takes 342.41 s using a single processor and ddLDA takes 163.93 s using 2 processors and 93.3 s using 3 processors. Roughly speaking, the compute time per processor decreases almost linearly as we increase the number of subdomains.

6.2. Experiments with idLDA

In this subsection, we test the proposed idLDA method using the gene dataset [31] consisting of the genetic data of 5,629 patients and 11 classes of diseases including liver cancer, cell line cancer, breast cancer, colon cancer, kidney cancer, uterus cancer, ovary cancer, pancreatic stellate cell line cancer (PSC), lung cancer, and heart disease. Each sample is represented as a vector with 18,441 double precision numbers. The dataset is randomly separated into a training dataset with 4,220 samples and a testing dataset with 1,409 samples.

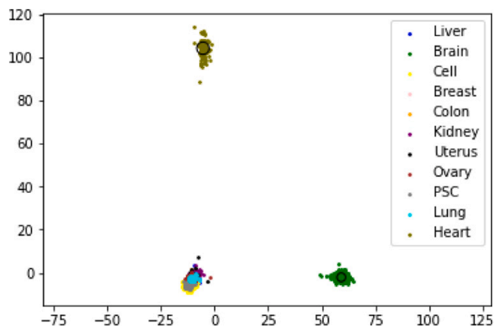


Fig. 8. Classification by the traditional LDA with $d = 2$. Two separable classes are determined (brain and heart). The horizontal axis corresponds to the largest singular value, the vertical axis corresponds to the second largest singular value.

Table 4
The table shows the training time (in seconds) for the CIFAR-100 dataset using the traditional LDA with $p = 1$ processor and ddLDA with the different number of processors (p) with $d_1 = \cdots = d_p = 2$.

LDA	ddLDA					
$p = 1$	$p = 2$	$p = 3$	$p = 4$	$p = 10$	$p = 25$	$p = 33$
369.38	167.03	132.23	84.59	41.08	33.36	29.12

We first present the classification results obtained using the traditional LDA algorithm when the dimension of the reduced space is $d = 2$. In Fig. 8 the horizontal axis corresponds to the largest singular value, and the vertical axis corresponds to the second largest singular value. One can tell that there are 3 clusters in the figure; one for the heart disease at the top of the figure, one for the brain cancer at the bottom right of the figure, and all the other classes are in the same cluster on the bottom left of the figure. It is clear that the traditional LDA is not able to separate all the classes in this experiment.

Based on $V^{(0)}$ and $Y^{(0)}$ from the LDA step, we can also form a distance matrix as given in Box I (see Box I). The diagonal entries $M^{(0)}_{2,2}$ and $M^{(0)}_{11,11}$ are strictly smaller than the corresponding values in the 2nd row (column) and 11th row (column). Following the definition of separable classes, these two classes are separable from other classes. For all the other rows (columns), the diagonal entry is not the smallest, therefore they are not separable.

Note that for this experiment, the quantile radius for the training phase is 75% and 90% for the testing phase. Different values of the quantile radius will be studied later in Table 1.

Based on the result from the traditional LDA, two classes are well-separated, corresponding to $j = 2, 11$; i.e., brain and heart. We delete these two classes from the dataset $D^{(0)}$ to form $D^{(1)}$, then we apply a second iteration of LDA using the same values of d and quantile radius. The corresponding distance matrix is shown in Box II (see Box II). By the class-separation criteria, in this step, only one class is well-separated, corresponding to $j = 1$; i.e., liver as shown in the top left figure of Fig. 9. We then delete this class from the dataset $D^{(1)}$ to form $D^{(2)}$ for which we perform another round of LDA.

After the third iteration of LDA, we compute the distance matrix as given in Box III (see Box III). By checking the matrix values, two classes are well-separated, corresponding to $j = 2, 4$; i.e., breast and kidney as shown in the top right figure of Fig. 9. We delete these two classes from the dataset $D^{(2)}$ to form $D^{(3)}$. Then we go through the fourth iteration

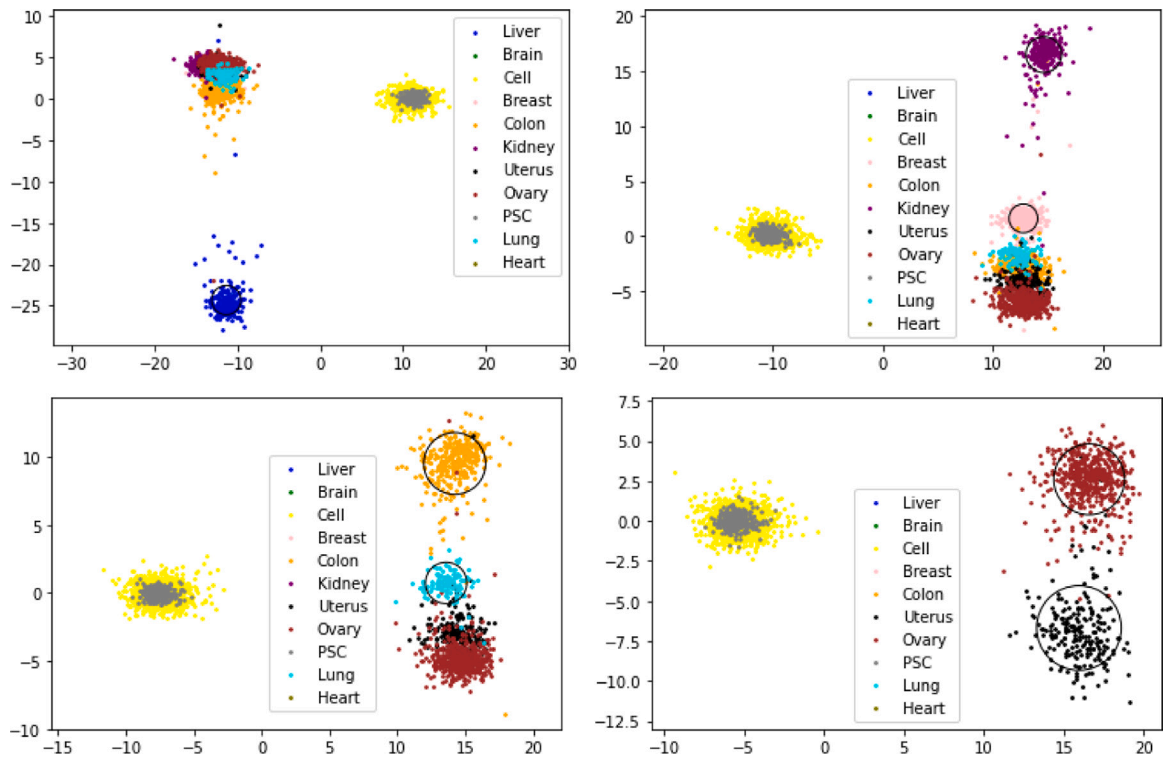


Fig. 9. The classification results using the idLDA with 4 iterations (after one step of the traditional LDA). The top left figure shows one separable class (liver) at $k = 1$; the top right figure shows two separable classes (breast and kidney) at $k = 2$; the bottom left figure shows two separable classes (colon and lung) at $k = 3$; the bottom right figure shows two separable classes (uterus and ovary) at $k = 4$. The circles are centered at the center of the separable classes and the radius is the 95% quantile radius.

$$M^{(0)} = \begin{array}{c|ccccccccccc} & Liver & Brain & Cell & Breast & Colon & Kidney & Uterus & Ovary & PSC & Lung & Heart \\ \hline Liver & 1.44 & 68.85 & 3.69 & 1.11 & 1.39 & 2.25 & 1.09 & 1.18 & 3.72 & 0.53 & 107.56 \\ Brain & 68.85 & 2.02 & 71.25 & 69.57 & 70.22 & 68.02 & 69.59 & 69.94 & 71.28 & 69.19 & 124.56 \\ Cell & 3.69 & 71.25 & 1.32 & 4.03 & 3.22 & 5.89 & 3.99 & 3.54 & 0.04 & 3.82 & 110.55 \\ Breast & 1.11 & 69.57 & 4.03 & 1.28 & 0.90 & 2.00 & 0.04 & 0.53 & 4.07 & 0.58 & 106.76 \\ Colon & 1.39 & 70.22 & 3.22 & 0.90 & 1.36 & 2.90 & 0.86 & 0.37 & 3.25 & 1.04 & 107.42 \\ Kidney & 2.25 & 68.02 & 5.89 & 2.00 & 2.90 & 1.32 & 2.04 & 2.53 & 5.91 & 2.06 & 105.44 \\ Uterus & 1.09 & 69.59 & 3.99 & 0.04 & 0.86 & 2.04 & 1.41 & 0.49 & 4.02 & 0.56 & 106.80 \\ Ovary & 1.18 & 69.94 & 3.54 & 0.53 & 0.37 & 2.53 & 0.49 & 1.32 & 3.57 & 0.75 & 107.16 \\ PSC & 3.72 & 71.28 & 0.04 & 4.07 & 3.25 & 5.91 & 4.02 & 3.57 & 0.97 & 3.85 & 110.58 \\ Lung & 0.53 & 69.19 & 3.82 & 0.58 & 1.04 & 2.06 & 0.56 & 0.75 & 3.85 & 1.21 & 107.17 \\ Heart & 107.56 & 124.56 & 110.55 & 106.76 & 107.42 & 105.44 & 106.80 & 107.16 & 110.58 & 107.17 & 2.90 \end{array}_{11\times 11},$$

Box I.

$$M^{(1)} = \begin{array}{c|ccccccccc} & Liver & Cell & Breast & Colon & Kidney & Uterus & Ovary & PSC & Lung \\ \hline Liver & 1.77 & 33.24 & 28.01 & 25.27 & 28.65 & 28.06 & 28.40 & 33.33 & 27.12 \\ Cell & 33.24 & 1.28 & 23.59 & 23.33 & 25.51 & 23.59 & 23.71 & 0.13 & 22.90 \\ Breast & 28.01 & 23.59 & 1.11 & 2.74 & 1.94 & 0.05 & 0.39 & 23.72 & 1.05 \\ Colon & 25.27 & 23.33 & 2.74 & 1.87 & 3.77 & 2.79 & 3.13 & 23.46 & 1.95 \\ Kidney & 28.65 & 25.51 & 1.94 & 3.77 & 1.25 & 1.93 & 1.81 & 25.64 & 2.82 \\ Uterus & 28.06 & 23.59 & 0.05 & 2.79 & 1.93 & 1.35 & 0.34 & 23.73 & 1.09 \\ Ovary & 28.40 & 23.71 & 0.39 & 3.13 & 1.81 & 0.34 & 1.39 & 23.83 & 1.42 \\ PSC & 33.33 & 0.13 & 23.72 & 23.46 & 25.64 & 23.73 & 23.83 & 0.78 & 23.02 \\ Lung & 27.12 & 22.90 & 1.05 & 1.95 & 2.82 & 1.09 & 1.42 & 23.02 & 1.37 \end{array}_{9\times 9}.$$

Box II.

$$M^{(2)} = \begin{array}{c|cccccccc} & Cell & Breast & Colon & Kidney & Uterus & Ovary & PSC & Lung \\ \hline Cell & 1.25 & 23.12 & 23.21 & 29.83 & 23.52 & 23.93 & 0.12 & 22.58 \\ Breast & 23.12 & 1.31 & 4.42 & 15.02 & 5.88 & 7.57 & 23.24 & 3.43 \\ Colon & 23.21 & 4.42 & 1.75 & 19.43 & 1.46 & 3.14 & 23.33 & 1.17 \\ Kidney & 29.83 & 15.02 & 19.43 & 1.63 & 20.87 & 22.54 & 29.92 & 18.45 \\ Uterus & 23.52 & 5.88 & 1.46 & 20.87 & 1.61 & 1.69 & 23.64 & 2.57 \\ Ovary & 23.93 & 7.57 & 3.14 & 22.54 & 1.69 & 1.53 & 24.05 & 4.24 \\ PSC & 0.12 & 23.24 & 23.33 & 29.92 & 23.64 & 24.05 & 0.77 & 22.70 \\ Lung & 22.58 & 3.43 & 1.17 & 18.45 & 2.57 & 4.24 & 22.70 & 1.34 \end{array}_{8\times 8}.$$

Box III.

of LDA and form the distance matrix

$$M^{(3)} = \begin{array}{c|cccccc} & Cell & Colon & Uterus & Ovary & PSC & Lung \\ \hline Cell & 1.20 & 23.76 & 22.10 & 22.63 & 0.10 & 21.14 \\ Colon & 23.76 & 2.28 & 12.45 & 14.27 & 23.86 & 8.82 \\ Uterus & 22.10 & 12.45 & 1.77 & 1.83 & 22.20 & 3.73 \\ Ovary & 22.63 & 14.27 & 1.83 & 1.67 & 22.73 & 5.56 \\ PSC & 0.10 & 23.86 & 22.20 & 22.73 & 0.75 & 21.24 \\ Lung & 21.14 & 8.82 & 3.73 & 5.56 & 21.40 & 1.52 \end{array}_{6\times 6},$$

which shows that two classes are well-separated, corresponding to $j = 2, 5$; i.e., colon and lung as shown in the bottom left figure of Fig. 9. We delete these two classes from the dataset $D^{(3)}$ to form $D^{(4)}$, and proceed to the fifth iteration of LDA and form the distance matrix

$$M^{(4)} = \begin{array}{c|cccc} & Cell & Uterus & Ovary & PSC \\ \hline Cell & 1.23 & 22.37 & 22.14 & 0.09 \\ Uterus & 22.37 & 2.65 & 9.31 & 22.45 \\ Ovary & 22.14 & 9.31 & 2.22 & 22.22 \\ PSC & 0.09 & 22.45 & 22.22 & 0.77 \end{array}_{4\times 4}.$$

In this case, two classes are well-separated, corresponding to $j = 2, 3$; i.e., ovary and uterus as shown in the bottom right figure of Fig. 9. We delete these two classes from the dataset $D^{(4)}$ to form $D^{(5)}$ and perform the sixth iteration of LDA. No separable class is found at this step, as a result, the process stops.

In each of the five iterations, a projection matrix is obtained from the singular vectors, $V^{(0)}, \dots, V^{(4)}$. Next, we show how the proposed method works for one sample (let us denote it as z, and is shown in Fig. 10 as a red solid triangle). We show step by step how to identify the class that z belongs to using the projections produced at the training stage of Algorithm 4.

We first compute the projection of z to the reduced space by the first projection matrix $V^{(0)}$, i.e., $z_0 = V^{(0)T} z$, then we compute the distance of z_0 to the centers of all classes $r_j^{(0)} = \|z_0 - v_0^j\|$, $j = 1, \dots, 11$, as shown below

$$\begin{array}{ccccccccccc} Liver & Brain & Cell & Breast & Colon & Kidney & Uterus & Ovary & PSC & Lung & Heart \\ (\,102.27 & 122.28 & 123.82 & 104.55 & 102.79 & 103.72 & 102.43 & 102.44 & 103.63 & 108.13 & 1.37\,)_{1\times 11}, \end{array} \tag{25}$$

which says that the distance to the heart class is the smallest. Using the second projection matrix $V^{(1)}$, we calculate the distance between the test sample and the classes as a vector

$$\begin{array}{ccccccccc} Liver & Cell & Breast & Colon & Kidney & Uterus & Ovary & PSC & Lung \\ (\,23.35 & 16.44 & 6.23 & 4.07 & 7.26 & 5.41 & 5.32 & 17.64 & 4.87\,)_{1\times 9}. \end{array} \tag{26}$$

The smallest value is 4.07 which shows that the sample belongs, possibly, to the colon class according to the projection at this iteration. Following this approach, we go through all projections and then compare the smallest values from each iteration, and then the smallest of all the smallest values determines the class it belongs to; i.e. the value $c(k)$ that solves the problem

$$\min_k \|V^{(k)T} z - v_k^{c(k)}\|.$$

For this particular test sample the calculations show that it belongs to the heart class.

Table 5
The classification accuracies of ddLDA, idLDA, ms-LDA, and ms-LDA-tr on the CIFAR-10/100 and the gene cancer datasets.

	CIFAR-10	CIFAR-100	Gene Cancer
ddLDA	93.9%	90.1%	90.3%
idLDA	91.5%	83.9%	89.5%
ms-LDA	84.7%	83.1%	81.3%
ms-LDA-tr	92.8%	90.3%	90.1%

We report the test results using LDA and idLDA for different values of quantile radius. The dataset has 5,629 samples, and we pick 75% of the samples for training (4,220 cases) and 25% for testing (1,409 cases). Without loss of generality, we randomly separate the samples into training and testing sets.

We repeat the test 15 times, and in Table 2, we show the best, the worst, and the average testing accuracy obtained with different values of quantile radius (QR) from the idLDA algorithm, as well as the results obtained with LDA. We find that idLDA significantly improves the classification results when compared with LDA. The classification results obtained with idLDA with different values of the training and testing QR vary from 81.32% to 89.51%. This suggests that a careful selection of the quantile radii is important and the optimal values are problem dependent.

Note that in this experiment, we find that cell line cancer and PSC are not separable because PSC is also a kind of cell line cancer. Therefore, mathematically speaking, the best classification rate is 91.6%, which is very close to the 89.5% average accuracy obtained by idLDA with a 75% training QR and 95% testing QR.

Next, we compare ddLDA and idLDA for the same datasets. In the gene expression dataset of cancer patients, ddLDA works well achieving a 90.3% average accuracy with an 80% training QR and an 85% testing QR. Compared to the 89.5% in idLDA, we see that ddLDA and idLDA offer similar accuracy but ddLDA is cheaper in terms of the computational time.

As mentioned earlier, idLDA sometimes does not work well for some datasets, and reducing the problem size is often needed. In the experiments of the CIFAR-10/100 datasets, we first decompose the dataset. After that, idLDA works well. In the CIFAR-10 dataset with 2 partitions, the average classification accuracy is 81.6%. For 3 partitions, the average classification accuracy is 91.5%. For the CIFAR-100 dataset with 10 partitions, the average classification accuracy is 79.2%. For 25 partitions, the average classification accuracy is 83.9%, which is better than the ddLDA algorithm.

We also compare ms-LDA and ms-LDA-tr proposed in Huang et al. [28] with the ddLDA algorithm for the gene cancer dataset and the CIFAR-10/100 datasets. Table 5 shows the classification accuracies of ddLDA, idLDA, ms-LDA, and ms-LDA-tr algorithms. The accuracy of ms-LDA is 81.3%, 84.7%, and 83.1% respectively, and the accuracy of ms-LDA-tr is 90.1%, 92.8%, and 90.3% respectively, that are similar to our results. The advantages of ddLDA are a high level of parallelism and reduced complexity. For problems with a large number of classes, idLDA is able

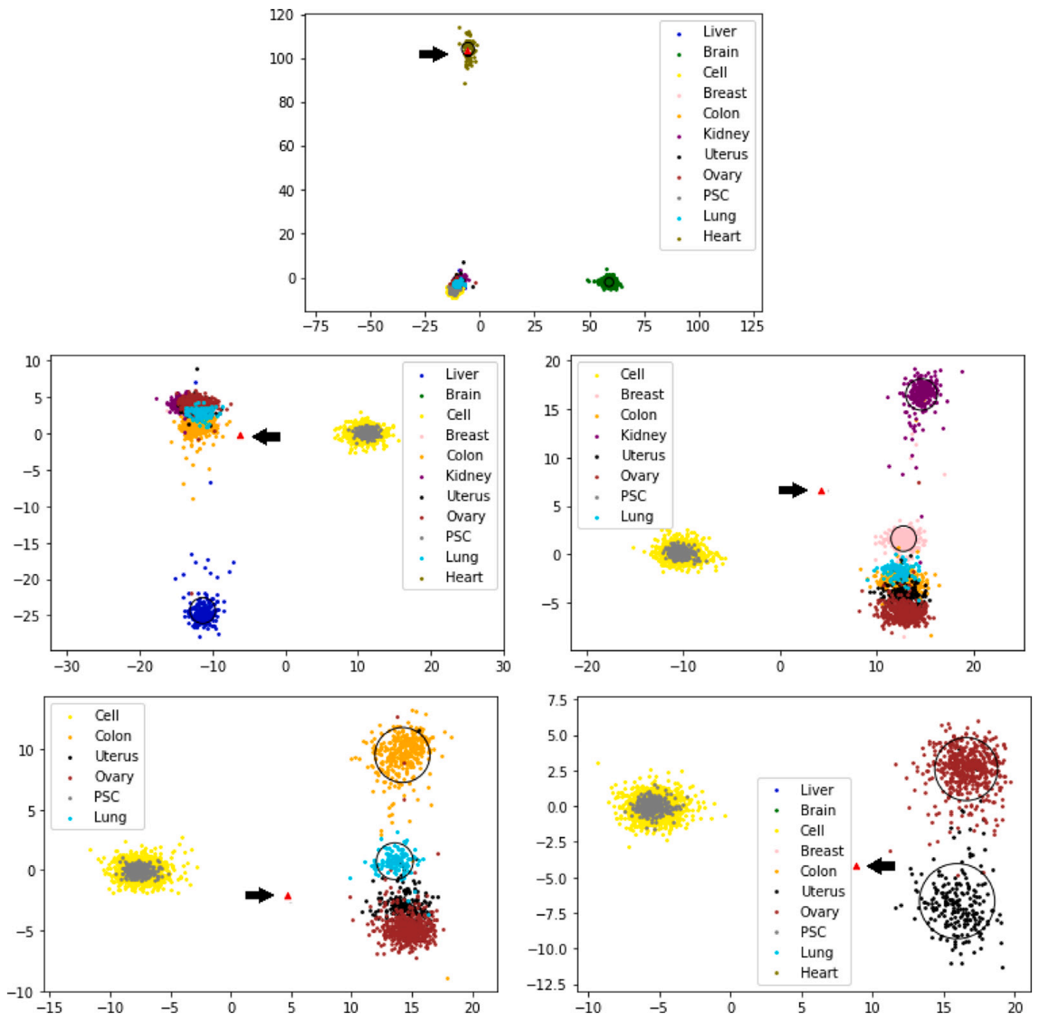


Fig. 10. We project a testing sample z (marked as a red triangle) onto the reduced space with $d = 2$ using different projections. The figure in the first row shows the projection using $V^{(0)}$, the figures in the second row show the projection using $V^{(1)}$ and $V^{(2)}$, respectively; the figures in the third row show the projection using $V^{(3)}$ and $V^{(4)}$, respectively. The circles are centered at the center of the separable classes with a radius that is equal to the 95% quantile radius.

to remove some of the classes and reduce the overall complexity of the problem. Although not discussed, both ms-LDA and ms-LDA-tr can be accelerated by parallelization with a similar idea as multi-subspace methods that are closely related to domain decomposition used in this paper.

7. Conclusions

The traditional LDA works well for classification problems when the number of classes is small, and the accuracy decreases when the number of classes increases. In this work we introduce two techniques for problems with a relatively large number of classes. In contrast to the traditional LDA, we use LDA only for subsets of the problem that have a smaller number of classes. The first technique is called domain decomposition LDA borrowed the idea from parallel methods for solving partial differential equations, and the second technique is called iterative deflation LDA borrowed from the idea in linear algebra for eigenvalue calculations.

In the domain decomposition method, the training set is decomposed into several sub-classes called subdomains. In each subdomain, we select a dimension of the reduced space and then apply the traditional LDA in this subspace. The testing is carried out for all subdomains and the classification is determined by a distance vector whose pieces are defined in the subdomains. Using different dimensions for different subdomains, for the CIFAR-10 dataset, we improved the accuracy of the traditional LDA from 47.3% to 93.7%. In comparison, in a separate work by the co-author [15], the same dataset is studied with several DCNN methods, and the best result is 94.8%. In other words, ddLDA is almost as accurate as DCNN, and it is a deterministic method with far fewer parameters to adjust. For the CIFAR-100 dataset, we improve the accuracy of the traditional LDA from 21.7% to 75.3%. Moreover, we introduce a data-specific partition to further improve ddLDA. With the data-specific ddLDA, the testing accuracy increases up to 90.1% for the CIFAR-100 dataset. Another advantage of ddLDA is that the subdomain training problems are all independent of each other and can be carried out in parallel which is important for solving large scale problems on large scale parallel computers.

The iterative deflation method is identical to LDA in the first iteration, but after each iteration, we deflate the classes that are well-separated until we exhaust all separable classes. In the traditional LDA, only one projection matrix is produced, while in the new approach, several projection matrices are generated. Therefore, to determine if a sample belongs to a certain class or classes, we need to solve another optimization problem which is often small and not time-consuming. As an example, the classification of cancer patients was studied based on their DNA data. The accuracy of the traditional LDA is 71.37%, and the new approach offers a much better accuracy at 89.51%. We mention that because PSC is also a kind of cell line cancer, therefore, mathematically speaking, the best classification result is 91.6%. In other words, our result is almost the best possible result.

CRediT authorship contribution statement

Jingwei Li: Conceptualization, Methodology, Writing – original draft. **Xiao-Chuan Cai:** Supervision, Resources, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

References

[1] A. Tharwat, T. Gaber, A. Ibrahim, A.E. Hassanien, Linear discriminant analysis: A detailed tutorial, AI Commun. 30 (2017) 169–190, http://dx.doi.org/10.3233/AIC-170729.

[2] R.A. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugen. 7 (2) (1936) 179–188.

[3] C.R. Rao, The utilization of multiple measurements in problems of biological classification, J. R. Stat. Soc. Ser. B Stat. Methodol. 10 (2) (1948) 159–203, http://www.jstor.org/stable/2983775.

[4] D. Huang, Y. Quan, M. He, B. Zhou, Comparison of linear discriminant analysis methods for the classification of cancer based on gene expression data, J. Exp. Clin. Cancer Res. 28 (2) (2009) 149–203, http://dx.doi.org/10.1186/1756-9966-28-149.

[5] C. Ricciardi, A.S. Valente, K. Edmund, V. Cantoni, R. Green, A. Fiorillo, I. Picone, S. Santini, M. Cesarelli, Linear discriminant analysis and principal component analysis to predict coronary artery disease, Health Inf. J. 26 (3) (2020) 2181–2192.

[6] K.T. Le, C. Chaux, F. Richard, E. Guedj, An adapted linear discriminant analysis with variable selection for the classification in high-dimension, and an application to medical data, Comput. Statist. Data Anal. 152 (2020) 107031, http://dx.doi.org/10.1016/j.csda.2020.107031, https://www.sciencedirect.com/science/article/pii/S0167947320301225.

[7] A. Sengur, An expert system based on linear discriminant analysis and adaptive neuro-fuzzy inference system to diagnosis heart valve diseases, Expert Syst. Appl. 35 (1) (2008) 214–222, http://dx.doi.org/10.1016/j.eswa.2007.06.012, https://www.sciencedirect.com/science/article/pii/S0957417407002291.

[8] M. Toğaçar, B. Ergen, Z. Cömert, Application of breast cancer diagnosis based on a combination of convolutional neural networks, ridge regression and linear discriminant analysis using invasive breast cancer images processed with autoencoders, Med. Hypotheses 135 (2020) 109503, http://dx.doi.org/10.1016/j.mehy.2019.109503.

[9] X.-C. Cai, Y. Saad, Overlapping domain decomposition algorithms for general sparse matrices, Numer. Linear Algebra Appl. 3 (3) (1996) 221–237.

[10] B. Smith, P. Bjørstad, W. Gropp, Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations, Cambridge University Press, United Kingdom, 1996.

[11] L. Luo, L. Liu, X.-C. Cai, D. Keyes, Fully implicit hybrid two-level domain decomposition algorithms for two-phase flows in porous media on 3D unstructured grids, J. Comput. Phys. 409 (2020) 109312, http://dx.doi.org/10.1016/j.jcp.2020.109312.

[12] A. Toselli, O. Widlund, Domain Decomposition Methods-Algorithms and Theory, Vol. 34, Springer Science & Business Media, 2004.

[13] L. Gu, X.-C. Cai, Fusing 2D and 3D convolutional neural networks for the segmentation of aorta and coronary arteries from CT images, Artif. Intell. Med. 121 (2021) 102189, http://dx.doi.org/10.1016/j.artmed.2021.102189.

[14] L. Gu, W. Zhang, J. Liu, X.-C. Cai, Decomposition and preconditioning of deep convolutional neural networks for training acceleration, in: Proceedings of the 26th International Conference on Domain Decomposition Methods, 2023.

[15] L. Gu, W. Zhang, J. Liu, X.-C. Cai, Decomposition and composition of deep convolutional neural networks and training acceleration via sub-network transfer learning, Electron. Trans. Numer. Anal. 56 (2022) 157–186.

[16] J. Li, X.-C. Cai, Summation pollution of principal component analysis and an improved algorithm for location sensitive data, Numer. Linear Algebra Appl. 28 (5) (2021) http://dx.doi.org/10.1002/nla.2370.

[17] G.H. Golub, C.F. Van Loan, Matrix Computations, third ed., The Johns Hopkins University Press, 1996.

[18] E. Altman, Financial ratios, discriminant analysis and the prediction of corporate bankruptcy, J. Finance 23 (4) (1968) 589–609.

[19] P. Panayides, Marketing in Asia-Pacific logistics companies: A discriminant analysis between marketing orientation and performance, Asia Pacific J. Market. Logist. 16 (2004) 42–68, http://dx.doi.org/10.1108/13555850410765122.

[20] M. Sunny, N. Islam, S. Afroz, M. Hossain, Predicting stock market price of Bangladesh: A comparative study of linear classification models, Ann. Data Sci. 8 (1) (2021) 21–38, http://dx.doi.org/10.1007/s40745-020-00318-5.

[21] P. Tahmasebi, A. Hezarkhani, M. Mortazavi, Application of discriminant analysis for alteration separation; sungun copper deposit, East Azerbaijan, Iran, Australian J. Basic Appl. Sci. 4 (2010) 564–576.

[22] A.N. Gorban, A. Golubkov, B. Grechuk, E.M. Mirkes, I.Y. Tyukin, Correction of AI systems by linear discriminants: Probabilistic foundations, Inform. Sci. 466 (2018) 303–322.

[23] M. Mngadi, J. Odindi, K. Peerbhay, O. Mutanga, Examining the effectiveness of sentinel-1 and 2 imagery for commercial forest species mapping, Geocarto Int. 36 (1) (2021) 1–12, http://dx.doi.org/10.1080/10106049.2019.1585483.

[24] A. Khoder, F. Dornaika, An enhanced approach to the robust discriminant analysis and class sparsity based embedding, Neural Netw. 136 (2021) 11–16, http://dx.doi.org/10.1016/j.neunet.2020.12.025.

[25] D. Kim, B. Song, Virtual sample-based deep metric learning using discriminant analysis, Pattern Recognit. 110 (2021) 107643.

[26] H. Sifaou, A. Kammoun, M. Alouini, High-dimensional linear discriminant analysis classifier for spiked covariance model, J. Mach. Learn. Res. 21 (112) (2020) 1–24, http://jmlr.org/papers/v21/19-428.html.

[27] A. Ortega-Martinez, A. Von Lühmann, M.A. Yücel, P. Farzam, D. Rogers, D.A. Boas, Real-time regression and classification of functional near infrared spectroscopy signals acquired during motor tasks, in: Proceedings of the Optical Techniques in Neurosurgery, Neurophotonics, and Optogenetics, 2021, http://dx.doi.org/10.1117/12.2578674, 116292M.

[28] Y. Huang, Y. Guan, On the linear discriminant analysis method for large number of classes, Eng. Appl. Artif. Intell. 43 (2015) 15–26.

[29] E. Kokiopoulou, J. Chen, Y. Saad, Trace optimization and eigenproblems in dimension reduction methods, Numer. Linear Algebra Appl. 18 (2011) 565–602, http://dx.doi.org/10.1002/nla.743.

[30] A. Krizhevsky, V. Nair, G. Hinton, [CIFAR]-10/100 (Canadian, Institute for Advanced Research, http://www.cs.toronto.edu/kriz/cifar.html.

[31] M. Lenz, F. Müller, M. Zenke, A. Schuppert, Principal components analysis and the reported low intrinsic dimensionality of gene expression microarray data, Sci. Rep. 6 (1) (2016) 1–11.

Jingwei Li received his Ph.D. in Computer Science at the University of Colorado Boulder. His research interests are unsupervised learning and supervised learning. His current research focuses on domain decomposition methods and their innovative application in machine learning.

Xiao-Chuan Cai is UMDF Chair Professor of Applied Mathematics at the University of Macau and a SIAM Fellow. He received his B.Sc. degree in 1984 from Peking University, his M.Sc. in 1988 and Ph.D. in 1989 from Courant Institute, New York University. His research interests are in the general area of scientific and engineering computing including parallel algorithms and high-performance software for linear and nonlinear partial differential equations, domain decomposition methods, multigrid methods, numerical linear algebra, PDE constrained optimizations, stochastic partial differential equations, computational fluid dynamics, computational biomechanics, and parallel processing.