AIMS Mathematics, 9(7): 18777–18795.
DOI: 10.3934/math.2024914
Received: 02 March 2024
Revised: 23 May 2024
Accepted: 28 May 2024
Published: 04 June 2024

https://www.aimspress.com/journal/Math

Research article

A breakdown-free block conjugate gradient method for large-scale discriminant analysis

Wenya Shi* **and Zhixiang Chen**

School of Computer Science and Artifical Intelligence, Changzhou University, Changzhou, 213164, Jiangsu, China

* **Correspondence:** Email: wenyashi@cczu.edu.cn.

Abstract: Rayleigh-Ritz discriminant analysis (RRDA) is an effective algorithm for linear discriminant analysis (LDA), but there are some drawbacks in its implementation. In this paper, we first improved Rayleigh-Ritz discriminant analysis (IRRDA) to make its framework more concise, and established the equivalence theory of the solution space between our discriminant analysis and RRDA. Second, we proposed a new model based on positive definite systems of linear equations for linear discriminant analysis, and certificated the rationality of the new model. Compared with the traditional linear regression model for linear discriminant analysis, the coefficient matrix of our model avoided forming a centralized matrix or appending the original data matrix, but the original matrix itself, which greatly reduced the computational complexity. According to the size of data matrix, we designed two solution schemes for the new model based on the block conjugate gradient method. Experiments in real-world datasets demonstrated the effectiveness and efficiency of our algorithm and it showed that our method was more efficient and faster than RRDA.

Keywords: Rayleigh-Ritz discriminant analysis; block conjugate gradient method; discriminant analysis; positive definite systems of linear equations
Mathematics Subject Classification: 65F05

1. Introduction

Dimensionality reduction is a common technique for extracting effective information from high-dimensional data, which can validly reduce the computational complexity and storage demands. Therefore, dimensionality reduction has become a research hotspot in recent years [1–3]. Linear discriminant analysis (LDA) [4–8] is a well-known method for dimensionality reduction, which has been widely used in many applications such as face recognition [1, 9, 10], machine learning [11, 12], image classification [13, 14], image reconstruction [15], and information retrieval [2, 16].

Suppose n data points are divided into c classes $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n] = [X_1, \ldots, X_c] \in \mathbb{R}^{d\times n}$, where $X_j \in \mathbb{R}^{d\times n_j}$ is the j-th class with n_j being the number of samples, and $\sum_{j=1}^{c} n_j = n$. Denote by $\mathbf{c}_j$ the centroid of class X_j, and by $\mathbf{c}$ the global centroid. The within-class scatter, between-class scatter and total scatter matrices are defined as

$$S_w = \sum_{j=1}^{c} \sum_{\mathbf{x}_i \in X_j} (\mathbf{x}_i - \mathbf{c}_j)(\mathbf{x}_i - \mathbf{c}_j)^T, \quad S_b = \sum_{j=1}^{c} n_j(\mathbf{c}_j - \mathbf{c})(\mathbf{c}_j - \mathbf{c})^T, \quad S_t = \sum_{j=1}^{n} (\mathbf{x}_j - \mathbf{c})(\mathbf{x}_j - \mathbf{c})^T,$$

respectively. LDA looks for a classification that projects high-dimensional data onto a low-dimensional space and achieves the maximum class separability of data [17, 18]. In order to make LDA more suitable for dimensionality reduction, various extensions of LDA are proposed [19, 20]. Among those extensions, one popular LDA criterion is

$$W = \arg \max_{W^T W = I_r} \operatorname{tr}((W^T S_t W)^{\dagger}(W^T S_b W)). \tag{1.1}$$

As S_t is a symmetric positive semidefinite matrix, the regularization parameter is often used to overcome singularity problems [19]. S_t in (1.1) is replaced by $S = S_t + \alpha I_d$, which in results the regularized linear discriminant analysis (RLDA) problem. The literature [21] proposes a branching and binding method for solving problems in the case of reducing to one-dimensional space. The literature [22] proposes a new dual parameter regularization BLDA (RBLDA) for MTS data classification and develops an efficient model selection algorithm.

Generally, an LDA problem can be solved equivalently by a generalized eigenvalue problem [4, 17, 18]

$$S_b \mathbf{w} = \lambda S_t \mathbf{w}.$$

However, straightforward implementation of generalized eigenvalue decomposition is very time-consuming and prohibitive for high-dimensional data [18, 23]. To cure this drawback, researchers use QR decomposition, singular value decomposition, the Krylov subspace method and other techniques to transform the large-scale generalized eigenvalue decomposition problem into a small size matrix decomposition problem, resulting in a series of methods for high-dimensional data, such as QRLDA [24], GSVDLDA [25], RRDA [18], and so on [3, 25–27]. Recently, random sampling [17] and randomized SVD (RSVD) [26, 28] techniques have been used to accelerate LDA algorithms, and it shows that RSVD is very efficient for high-dimensional and large-scale dense data. Among these algorithms, RRDA [18] solved the generalized eigenvalue problem using a gradient-like method using the Rayleigh-Ritz framework, showng its superiority to some popular LDA algorithms both on the optimal target value and classification accuracy. However, RRDA is not concise in the implementation framework, hence we are committed to studying its convenient framework in this paper.

In addition to solving LDA by eigenvalue problems, LDA can also be solved by multivariate linear regression with a specific class indicator matrix [5, 8, 29]. In [8], the author showed that the solution of multivariate linear regression $\min_M \|\frac{1}{\sqrt{n}}\bar{X}^T M - Y\|_F^2$ can solve the corresponding LDA problem (1.1), where

$$Y_{jk} = \begin{cases} \sqrt{\frac{n}{n_k}} - \sqrt{\frac{n_k}{n}} & if \quad j \in \mathcal{N}_k, \\ -\sqrt{\frac{n_k}{n}} & otherwise, \end{cases}$$

is a class indicator matrix and $\bar{X} = X(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T)$ is the centralized data matrix. Similarly, [27] established an equivalence between RLDA and a ridge regression, and [29] investigated the relationship between LDA and the minimum squared error. Chen et al. [30] proposed a bidirectional linear discriminant analysis method for image data based on low rank approximation. SRDA [23] cast LDA into a regression framework using spectral graph analysis is a representative algorithm for solving high-dimensional and large-scale sparse data. LDADL [20] showed an equivalence solution space between LDA and a multivariate linear regression, and proposed an incremental method for large-scale data. Most least square (LS) problems for LDA are based on the data matrix, not the data itself, which will increase computational complexity. In this paper, we consider establishing the relationship between LDA and multivariate linear regression through the original matrix itself.

The structure of this paper is as follows. In Section 2, we first introduce the RRDA method and analyze its advantages and disadvantages, then put forward some settlements for the defects and optimize RRDA. In Section 3, we establish the equivalence of LDA and a new linear symmetric and positive equations system, which contains only data matrix and class indicator matrix. In Section 4, we conduct some numerical experiments by a collection of large-scale and high-dimensional databases from real face images, speech corpus, and video. Numerical results show the effectiveness of our proposed method, and some concluding remarks are given in Section 5.

Here are some notations in our paper. $\mathbf{1}_i$ stands for the vector of all ones with dimension i, I_i is the identity matrix whose dimension is i, $\mathbf{0}$ means a zero matrix or vector. A^T, A^{-1} and $A^\dagger$ are the transposition, inverse, and Moore-Penrose inverse of A, respectively. $\|A\|_2$, $\|A\|_F$ represent the 2-norm and Frobenius norm of A. $span\{A\}$ consists of the space spanned by the columns of A, $rank(A)$ stands for the rank of A, $orth(A)$ is an orthonormal basis for the range of A, and $tr(A)$ indicates the trace of A.

2. The RRDA method and its improved version

In this section, we briefly review the (RRDA) method [18] and optimize it.

2.1. RRDA

RRDA is a conjugate-gradient-like method based on the Rayleigh-Ritz framework for the generalized eigenvalue problem $S_b\mathbf{w} = \lambda S\mathbf{w}$ of RLDA, where

$$S = S_t + \alpha I_d. \tag{2.1}$$

For the sake of the Rayleigh-Ritz framework, RRDA rewrote S_t and S_b as [18]

$$S_t = \bar{X}\bar{X}^T, \quad S_b = HH^T, \tag{2.2}$$

where

$$\bar{X} = X(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T), \quad H = \bar{X}Y, \quad Y = diag(\frac{1}{\sqrt{n_1}}\mathbf{1}_{n_1}, \frac{1}{\sqrt{n_2}}\mathbf{1}_{n_2}, \cdots, \frac{1}{\sqrt{n_c}}\mathbf{1}_{n_c}). \tag{2.3}$$

Therefore, RRDA solves the following problem for dimensionality reduction:

$$HH^T\mathbf{w} = \lambda S\mathbf{w}, \ \lambda > 0. \tag{2.4}$$

RRDA designs efficient subspace expansion and extraction strategy on the Rayleigh-Ritz framework for generalized eigenvalue problems (2.4). In the expansion phase, it chooses a subspace basis P_k satisfying

$$\min_{P_k} \|grad_{M_{k-1}} \mathrm{tr}((M_{k-1}^T S M_{k-1})^\dagger M_{k-1}^T H H^T M_{k-1}) - P_k\|_F^2, \ \text{ s.t. } P_k S P_i = \mathbf{0}, \ \ i < k,$$

such that $V_k = [V_{k-1}, P_k]$ in iteration k, where M_{k-1} from the prior extraction phase with the initial $M_0 = \mathbf{0}$. P_k in the above formula can be computed by the following theorem [18].

Theorem 1. *[18] Define* $\{P_i\}$ *and* $\{R_i\}$ *as*

$$R_i = \begin{cases} H, & i = 1 \\ R_{i-1} - S P_{i-1}(P_{i-1}^T S P_{i-1})^\dagger P_{i-1}^T R_{i-1}, & i > 1 \end{cases}, \ P_i = \begin{cases} H & i = 1 \\ R_i - P_{i-1}(P_{i-1}^T S P_{i-1})^\dagger P_{i-1}^T S R_i & i > 1 \end{cases},$$

then

$$R_k^T P_{k-1} = \mathbf{0}, \quad R_k^T R_j = \mathbf{0}, \quad P_k^T S P_j = \mathbf{0}, \quad j < k. \tag{2.5}$$

In the extraction phase, it computes the eigenpair (λ_i, η_i) of $(V_k^T H H^T V_k, V_k^T S V_k)$ and computes the Ritz vectors $\mathbf{w}_i = V_k \eta_i$ until convergence. Let $W_k = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_c]$, where c is the reduced dimension, then W_k is the approximation of the eigenvectors of (2.4) associated with the first c eigenvalues.

As S is the positive definite matrix, computing the eigenpairs of $(V_k^T S V_k, V_k^T H H^T V_k)$ can transform into computing the eigenpairs of $\lambda\eta = (V_k^T S V_k)^\dagger (V_k^T H)(V_k^T H)^T \eta$ in the following problem [31]. It is known that if (λ, η) is the nonzero eigenpair of $B^T A B$, then ABB^T has the eigenvector $AB\eta$ with the same eigenvalue. Hence, let $T_k = H^T M_k$, $M_k = V_k (V_k^T S V_k)^\dagger V_k^T H$, and U_k be the eigenvectors with the positive eigenvalues of T_k. Then, the Ritz pair of (HH^T, S) can be recovered by $V_k \cdot (V_k^T S V_k)^\dagger V_k^T H \cdot U_k = M_k U_k$. On the other hand, according to Theorem 1, $M_k = V_k (V_k^T S V_k)^\dagger V_k^T H$ can be computed equivalently by

$$M_k = \sum_{i=1}^{k} P_i (P_i^T S P_i)^\dagger P_i^T H. \tag{2.6}$$

Thus, the specific algorithm of RRDA is shown in Algorithm 1.

Algorithm 1 Rayleigh-Ritz Discriminant Analysis (RRDA) [18].

1) Input: Data matrix X, class indicator matrix Y, and a tolerance ϵ.
2) Start: Initialize $R_k = H$, $M_k = Q_k = \mathbf{0}_{d\times c}$, $T_k = L_k = E_k = \mathbf{0}_{c\times c}$, $S = S_t + \alpha I_d$.
3) Inner loop: For k=1,2,..., do:
   3.1) Expansion: $P_k = R_k - P_k E_k Q_k^T R_k$.
   3.2) Extraction:
      3.2.1) $Q_k = S P_k$, $E_k = (P_k^T Q_k)^\dagger$, $L_k = P_k^T H$;
      3.2.2) $M_k = M_k + P_k E_k L_k$, $R_k = R_k - Q_k E_k L_k$, $T_k = H^T M_k$;
      3.2.3) Compute the EVD of T_k, set $W_k = M_k U_k$, U_k is the eigenvectors of T_k associated with the positive eigenvalues.
   3.3) Stop if $\|W_k - W_{k-1}\|_F < \epsilon$.
4) End: Output W_k as the approximation of W_{opt}.

2.2. An improved algorithm of RRDA

Throughout the implementation of RRDA, there are some shortcomings and suggestions here.

First, the large-scale matrix $S = \bar{X}\bar{X}^T + \alpha I_d$ is formed in the initial step, which results in a waste of time and memory. Notice that S is only used in $Q_k = SP_k$. We suggest computing Q_k by its equivalent form $Q_k = \bar{X}(\bar{X}^T P_k) + \alpha P_k$ without forming S, which will save a lot of computing memory.

Second, M_k can be updated by $M_k = M_k + P_k E_k L_k$ as (2.6), and (2.6) holds because of the orthogonality conditions of (2.5). Therefore, the premise for the effective implementation of RRDA is the orthogonality of P_k and R_k, while their orthogonality may lose gradually during its running due to computational error. Hence, we need to re-orthogonalize P_k in Step 3.1, i.e., set

$$P_k = orth(P_k).$$

Fortunately, [32] has proved that $R_k^T P_{k-1} = \mathbf{0}$, $R_k^T R_j = \mathbf{0}$ would still be established both on theoretical and computational results with $P_k = orth(P_k)$. Further, $P_k^T Q_k$ is symmetric positive definite matrix under the orthogonality of P_k, thus one can compute $E_k = (P_k^T Q_k)^{-1}$ instead of computing the Moore-Penrose inverse $E_k = (P_k^T Q_k)^{\dagger}$ in Step 3.2.1.

Third, we suggest placing the term $T_k = H^T M_k$ in Steps 3.2.2 and the 3.2.3 outside the loop, as the update of T_k and W_k is only related to M_k. This is our essential improvement proposal for RRDA. In this way, we can reduce the CPU time of RRDA since it only computes the matrix-matrix products $T_k = H^T M_k$, $W_k = M_k U_k$ and the eigenvalue decomposition of T_k once, while RRDA needs to compute k times. In essence, ignoring the calculation of T_k and W_k, the expansion and extraction of RRDA (Algorithm 1) is the process of solving positive definite equations

$$SM = H, \tag{2.7}$$

by the block conjugate gradient method, where P_k is the descent direction, R_k is the residual term, and M_k is the iterative solution. Therefore, the stopping criterion $\|W_k - W_{k-1}\|_F < \epsilon$ in Step 3.3 is improper. In fact, due to the gradual loss of orthogonality of P_k, the dimensions of W_k and W_{k-1} may be different during running, resulting in the failure of the RRDA. Thus, we suggest using $\|R_k\|_F < \epsilon$ instead of $\|W_k - W_{k-1}\|_F < \epsilon$ as the stopping criterion.

Remark 1. *The solving process of Algorithms 2 and 1 is similar, so convergence analysis can refer to the convergence analysis of Algorithm 1. Let us briefly discuss the computational complexities of this algorithm. The main computational complexity is in Step 3, which costs about $O(dnc + d^2c+)$ flops. In Case 2, the main computational complexity is in Step 10, which costs about $O(dnc + c^3 + c^2d + d^2c)$ flops. Generally speaking, the class number c is far less than the characteristic dimension d and the sample number n, thus the total costs are about $O(dnc + c^2d + d^2c)$ flops.*

In conclusion, we summarize the improved algorithm for RRDA; see Algorithm 2. The essential difference between RRDA and IRRDA is the handing of T_k and W_k. RRDA updates T_k and W_k by M_k iteratively in the inner loop until W_k converges, while IRRDA calculates T_k and W_k outside the loop. Throughout RRDA, T_k converges as long as M_k converges because of the continuity of matrix elements. Thus, the iterations of RRDA and IRRDA may be close to each other when they converge. However, RRDA calculates eigenvalue decompositions more than IRRDA, so IRRDA has better computational efficiency.

Algorithm 2 An improved algorithm of RRDA (IRRDA).

1) Input: Data matrix X and a tolerance ϵ.
2) Start: Initialize $R_k = H$, $M_k = Q_k = \mathbf{0}_{d\times c}$.
% Solving $SM = H$ by the breakdown-free block conjugate gradient (BCG) algorithm [32]
3) Inner loop: For $k = 1, 2, \ldots$, do:
3.1) Expansion: $P_k = R_k - P_k E_k Q_k^T R_k$, $P_k = orth(P_k)$.
3.2) Extraction:
3.2.1) $Q_k = \bar{X}(\bar{X}^T P_k) + \alpha P_k$, $E_k = (P_k^T Q_k)^{-1}$, $L_k = P_k^T H$;
3.2.2) $M_k = M_k + P_k E_k L_k$, $R_k = R_k - Q_k E_k L_k$;
3.3) Stop if $\|R_k\|_F < \epsilon$.
4) Compute $W_k = M_k U_k$, where U_k is the eigenvectors of $T_k = H^T M_k$ associated with the positive eigenvalues.
5) End: Output W_k as the approximation of W_{opt}.

According to the implementation of IRRDA, IRRDA essentially computes $\mathbf{w}$ and satisfies the following problems:

$$\begin{cases} SM = H, \\ H^T M\mathbf{u} = \lambda\mathbf{u}, \ \ \lambda > 0, \\ \mathbf{w} = M\mathbf{u}. \end{cases} \tag{2.8}$$

Recall that S is symmetric positive definite. We have the following result relating RRDA problem (2.4) and IRRDA problem (2.8).

Theorem 2. *Let M be the solution of $SM = H$, then RRDA problem* (2.4) *and IRRDA problem* (2.8) *have the same solution.*

Proof. As $SM = H$ and S is symmetric positive definite, then $M = S^{-1}H$ and RRDA problem (2.4) can be rewritten as the following eigenvalue problem:

$$MH^T\mathbf{w} = \lambda\mathbf{w}, \ \ \lambda > 0. \tag{2.9}$$

Now, we prove that the solutions of RRDA problem (2.9) and IRRDA problem (2.8) are the same.

First, we prove that $\mathbf{w}$ in (2.8) solves RRDA problem (2.4). Left multiplying M from both sides of $H^T M\mathbf{u} = \lambda\mathbf{u}$, $\lambda > 0$ yields $MH^T(M\mathbf{u}) = \lambda(M\mathbf{u})$, which implies that if $\mathbf{u}$ is an eigenvector of $H^T M$ corresponding to λ, then $\mathbf{w} = M\mathbf{u}$ is an eigenvector of MH^T corresponding to λ.

Second, we prove that the solution of RRDA satisfies the IRRDA problem (2.8). Left multiplying H^T from both sides of (2.9) yields

$$H^T M(H^T\mathbf{w}) = \lambda(H^T\mathbf{w}), \ \ \lambda > 0, \tag{2.10}$$

which indicates that if $\mathbf{w}$ is an eigenvector of MH^T corresponding to λ, then $H^T\mathbf{w}$ is an eigenvector of $H^T M$ corresponding to λ. In other words, the eigenvectors of $H^T M$ corresponding to nonnegative eigenvalues can be computed by $\mathbf{u} = H^T\mathbf{w}$. Left multiplying M from both sides of (2.10) yields

$$MH^T M(H^T\mathbf{w}) = \lambda M(H^T\mathbf{w}), \ \ \lambda > 0, \tag{2.11}$$

which indicates that if $\mathbf{u} = H^T\mathbf{w}$ is an eigenvector of H^TM corresponding to λ, then $M\mathbf{u}$ is an eigenvector of MH^T corresponding to λ. Note $\mathbf{w}$ is also the eigenvector of MH^T corresponding to λ, thus $\mathbf{w} = M\mathbf{u}$. □

Theorem 2 depicts that the solution spaces of RRDA and IRRDA are equivalent. Fortunately, algorithm IRRDA has less computational complexity, which shows the effectiveness of the improved algorithm IRRDA. Moreover, let U be the eigenvectors of H^TM associated with the positive eigenvalues. IRRDA and Theorem 2 indicate $W = MU$ and solve the RLDA problem. More importantly, it is clear that the solution space of RLDA is a subset of $span\{M\}$.

3. A new linear system solving for RLDA

Notice from [8] that the solution of LDA and $S_t^{\dagger}H$ are equivalent, thus $M = S^{-1}H = (S_t + \alpha I_d)^{-1}H$ can be used to approximate the RLDA. Computing S and H requires forming $\bar{X} = X(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T)$, which is time-consuming for large-scale high-dimensional datasets. In large-scale data, the dimension d and sample number n of data are usually huge, and $\frac{1}{n}$ will be very small and even tend to 0. In this section, we consider replacing $\bar{X}$ with X, and characterize the rationality of solving RLDA by the following linear system:

$$\check{S}\check{M} = \check{H}, \tag{3.1}$$

where

$$\check{S} = XX^T + \alpha I_d,\ \check{H} = XY. \tag{3.2}$$

It is clear that

$$\begin{aligned} \check{M} &= \check{S}^{-1}\check{H} \\ &= (XX^T + \alpha I_d)^{-1}XY, \qquad (3.3) \\ &= X(X^TX + \alpha I_n)^{-1}Y. \qquad (3.4) \end{aligned}$$

According to the feature of the data and the number of samples, we compute $\check{M}$ in two different schemes:

(1) When $d \le n$, we compute $\check{M}$ by (3.3).

(2) When $d > n$, we first compute the solution $\check{M}$ by (3.4).

Here, we apply the breakdown-free block conjugate gradient (BCG) algorithm [32] to compute the linear equations for the symmetric semi-positive definite matrix. The specific algorithm of our new model for RLDA is shown in Algorithm 3.

Algorithm 3 A breakdown-free BCG algorithm for LDA (BBCGLDA).

Require: Data matrix $X \in \mathbb{R}^{d\times n}$ and a tolerance ϵ
Ensure: Transform matrix $\check{M}$
1: Case1: If $d > n$
2: Initialize $\check{R}_k = Y$, $\check{M}_k = \mathbf{0}_{n\times c}$, $\check{P}_k = orth(\check{R}_k)$.
3: **while** $\|\check{R}_k\|_F \geq \epsilon$ **do**
4: $\check{Q}_k = \alpha\check{P}_k + X^T(X\check{P}_k)$, $\check{E}_k = (\check{P}_k^T\check{Q}_k)^{-1}$, $\check{L}_k = E_k(\check{P}_k^T\check{R}_k)$, $\check{M}_k = \check{M}_k + \check{P}_k\check{L}_k$, $\check{R}_k = \check{R}_k - \check{Q}_k\check{L}_k$, $\check{P}_k = \check{R}_k - (\check{P}_k\check{E}_k)(\check{Q}_k^T\check{R}_k)$, $\check{P}_k = orth(\check{P}_k)$.
5: **end while**
6: $\check{M} = X\check{M}_k$
7: Case2: If $d \leq n$
8: Initialize $\check{R}_k = \check{H} = XY$, $\check{M}_k = \mathbf{0}_{d\times c}$, $\check{P}_k = orth(\check{R}_k)$
9: **while** $\|\check{R}_k\|_F \geq \epsilon$ **do**
10: $\check{Q}_k = \alpha\check{P}_k + X(X^T\check{P}_k)$, $\check{E}_k = (\check{P}_k^T\check{Q}_k)^{-1}$, $\check{L}_k = E_k(\check{P}_k^T\check{R}_k)$, $\check{M}_k = \check{M}_k + \check{P}_k\check{L}_k$, $\check{R}_k = \check{R}_k - \check{Q}_k\check{L}_k$, $\check{P}_k = \check{R}_k - (\check{P}_k\check{E}_k)(\check{Q}_k^T\check{R}_k)$, $\check{P}_k = orth(\check{P}_k)$.
11: **end while**
12: $\check{M} = \check{M}_k$

In Algorithm 3, the convergence of Cases 1 and 2 is similar. This article explains the convergence of Case 2 and provides a convergence theorem below.

Theorem 3. *Suppose $\check{H}$ is rank deficient with rank r_0 ($r_0 < c$). Let $\kappa = \lambda_{max}/\lambda_{min}$, where λ_{max} and λ_{min} are the maximum and nonzero minimum eigenvalues of matrix $\check{H}$, respectively. Let $E_k = \check{M}^* - \check{M}_k = [e_k^{(1)}, e_k^{(2)}, ..., e_k^{(c)}]$. The minimum error square norm $\|e_k^{(i)}\|^2$ is bounded as*

$$\|e_k^{(i)}\|^2 \leq t(\frac{1-\sqrt{\kappa^{-1}}}{1+\sqrt{\kappa^{-1}}})^{2(i+1)}, \tag{3.5}$$

where t is a constant only related to E_1.

Proof. Because Algorithm 3 is a special case in the literature [32], the proof is trivial and we refer to [32][Theorem 4.1]. □

Remark 2. *Let us briefly discuss the computational complexities of Algorithm 3. Algorithm 3 is divided into two cases, and their complexity is calculated separately. In Case 1, the main computational complexity is in Step 4, which costs about $O(dnc+nc^2+cn^2+c^3)$ flops. In Case 2, the main computational complexity is in Step 10, which costs about $O(dnc + nc^2 + cn^2 + c^3)$ flops. Generally speaking, the class number c is far less than the characteristic dimension d and the sample number n, thus the total costs are about $O(dnc + cn^2)$ flops.*

Algorithm 3 could perfectly avoid the disadvantages of RRDA. For example, $\check{R}_k^T\check{P}_{k-1} = \mathbf{0}$, $\check{R}_k^T\check{R}_j = \mathbf{0}$, $j < k$ are always held, and neither T_k, W_k nor eigenvalue decomposition need to be computed in Algorithm 3, which saves a lot of workloads than RRDA.

LDADL [20] is an effective RLDA method for large-scale data. It also avoids forming $\bar{X}$, but adds a new component '1' to each datum $\mathbf{x}_j$. Denote

$$\underline{X} = \begin{bmatrix} X \\ \mathbf{1}_n^T \end{bmatrix},\ D = diag(\sqrt{n_1}, \sqrt{n_2}, \ldots, \sqrt{n_c}),\ E = YD, \tag{3.6}$$

then LDADL solves the following problem [20]:

$$\underline{M}^* = \arg\min_{\underline{M}\in\mathbb{R}^{(d+1)\times c}} \|\underline{X}^T\underline{M} - E\|_F^2 + \alpha\|\underline{M}\|_F^2. \tag{3.7}$$

Let $\underline{M}^* = \begin{bmatrix} M^* \\ \mathbf{m}^T \end{bmatrix}$, Zhang et al. [20] showed that M^* is a solution of LDA problem (1.1) when $\alpha = 0$. Moreover, it is known that (3.7) has a unique explicit solution given by

$$\underline{M}^* = (\underline{X}\underline{X}^T + \alpha I_d)^{-1}\underline{X}E = \underline{X}(\underline{X}^T\underline{X} + \alpha I_n)^{-1}E = \begin{bmatrix} X(X^TX + \mathbf{1}_n\mathbf{1}_n^T + \alpha I_n)^{-1}E \\ \mathbf{1}_n^T(X^TX + \mathbf{1}_n\mathbf{1}_n^T + \alpha I_n)^{-1}E \end{bmatrix}.$$

Hence,

$$M^* = X(X^TX + \mathbf{1}_n\mathbf{1}_n^T + \alpha I_n)^{-1}E, \tag{3.8}$$

can solve the LDA problem.

Considering that both (3.1) and (3.7) are linear systems for solving LDA without forming the centralized data $\bar{X}$, we will focus on the connection between our model and LDADL in the following part.

Denote by

$$\mathbf{t} = [\sqrt{n_1}, \sqrt{n_2}, \ldots, \sqrt{n_c}]^T, \quad T = \mathbf{t}\mathbf{1}_n^T(X^TX + \alpha I_n)^{-1}Y, \quad \beta = \Sigma_{i,j=1}^n (X^TX + \alpha I_n)_{ij}^{-1}. \tag{3.9}$$

Then, we have the following theorem.

Theorem 4. *Let* $\underline{M}^* = \begin{bmatrix} M^* \\ \mathbf{m}^T \end{bmatrix}$ *be the solution of* (3.7) *and* $\check{M}$ *be the solution of* (3.1)*, and assume that* β*,* T *and* D *satisfy the formulas* (3.9) *and* (3.6)*, respectively. Then,* $M^* = \check{M}(I_c - \frac{1}{1+\beta}T)D$ *and* $span\{\check{M}\} = span\{M^*\}$.

Proof. Formula (2.3) indicates that $\mathbf{1}_n = Y\mathbf{t}$. According to (3.8), we have

$$\begin{aligned} M^* &= X(X^TX + \mathbf{1}_n\mathbf{1}_n^T + \alpha I_n)^{-1}E \\ &= X[(X^TX + \alpha I_n)^{-1} - \frac{(X^TX + \alpha I_n)^{-1}\mathbf{1}_n\mathbf{1}_n^T(X^TX + \alpha I_n)^{-1}}{1 + \mathbf{1}_n^T(X^TX + \alpha I_n)^{-1}\mathbf{1}_n}]YD \\ &= X(X^TX + \alpha I_n)^{-1}Y \cdot D - X(X^TX + \alpha I_n)^{-1}Y \cdot \frac{\mathbf{t}\mathbf{1}_n^T(X^TX + \alpha I_n)^{-1}Y}{1 + \Sigma_{i,j=1}^n (X^TX + \alpha I_n)_{ij}^{-1}} \cdot D \\ &= \check{M}(I_c - \frac{1}{1+\beta}T)D. \end{aligned} \tag{3.10}$$

Equation (3.10) implies $span\{M^*\} \subseteq span\{\check{M}\}$. Note that

$$rank(T) = rank(\mathbf{t}\mathbf{1}_n^T(X^TX + \alpha I_n)^{-1}Y) = 1,$$

and

$$trace(T) = trace(\mathbf{t}\mathbf{1}_n^T(X^TX + \alpha I_n)^{-1}Y) = \Sigma_{i,j=1}^n (X^TX + \alpha I_n)_{ij}^{-1} = \beta,$$

thus $I_c - \frac{1}{1+\beta}T$ is full of rank. Recall that $D = diag(\sqrt{n_1}, \sqrt{n_2}, \ldots, \sqrt{n_c})$ is full of rank, and there holds $rank\{\check{M}\} = rank\{M^*\}$. Hence, $span\{\check{M}\} = span\{M^*\}$. □

Since we are only interested in an orthogonal basis matrix of the solution space for the LDA problem, and we know from Theorem 4 that the transformation matrix of LDADL for RLDA and the transformation matrix of our paper for RLDA are the same, we can expect that our method shall achieve performance similar to LDADL. Moreover, as LDADL and SRDA have the similar performance [20], the performance of LDADL, SRDA, and our method BBCGLDA are similar. Compared BBCGLDA with LDADL and SRDA, BBCGLDA has two advantages over LDADL and SRDA: (1) The size of the BBCGLDA problem is smaller than that of LDADL and SRDA. (2) Our proposed model (3.1) is a symmetric positive definite linear equation, and the SRDA and LDADL problem are least squares problems. Since linear equations is a special case of the least squares problem, the methods for SRDA and LDADL can also be used to solve our model. In addition, our proposed problem for LDA is similar with the problem proposed in [28], which aims to compute $(X)^T Y$ by RSVD under the condition that the data matrix X is full of column rank. Therefore, our algorithm is more applicable because it has no restrictions.

4. Experiments

In this section, we perform some numerical experiments on some real-world datasets, and show the numerical behavior of IRRDA and our new framework BBCGLDA. All the experiments are run on a Hp workstation with 16 cores double Intel(R)Xeon(R) Platinum 8253 processors, and with CPU 2.20 GHz and RAM 256 GB. The operation system is 64-bit Windows 10. All the numerical results are obtained from running the MATLAB R2018b software.

Five real-world databases, including video data, text documents, face images, and audio data, are used in our experiment. The details of these databases, such as dimensionality, and the number of total samples are summarized in Table 1.

• The `AR` database* contains over 4000 color images corresponding to 126 people's faces (70 men and 56 women). Images feature frontal view these faces with different facial expressions, illumination conditions, and occlusions (e.g., sunglasses and scarf). The pictures were taken at the CVC under strictly controlled conditions. No restrictions on wear (clothes, glasses, etc.), make-up, and hairstyle were imposed to participants. A subset of 100 with 26 images per person, i.e., 2600 images are used in our experiment. We crop and scale the images to 120×165 pixels.

• The `Extended YaleB`† dataset contains 5760 single light source images of 10 subjects, each seen under 576 viewing conditions (9 different poses and 64 illumination conditions of each person). The images have normal, sleepy, sad, and surprising expressions. A subset of 38 persons with 64 images per person, i.e., 2432 images are used in the example. We crop and scale the images to 100×100 pixels in our experiment.

• The `Youtube` dataset [33] is a large-scale video classification database. It contains 80 million YouTube video links, which are labeled as 4800 knowledge graph entities. Here, we provide a link of 5020 videos with 1595 persons' labels. In our experiments we have employed up to 90 samples of each class, resulting to a dataset of 124819 feature samples data and 1595 classes.

• The `20Newsgroups` database‡ is a collection of approximately 20000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. The data contains 18846 documents that

*http://www2.ece.ohio-state.edu/˜ aleix/ARdatabase.html

†http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html

‡http://www.cad.zju.edu.cn/home/dengcai/

are organized into 20 different newsgroups, each corresponding to a different topic.

• The original TDT2 (Nist Topic Detection and Tracking) corpus[§] collects during the first half of 1998 and is taken from 6 sources. It consists of 11201 on-topic documents which are classified into 96 semantic categories. In this subset, those documents appearing in two or more categories were removed, and only the largest 30 categories were kept, thus leaving us with 9394 documents in total.

Table 1. Details of the databases: Dimensionality (d), the number of total samples (N), class number (c), and data type (sparse or dense).

Database	Dimensionality (d)	Number of total samples (N)	class number (c)	Type
AR	19800	2600	100	dense
Extended YaleB	10000	2432	38	dense
YouTube	102400	124819	1595	dense
TDT2	36771	9394	30	sparse
20Newsgroups	26214	18846	20	sparse

To show the efficiency of IRRDA and BBCGLDA, we compare them with some state-of-the-art algorithms for large-scale discriminant analysis including RRDA [18], SRDA [23], LDADL [20], and the randomized algorithm in [28]. For simplicity, we call the randomized algorithm in [28] RandLDA. In RandLDA, we set the iteration power $q = 2$ and the over-sampling parameter $p = 20$. The target rank r is chosen in the following way: $r = 100$ if the number of the training sample is less than 1000, and $r = 200$ if the number of the training sample is in the interval $(1000,\ 3500]$, otherwise, $r = 400$. For RRDA, we find that the dimension of W_k and W_{k-1} may not be consistent in running. In our experiments, we let $W_0 = \mathbf{0}_{d\times(c-1)}$, and let U_k be the eigenvectors of T_k associated with the first $c-1$ largest eigenvalues, then the size of W_k is always $d \times (c-1)$. In this case, the stopping criterion $\|W_k - W_{k-1}\|_F < \epsilon$ can be valid. As for SRDA and LDADL, we exploit the *lsrq* package provided by Cai et al [23], to solve the least squares problems.

For all iterative algorithms RRDA, SRDA, LDADL, and BBCGLDA, we set the maximum number of iterations to 20, the tolerance $\epsilon \leq 10^{-4}$, and the regularization parameter is $\alpha = 0.01$. In all the experiments, we randomly pick $n = 30\%N, 50\%N$, and $70\%N$ samples as the training set, and the remaining samples are used as the testing set, where N is the total number of samples. We make use of the nearest neighbor classifier (NN) [34] for classification. Each experiment will be repeated 10 times, and all the numerical results, i.e., the CPU time for training in seconds, the recognition rate, and the standard deviation (Std-Dev), are the mean from the 10 runs.

Example 4.1. In this example, the *AR* database was employed to demonstrate the superiority of IRRDA over RRDA and to show the efficiency of BBCGLDA both in CPU time and recognition rates. We set $\alpha = 0.01$, and Table 2 lists the numerical results of the algorithms.

First, Table 2 shows that RRDA is far inferior to other algorithms both in recognition rate and CPU time, which shows the effectiveness and rationality of our improved algorithm IRRDA of RRDA. Second, in terms of CPU time, RandLDA runs the fastest, followed by BBCGLDA. They are about three times faster than IRRDA, tens times faster than RRDA, and about thirty times faster than SRDA and LDADL. Although RandLDA is the fastest, its recognition rate is much lower than other

[§]http://www.cad.zju.edu.cn/home/dengcai/

algorithms, even about 7% lower than other algorithms when the training sets are small, expect for RRDA. In contrast, our algorithm BBCGLDA has good performance both in CPU time and recognition rate, as the CPU time of BBCGLDA is only $O(0.1)$s slower than RandLDA, and the recognition rate is only $O(0.001)$ lower than the highest recognition rate. In practical application, these numerical gaps can be ignored due to calculation errors and machine performance. Therefore, considering both the recognition rate and CPU time, BBCGLDA has advantages.

On the other hand, the recognition rates of IRRDA, SRDA, LDADL, and BBCGLDA are similar, due to the one to one relationship being established between their explicit solutions, refer to Theorem 4 in this paper, Lemma 4.3 in [20], and Theorem 3.6 in [28]. Moreover, the CPU time of SRDA and LDADL are the slowest. In fact, the *AR* database is a dense matrix, while SRDA and LDADL are aimed at large-scale sparse data.

Furthermore, we find that the more training samples, the higher recognition rates. That is, the recognition rates for 50%N training samples are 5% higher than those for 30%N training samples, and the recognition rates for 70%N training samples are 2% higher than those for 50%N training samples.

Table 2. Numerical results on the AR database for Example 4.1, $d = 19800$, $N = 2600$, $c = 100$, $\alpha = 0.01$.

Algorithms	**CPU Time / s (Recognition Rate ± Std-Dev%)**		
(Methods)	***n=30%N***	***n=50%N***	***n=70%N***
RRDA	14.35 (87.54±1.25%)	14.84 (94.12±0.31%)	15.13 (96.29±0.30%)
IRRDA	1.366 (92.24±1.91%)	1.844 (97.06±0.34%)	3.456 (98.39±0.38%)
BBCGLDA	0.547 (92.19±1.91%)	0.849 (96.95±0.23%)	1.320 (98.31±0.51%)
SRDA	15.10 (91.53±1.95%)	24.47 (96.68±0.13%)	33.76 (98.36±0.37%)
LDADL	15.26 (91.27±2.16%)	24.73 (96.69±0.24%)	34.16 (98.05±0.44%)
RandLDA	0.421 (85.36±1.09%)	0.836 (94.72±0.48%)	1.041 (96.59±0.28%)

Example 4.2. In this example, we do experiments with the *Youtube* databse to show that our algorithm has an absolute advantage in high-dimensional dense large sample database. We do the numerical experiments with $\alpha = 0.01$. The numerical results are listed in Table 3.

Table 3. Numerical results on the Youtube database for Example 4.2, $d = 102400$, $N = 124819$, $c = 1595$, $\alpha = 0.01$. Here, '—' denotes that the CPU time exceeds 1 hour.

Algorithms	***n=30%N=37445***	***n=50%N=62409***	***n=70%N=87373***
RRDA	—	—	—
IRRDA	1645 (94.79±0.21%)	—	—
BBCGLDA	1155 (93.76±0.23%)	1863 (95.95±0.09%)	3521 (96.86±0.12%)
SRDA	—	—	—
LDADL	—	—	—
RandLDA	107.6 (91.82±0.08%)	270.9 (93.03±0.12%)	362.4 (93.31±0.21%)

Table 3 depicts the absolute computational dominant of RandLDA for high-dimensional large-scale dense data in terms of CPU time. When both the dimension d and the number of training samples n are very large, say, $n = 50\%N, 70\%N$, all the algorithms do not work, except for BBCGLDA and RandLDA. RandLDA is 10 times faster than BBCGLDA, while the recognition rates of RandLDA are about 3% lower than that of BBCGLDA. Even with the increase of the training set, the increase of recognition rates is very weak, and the highest recognition rates of RandLDA only reach 93%. In fact, the recognition rate of RandLDA is closely related to the target rank r for RSVD of training matrix X, but the optimal target rank r is difficult to choose, which is a disadvantage of algorithm RandLDA. Next, we will explain the dependence of the algorithm RandLDA on the sampling number, and BBCGLDA is not sensitive to parameter. Therefore, our proposed method BBCGLDA is very powerful for high-dimensional and large-sample dense databases.

Example 4.3. In this example, we try to show the weak influence of the regularization parameter for our proposed algorithm BBCGLDA. Note that except for RandLDA, other algorithms including BBCGLDA, RRDA, SRDA, and LDADL have the regularization parameter. We set the regularization parameter $\alpha = \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$ and analyze the numerical performance of these algorithms with the change of regularization parameters. Since RandLDA also needs to determine the target rank r for RSVD of training matrix X, we set the target rank $r = \{50, 80, 120, 150, 180\}$ and then focus on its numerical results. The test data is *Extended YaleB*. In order to observe the numerical results more clearly, we plot the figures of recognition rates VS regularization parameters, and CPU time VS regularization parameters for $n = \{30\%N,\ 50\%N,\ 70\%\}$ in Figure 1.

According to Figure 1, the recognition rate of RRDA is the worst, whose recognition rates of RRDA are about 10% lower than BBCGLDA, IRRDA, SRDA and LDADL, and the CPU time of RRDA is about 3 times slower than IRRDA, which both again emphasize the improved method for RRDA proposed in our paper. Next, we focus on the five algorithms except RRDA, namely, BBCGLDA, IRRDA, SRDA, LDADL, and RandLDA.

The first line of Figure 1 shows that the recognition rates of BBCGLDA, IRRDA, SRDA, and LDADL are comparable, which again emphasizes the validity and rationality of the Theorem 4. It is found that when $\alpha \leq 0.1$, i.e., $\alpha = \{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$, the recognition rate of the four algorithms is relatively stable and almost unchanged. The recognition rates decreased significantly when $\alpha = 1$, which is caused by the normalization of data. In order to avoid overflow during running algorithms, we first normalize the columns of data, that is, the norm of each column is 1, which means the maximum element of the dataset is far less than 1. Thus, 1 is too large for solving the multivariate linear regression for RLDA as it drowns out the information in the dataset, and one could result in the dynamic of recognition rate. Similarly, 0.1 may be a little large for some high-dimensional normalized data. The second part of Figure 1 shows that the CPU time of SRDA and LDADL is much slower than IRRDA and BBCGLDA, and BBCGLDA is the fastest algorithms of the four LDA algorithms BBCGLDA, IRRDA, SRDA and LDADL with the regularization parameter. This numerical performance is consistent with that of Example 4.1, which again shows the fast and effectiveness of our proposed algorithm. With the increase of the regularization parameter, the CPU time of IRRDA and BBCGLDA decrease gradually. Indeed, IRRDA and BBCGLDA are based on the BCG algorithm. Theorem 4.1 in [32] indicates that the smaller condition number of the coefficient matrix, the faster convergence of BCG, and the condition number decreases with the increase of regularization parameter. Therefore, considering the convergence speed and recognition rate, it is

more appropriate to set $\alpha = 0.01$ for IRRDA and BBCGLDA.

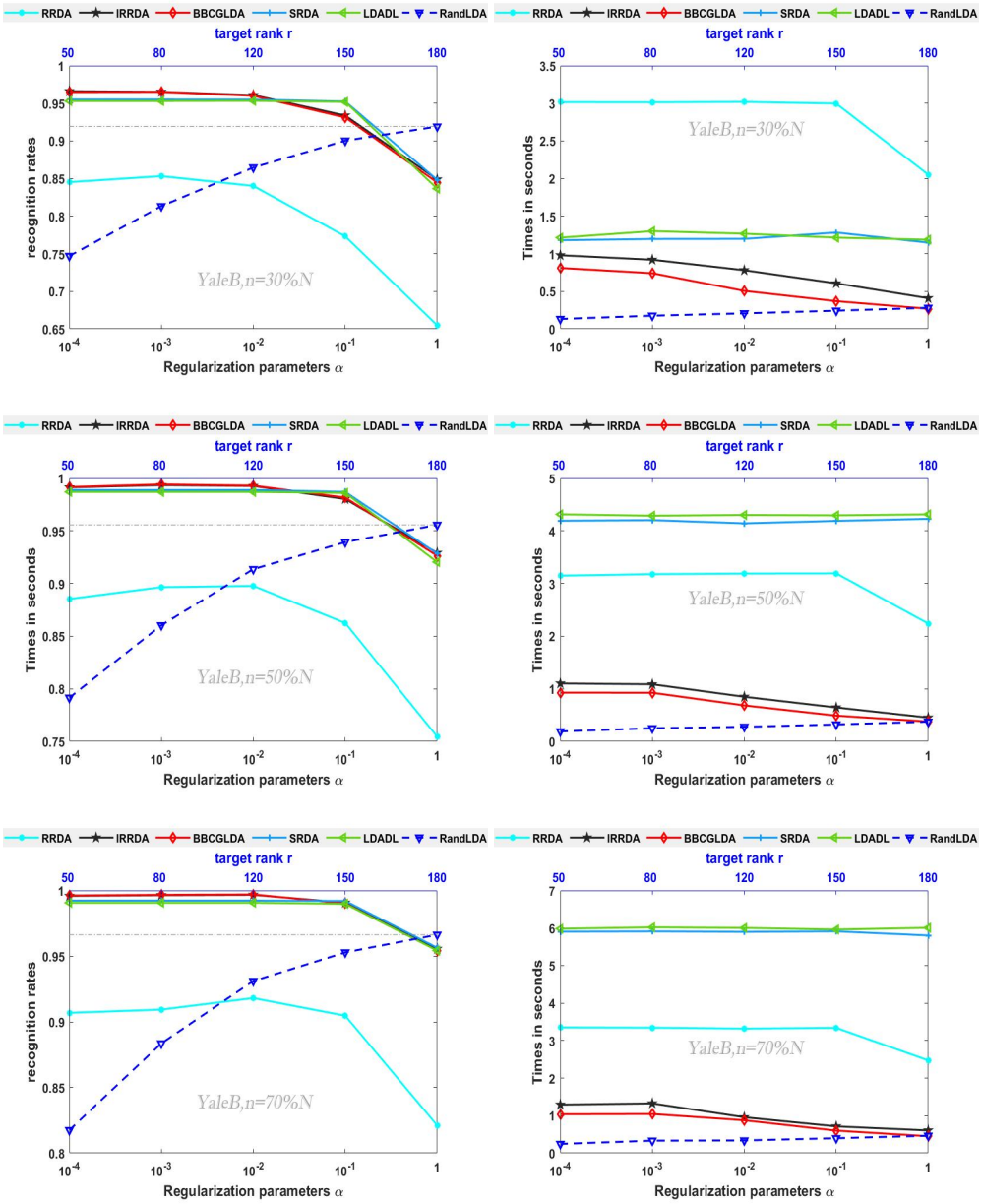

Figure 1. Numerical results for Extended YaleB. The first col is the curves of recognition rates with the regularization parameters or target rank, and the second line is that of CPU times. Each subgraph is a graph of double x-axis, in which the curve of RandLDA is drawn by the value corresponding to the upper x-axis, i.e., the target rank. The curve of RandLDA is the blue dotted line with triangular mark, and the other curves are drawn by the value corresponding to the lower x-axis, i.e., the regularization parameter.

In terms of RandLDA, its recognition rates increase with the increase of target rank. For example, when $n = 30\%N$, its recognition rate increases from about 75% to 92%; when $n = 50\%N$, its

recognition rate increases about from about 79% to 96%; when $n = 70\%N$, its recognition rate increases about from about 82% to 97%, in which the maximum increase of recognition rate is about 17%. These numerical results indicate the recognition rate of RandLDA is seriously affected by the target rank r. However, how to choose the optimal target rank is a difficulty of RandLDA. In addition, it can be observed from the first line of Figure 1 that the highest recognition rates of RandLDA with target rank $r = 180$ are still lower than that of the other four algorithms when $\alpha \leq 0.1$. If one wants to have a higher recognition rate of RandLDA, it needs to increase the target rank r. However, according to the second line of Figure 1, its CPU time will increase with the increase of the target rank r. In other words, RandLDA cannot have both the higher recognition rate and faster time, which is another disadvantage of the algorithm. Note that the recognition rate of BBCGLDA is hardly affected by the regularization parameters, and the convergence speed increases with the increase of the regularization parameters. Therefore, compared with RandLDA, BBCGLDA is a good choice for large-scale discriminant analysis.

Example 4.4. In this example, we aim to show the effectiveness of BBCGLDA for high-dimensional and sparse samples data. The *TDT2* and *20Newsgroups* databases are used in this example. The numerical results are listed in Tables 4 and 5.

Table 4. Numerical results on the *20Newgroups* database for Example 4.4, $d = 26214$, $N = 18846$, $c = 20$.

Algorithms	**CPU Time / s (Recognition Rate ± Std-Dev%)**		
(Methods)	***n=30%N***	***n=50%N***	***n=70%N***
RRDA	24.08 (77.02±1.55%)	29.51 (78.92±1.05%)	35.27 (81.90±0.64%)
IRRDA	4.118 (84.39±0.50%)	6.323 (87.39±0.29%)	13.67 (89.36±0.43%)
BBCGLDA	0.557 (84.36±0.50%)	1.064 (87.38±0.29%)	1.506 (89.31±0.40%)
SRDA	0.620 (85.57±0.42%)	0.873 (88.88±0.28%)	1.184 (90.59±0.36%)
LDADL	0.647 (85.40±0.36%)	0.926 (88.63±0.18%)	1.222 (90.33±0.30%)
RandLDA	1.887 (77.06±0.48%)	2.985 (77.79±0.38%)	4.684 (78.12±0.64%)

Table 5. Example 4.4: Numerical results on the *TDT2* database, $d = 36771$, $N = 9394$, $c = 30$.

Algorithms	**CPU Time / s (Recognition Rate ± Std-Dev%)**		
(Methods)	***n=30%N***	***n=50%N***	***n=70%N***
RRDA	41.73 (96.37±0.11%)	46.80 (95.72±0.60%)	52.27 (95.57±0.67%)
IRRDA	3.835 (95.54±0.21%)	5.328 (96.13±0.12%)	6.765 (96.73±0.26%)
BBCGLDA	0.721 (95.54±0.23%)	1.144 (96.20±0.12%)	1.544 (96.66±0.30%)
SRDA	0.913 (95.72±0.23%)	1.227 (96.48±0.12%)	1.409 (97.09±0.20%)
LDADL	0.946 (95.69±0.18%)	1.270 (96.40±0.14%)	1.454 (96.97±0.18%)
RandLDA	0.914 (96.27±0.19%)	2.379 (96.66±0.19%)	3.124 (96.91±0.37%)

We can see from Tables 4 and 5 that the recognition rates of RRDA and RandLDA can be

comparable with others on the *TDT2* dataset, while those that are lowest are on the *20Newsgroups* dataset, which implies that the applicability of RRDA and RandLDA are not as wide as others. In addition, the standard deviations and recognition rates of the other four algorithms including IRRDA, BBCGLDA, SRDA and LDADL can be comparable, and these results are consistent with the those of the previous examples, which again verifies and determines the correctness of the Theorem 4.

In terms of CPU time, BBCGLDA, SRDA, and LDADL are the fastest; they are about 2–4 times faster than RandLDA, about 6 times faster than IRRDA, and about 30–40 times faster than RRDA. In fact, RRDA computes some eigenvalue decompositions and results in a waste of time. In contrast, IRRDA only needs to copmute one eigenvalue decomposition, and its speed is faster than RRDA. However, it also involves matrix decomposition, and runs lower than other algorithms. Moreover, RandLDA computes an RSVD which will destroy the sparse structure of data. We can find from Tables 4 and 5 that the computing speed of RandLDA for sparse data is not as fast as those of dense data, which is much slower than the other three algorithms BBCGLDA, SRDA, and LDADL, which are based on linear systems. This is because RandLDA needs to compute a randomized singular value decomposition, which will destroy the sparse structure of data matrix.

In general, BBCGLDA can effectively solve not only dense data, but also sparse data. Compared with dense data, it is dozens times faster than SRDA and LDADL, and its recognition rate is higher than that of RandLDA. Compared with sparse data, its CPU time is comparable with SRDA and LDADL, and it is about 3–5 times faster than RandLDA. Considering comprehensively, our algorithm BBCGLDA is a good choice for high-dimensional and large-scale discriminant analysis.

It can be seen from Tables 2–5 that the higher the sample size of the training set, the higher the accuracy rate. Tables 2 and 5 indicate that the accuracy of our algorithm in this paper is higher when $d \gg N$. On the contrary, when d and N are similar in size, the efficiency is lower. It shows that the algorithm in this paper is slow for very large-scale data.

5. Conclusions

RRDA is an efficient algorithm for solving LDA, while its implementation framework has some defects. In this paper, we improve RRDA by re-orthogonalizing the descent direction and putting the eigenvalue decomposition outside the loop. Experiments show the numerical performance of our improved approach IRRDA is better than RRDA in terms of the CPU time, recognition rates, and standard deviations.

To reduce the storage requirement and computational complexity, we replace the centralized matrix $\bar{X}$ with the feature matrix X, and propose a linear system of symmetric and positive equations for solving the RLDA problem. The equivalent of our new model and RLDA is established. Numerical experiments show our proposed algorithm has an absolute advantage in computational efficiency and recognition rates both for high-dimensional large-scale dense data and high-dimensional large-scale sparse data.

Compared with other algorithms, our algorithm is the fastest, but it is slower than the RandLDA algorithm. It can be seen from the experiment that our algorithm cannot process the super large-scale data quickly, so the next idea is to use the random gradient of blocks for experimental research to enhance the processing speed of super large-scale data.

Author contributions

Wenya Shi: Material preparation, Data collection, Analysis, Study conception, Design, Write original draft; Zhixiang Chen: Material preparation, Data collection, Analysis, Study conception, Design. All authors have read and approved the final version of the manuscript for publication.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The first author is supported by [National Natural Science Foundation of China] (Grant numbers [12201075]).

Conflict of 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.

References

1. L. C. Hu, W. S. Zhang, Orthogonal neighborhood preserving discriminant analysis with patch embedding for face recognition, *Pattern Recogn.*, **106** (2020), 107450. http://doi.org/10.1016/j.patcog.2020.107450
2. A. Sasithradevi, S. M. M. Roomi, Video classification and retrieval through spatio-temporal Radon features, *Pattern Recogn.*, **99** (2020), 107099. https://doi.org/10.1016/j.patcog.2019.107099
3. W. Y. Shi, Y. W. Lou, G. Wu, On general matrix exponential discriminant analysis methods for high dimensionality reduction, *Calcolo*, **57** (2020), 18. http://doi.org/10.1007/s10092-020-00366-6
4. K. Fukunaga, *Introduction to statistical pattern classification*, USA: Academic Press, 1990.
5. T. Hastie, R. Tibshirani, J. Friedman, *The elements of statistical learning: Data mining, inference, and prediction*, New York: Springer, 2000.
6. J. W. Chen, S. Y. Xie, H. Jiang, H. Y. Yang, F. P. Nie, A novel k-Means framework via constrained relaxation and spectral rotation, *IEEE T. Neur. Net. Lear.*, 2023, 1–14. http://doi.org/10.1109/TNNLS.2023.3282938
7. Z. X. Li, F. P. Nie, R. Wang, X. L. Li, A revised formation of trace ratio LDA for small sample size problem, *IEEE T. Neur. Net. Lear.*, 2024, 1–7. http://doi.org/10.1109/TNNLS.2024.3362512
8. J. P. Ye, Least squares linear discriminant analysis, *Proceedings of the 24th international conference on machine learning*, 2007, 1087–1093. http://doi.org/10.1145/1273496.1273633
9. R. S. S. Kramer, A. W. Young, A. M. Burton, Understanding face familiarity, *Cognition*, **172** (2018), 46–58. http://doi.org/10.1016/j.cognition.2017.12.005

10. Y. D. Lu, G. Wu, Fast and incremental algorithms for exponential semi-supervised discriminant embedding, *Pattern Recogn.*, **108** (2020), 107530. http://doi.org/10.1016/j.patcog.2020.107530

11. M. Mohri, A. Rostamizadeh, A. Talwalkar, *Foundations of machine learning*, Cambridge: The MIT Press, 2018.

12. G. Wu, T. T. Feng, L. J. Zhang, M. Yang, Inexact implementation using Krylov subspace methods for large scale exponential discriminant analysis with applications to high dimensionality reduction problems, *Pattern Recogn.*, **66** (2017), 328–341. http://doi.org/10.1016/J.PATCOG.2016.08.020

13. C. X. Ren, D. Q. Dai, X. F. He, H. Yan, Sample weighting: An inherent approach for outlier suppressing discriminant analysis, *IEEE T. Knowl. Data En.*, **27** (2015), 3070–3083. http://doi.org/10.1109/TKDE.2015.2448547

14. Y. F. Yu, C. X. Ren, M. Jiang, M. Y. Sun, D. Q. Dai, G. D. Guo, Sparse approximation to discriminant projection learning and application to image classification, *Pattern Recogn.*, **96** (2019), 106963. http://doi.org/10.1016/J.PATCOG.2019.106963

15. L. Wu, C. H. Shen, A. V. D. Hengel, Deep linear discriminant analysis on sher networks: A hybrid architecture for person re-identication, *Pattern Recogn.*, **65** (2017), 238–250. https://doi.org/10.1016/j.patcog.2016.12.022

16. C. Moulin, C. Largeron, C. Ducottet, M. Gery, C. Barat, Fisher linear discriminant analysis for text-image combination in multimedia information retrieval, *Pattern Recogn.*, **47** (2014), 260–269. http://doi.org/10.1016/J.PATCOG.2013.06.003

17. H. S. Ye, Y. J. Li, C. Chen, Z. H. Zhang, Fast fisher discriminant analysis with randomized algorithms, *Pattern Recogn.*, **72** (2017), 82–92. http://dx.doi.org/10.1016/J.PATCOG.2017.06.029

18. L. Zhu, D. S. Huang, A Rayleigh-Ritz style method for large-scale discriminant analysis, *Pattern Recogn.*, **47** (2014), 1698–1708. http://doi.org/10.1016/j.patcog.2013.10.007

19. J. H. Friedman, Regularized discriminant analysis, *J. Am. Stat. Assoc.*, **84** (1989), 165–175. https://doi.org/10.1080/01621459.1989.10478752

20. X. W. Zhang, L. Chen, D. L. Chu, L. Z. Liao, M. K. Ng, R. C. E. Tan, Incremental regularized least squares for dimensionality reduction of large-scale data, *SIAM J. Sci. Comput.*, **38** (2016), B414–B439. http://doi.org/10.1137/15M1035653

21. A. Beck, R. Sharon, A branch and bound method solving the max-min linear discriminant analysis problem, *Optim. Method. Softw.*, **38** (2023), 1031–1057. https://doi.org/10.1080/10556788.2023.2198769

22. J. H. Zhao, H. Y. Liang, S. L. Li, Z. J. Yang, Z. Wang, Matrix-based vs. vector-based linear discriminant analysis: A comparison of regularized variants on multivariate time series data, *Inform. Sciences*, **654** (2024), 119872. https://doi.org/10.1016/j.ins.2023.119872

23. D. Cai, X. F. He, J. W. Han, SRDA: An efficient algorithm for large-scale discriminant analysis, *IEEE T. Knowl. Data En.*, **20** (2008), 1–12. http://dx.doi.org/10.1109/TKDE.2007.190669

24. J. P. Ye, Q. Li, LDA/QR: An efficient and effective dimension reduction algorithm and its theoretical foundation, *Pattern Recogn.*, **37** (2004), 851–854. http://doi.org/10.1016/J.PATCOG.2003.08.006

25. P. Howland, H. Park, Generalizing discriminant analysis using the generalized singular value decomposition, *IEEE T. Pattern Anal.*, **26** (2004), 995–1006. http://doi.org/10.1109/TPAMI.2004.46

26. E. I. G. Nassara, E. Maes, M. Kharouf, Linear discriminant analysis for large-scale data: Application on text and image data, *IEEE*, 2016, 961–964. http://doi.org/10.1109/ICMLA.2016.0173

27. Z. H. Zhang, G. Dai, C. F. Xu, M. I. Jordan, Regularized discriminant analysis, ridge regression and beyond, *J. Mach. Learn. Res.*, **11** (2010), 2199–2228. http://doi.org/10.5555/1756006.1859927

28. W. Y. Shi, G. Wu, New algorithms for trace-ratio problem with application to high-dimension and large-sample data dimensionality reduction, *Mach. Learn.*, 2021. https://doi.org/10.1007/s10994-020-05937-w

29. C. H. Park, H. Park, A relationship between linear discriminant analysis and the generalized minimum squared error solution, *SIAM J. Matrix Anal. A.*, **27** (2005), 474–492. http://doi.org/10.1137/040607599

30. X. H. Chen, T. Chen, Low-rank approximation-based bidirectional linear discriminant analysis for image data, *Multimed. Tools Appl.*, **83** (2024), 19369–19389. https://doi.org/10.1007/s11042-023-16239-3

31. L. Sun, B. Ceran, J. P. Ye, A scalable two-stage approach for a class of dimensionality reduction techniques, *Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining*, 2010, 313–322. http://doi.org/10.1145/1835804.1835846

32. H. Ji, Y. H. Li, A breakdown-free block conjugate gradient method, *BIT Numer. Math.*, **57** (2017), 379–403. http://doi.org/10.1007/s10543-016-0631-z

33. L. Wolf, T. Hassner, I. Maoz, Face recognition in unconstrained videos with matched background similarity, *IEEE*, 2011, 529–534. http://doi.org/10.1109/CVPR.2011.5995566

34. T. Cover, P. Hart, Nearest neighbor pattern classification, *IEEE T. Inform. Theory*, **13** (1967), 21–27. http://doi.org/10.1109/TIT.1967.1053964

AIMS Press