A Convex Programming Approach to the Trace Quotient Problem

Chunhua Shen^{1,2,3}, Hongdong Li^{1,2}, and Michael J. Brooks³

¹ NICTA

² Australian National University

³ University of Adelaide

Abstract. The trace quotient problem arises in many applications in pattern classification and computer vision, e.g., manifold learning, low-dimension embedding, etc. The task is to solve a optimization problem involving maximizing the ratio of two traces, i.e., $\max_W \text{Tr}(f(W))/\text{Tr}(h(W))$. This optimization problem itself is non-convex in general, hence it is hard to solve it directly. Conventionally, the *trace quotient* objective function is replaced by a much simpler *quotient trace* formula, i.e., $\max_W \text{Tr}(h(W)^{-1}f(W))$, which accommodates a much simpler solution. However, the result is no longer optimal for the original problem setting, and some desirable properties of the original problem are lost.

In this paper we proposed a new formulation for solving the trace quotient problem *directly*. We reformulate the original non-convex problem such that it can be solved by efficiently solving a sequence of semidefinite feasibility problems. The solution is therefore globally optimal. Besides global optimality, our algorithm naturally generates orthonormal projection matrix. Moreover it relaxes the restriction of linear discriminant analysis that the projection matrix's rank can only be at most $c - 1$, where c is the number of classes. Our approach is more flexible. Experiments show the advantages of the proposed algorithm.

1 Introduction

The problem of dimensionality reduction—extracting low dimensional structure from high dimensional data—is extensively studied in pattern recognition, computer vision and machine learning. Many of the dimensionality reduction methods, such as linear discriminant analysis (LDA) and its kernel version, end up with solving a trace quotient problem

$$W^* = \underset{W^T W = \mathbf{I}_d}{\operatorname{argmax}} \frac{\text{Tr } W^T S_\alpha W}{\text{Tr } W^T S_\beta W}, \quad (1)$$

where S_α, S_β are positive semidefinite (p.s.d.) matrices ($S_\alpha \succcurlyeq 0, S_\beta \succcurlyeq 0$), $\mathbf{I}_d$ the $d \times d$ identity matrix¹ and $\text{Tr}(\cdot)$ denoting the matrix trace. $W \in \mathbb{R}^{D \times d}$ is target projection matrix for dimensionality reduction (typically $d \ll D$). In the supervised learning framework, usually S_α represents the distance of different classes while S_β is the distance between data points in the same class. For example, S_α is the “between classes scatter matrix” and S_β is the “within classes scatter matrix” for LDA. By formulating

¹ The dimension of $\mathbf{I}$ is omitted when it can be seen from the context.

the problem of dimensionality reduction in a general setting and constructing S_α and S_β in different ways, we can analyze many different types of data in the above underlying mathematical framework.

Despite the importance of the trace quotient problem, it lacks a direct and globally optimal solution. Usually, as an approximation, the quotient trace

$$\text{Tr}(W^\top S_\alpha W)^{-1} W^\top S_\beta W,$$

(instead of the trace quotient) is used. Such an approximation readily leads to a generalized eigen-decomposition (GEVD) solution, via which a close-form solution is readily available.

It is easy to check that when $\text{rank}(W) = 1$, i.e., W is a vector, then Equation (1) is actually a Rayleigh quotient problem, which can be solved by the GEVD. The eigenvector corresponding to the eigenvalue of largest magnitude gives the optimal W^* .

Unfortunately, when $\text{rank}(W) > 1$, the problem becomes much more complicated. Heuristically, the dominant eigenvectors corresponding to the largest eigenvalues are used to form the W^* . It is believed that the largest eigenvalues contains more useful information. However such a GEVD approach cannot produce the optimal solution to the original optimization problem (1) [1]. Furthermore, the GEVD does not yield an orthogonal projection matrix.

Orthogonal LDA (OLDA) is proposed to compute a set of orthogonal discriminant vectors via the simultaneous diagonalisation of the scatter matrices [2].

In this paper, we proffer a novel semidefinite programming (SDP) based method to solve the trace quotient problem directly, which has the following properties:

- It optimises the original problem (Equation (1)) directly;
- The target low dimensionality is selected by the user and the algorithm guarantees an globally optimal solution since the optimisation is convex. In other words, it is local-optima-free;
- The projection matrix is orthonormal naturally;
- Unlike the GEVD approach to LDA, the data are not necessary to be projected to at most $c - 1$ dimensions with our algorithm solving LDA. c is the number of classes.

To our knowledge, this is the first attempt which directly solves the trace quotient problem and at the same time, a global optimum is deterministically guaranteed.

2 SDP Approach to the Trace Quotient Problem

In this section, we show how the trace quotient is reformulated into an SDP problem.

2.1 SDP Formulation

By introducing an auxiliary variable δ , the problem (1) is equivalent to

$$\text{maximize } \delta \tag{2a}$$

$$\text{subject to } \text{Tr } W^\top S_\alpha W \geq \delta \cdot \text{Tr } W^\top S_\beta W \tag{2b}$$

$$W^\top W = \mathbf{I}_d \quad (2c)$$

$$W \in \mathbb{R}^{D \times d} \quad (2d)$$

The variables we want to optimise here are δ and W . But we are only interested in W with which the value of δ is maximised. This problem is clearly not convex because the constraint (2b) is not convex and (2d) is actually a non-convex rank constraint.

Let us define a new variable $Z \in \mathbb{R}^{D \times D}$, $Z = WW^\top$, and now the constraint (2b) is converted to $\text{Tr}(S_\alpha - \delta S_\beta)Z \geq 0$ under the fact that $\text{Tr} W^\top SW = \text{Tr} SWW^\top = \text{Tr} SZ$. Because Z is a product of matrix W and its transpose, it must be p.s.d.

Overton and Womersley [3] have shown that the set of $\Omega_1 = \{WW^\top : W^\top W = \mathbf{I}_d\}$ is the set of extreme points of $\Omega_2 = \{Z : Z = Z^\top, \text{Tr} Z = d, 0 \preceq Z \preceq \mathbf{I}\}$.² That means, as constraints, Ω_1 is more strict than Ω_2 . Therefore constraints (2c) and (2d) can be relaxed into $\text{Tr} Z = d$ and $0 \preceq Z \preceq \mathbf{I}$, which are both convex. When the cost function is linear and it is subject to Ω_2 , the solution will be at one of the extreme points [4]. Consequently, for linear cost functions, the optimization problems subject to Ω_1 and Ω_2 are exactly equivalent. Moreover, the same nice property follows even when the objective function is a quotient (i.e. fractional programming), which is precisely the case we are dealing with here.

With respect to Z and δ , (2b) is still non-convex: the problem may have locally optimal points. But still the global optimum can be efficiently computed via a sequence of convex feasibility problems.

By observing that the constraint is linear if δ is known, we can convert the optimization problem into a set of convex feasibility problems. A *bisection search strategy* is adopted to find the optimal δ . This technique is widely used in *fractional programming*.

Let δ^* denote the unknown optimal value of the cost function. Given $\delta_* \in \mathbb{R}$, if the convex feasibility problem³

$$\text{find } Z \quad (3a)$$

$$\text{subject to } \text{Tr}(S_\alpha - \delta_* S_\beta)Z \geq 0 \quad (3b)$$

$$\text{Tr} Z = d \quad (3c)$$

$$0 \preceq Z \preceq \mathbf{I} \quad (3d)$$

is feasible, then we have $\delta^* \geq \delta_*$. Otherwise, if the above problem is infeasible, then we can conclude $\delta^* < \delta_*$. This way we can check whether the optimal value δ^* is smaller or larger than a given value δ_* . This observation motivates a simple algorithm for solving the fractional optimisation problems using bisection search, which solves a convex feasibility problem at each step. Algorithm 1 shows how it works.

At this point, a question remains to be answered: are constraints (3c) and (3d) equivalent to constraints (2c) and (2d) for the feasibility problem? Essentially the feasibility problem is equivalent to

$$\text{maximize } \text{Tr}(S_\alpha - \delta_* S_\beta)Z \quad (4a)$$

² Our notation is used here.

³ The feasibility problem has no cost function. The objective is to check whether the intersection of the convex constraints is empty.

Algorithm 1. Bisection search

Require: δ_L : Lower bounds of δ ; δ_U : Upper bound of δ and the tolerance $\sigma > 0$.

```

while  $\delta_U - \delta_L > \sigma$  do
   $\delta = \frac{\delta_L + \delta_U}{2}$ .
  Solve the convex feasibility problem described in (3a)–(3d).
  if feasible then
     $\delta_L = \delta$ ;
  else
     $\delta_U = \delta$ .
  end if
end while

```

$$\text{subject to} \quad \mathbf{Tr} \, Z = d \quad (4b)$$

$$0 \preceq Z \preceq \mathbf{I} \quad (4c)$$

If the maximum value of the cost function is non-negative, then the feasibility problem is feasible. Conversely, it is infeasible. Because this cost function is linear, we know that Ω_1 can be replaced by Ω_2 , i.e., constraints (3c) and (3d) are equivalent to (2c) and (2d).

Note that constraint (3d) is not in the standard form of SDP. It can be rewritten into the standard form as

$$\begin{bmatrix} Z & \mathbf{0} \\ \mathbf{0} & Q \end{bmatrix} \succeq 0, \quad (5a)$$

$$Z + Q = \mathbf{I}, \quad (5b)$$

where the matrix Q acts as a slack variable. Now the problem can be solved using standard SDP packages such as CSDP [5] and SeDuMi [6]. We use CSDP in all of our experiments.

2.2 Recovering W from Z

From the covariance matrix Z learned by SDP, we can recover the output W by eigen-decomposition. Let V_i denote the i^{th} eigenvector, with eigenvalue λ_i . Let $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_D$ be the sorted eigenvalues. It is straightforward to see that $W = \mathbf{diag}(\sqrt{\lambda_1}, \sqrt{\lambda_2}, \dots, \sqrt{\lambda_D}) V^T$, where $\mathbf{diag}(\cdot)$ is a square matrix with the input as its diagonal elements. To obtain a $D \times d$ projection matrix, the smallest $D - d$ eigenvalues are simply truncated.

This is the general treatment for recovering a low dimensional projection from a covariance matrix, e.g., principal component analysis (PCA). In our case, this procedure is precise, i.e., there is no information loss. This is obvious: λ_i , the eigenvalues of $Z = WW^T$, are the same as the eigenvalues of $W^T W = \mathbf{I}_d$. That means, $\lambda_1 = \lambda_2 = \dots = \lambda_d = 1$ and the left $D - d$ eigenvalues are all zeros. Hence in our case we can simply stack the first d leading eigenvectors to obtain W .

2.3 Estimating Bounds of δ

The bisection search procedure requires a low bound and an upper bound of δ . The following theorem from [3] is useful for estimating the bounds.

Theorem 1. Let $S \in \mathbb{R}^{D \times D}$ be a symmetric matrix, and $\mu_1^S \geq \mu_2^S \geq \dots \geq \mu_D^S$ be the sorted eigenvalues of S from largest to smallest, then $\max_{W^\top W = \mathbf{I}_d} \text{Tr } W^\top S W = \sum_{i=1}^d \mu_i^S$.

Refer to [3] for the proof. This theorem can be extended to obtain the following corollary (following the proof for Theorem 1):

Corollary 1. Let $S \in \mathbb{R}^{D \times D}$ be a symmetric matrix, and $\nu_1^S \leq \nu_2^S \leq \dots \leq \nu_D^S$ be its sorted eigenvalues from smallest to largest, then $\min_{W^\top W = \mathbf{I}_d} \text{Tr } W^\top S W = \sum_{i=1}^d \nu_i^S$.

Therefore, we estimate the upper bound of δ :

$$\delta_U = \frac{\sum_{i=1}^d \mu_i^{S_\alpha}}{\sum_{i=1}^d \nu_i^{S_\beta}}. \quad (6)$$

In the trace quotient problem, both S_α and S_β are p.s.d. This is equivalent to say, all of their eigenvalues are non-negative. Recall that the denominator of (6) could be zeros and $\delta_U = +\infty$. This occurs when the d smallest eigenvalues of S_β are all zeros. In this case, $\text{rank}(S_\beta) \leq D - d$. For LDA, $\text{rank}(S_\beta) = \min(D, N)$. Here N is the number of training data. When $N \leq D - d$, which is termed the *small sample problem*, δ_U is invalid.

A PCA data prep-processing can always be performed to remove the null space of the covariance matrix of the data, such that δ_U becomes valid.

A lower bound of δ is then

$$\delta_L = \frac{\sum_{i=1}^d \nu_i^{S_\alpha}}{\sum_{i=1}^d \mu_i^{S_\beta}}. \quad (7)$$

Clearly $\delta_L \geq 0$.

3 Related Work

The closest work to ours is [1] in the sense that it also proposes a method to solve the trace quotient directly. [1] finds the projection matrix W in the Grassmann manifold. Compared with optimization in the Euclidean space, the main advantage of optimization on the Grassman manifold is fewer variables. Thus the scale of the problem is smaller. There are major differences between [1] and our method: (i) [1] optimises $\text{Tr } W^\top S_\alpha W - \delta \cdot \text{Tr } W^\top S_\beta W$ and they have no a principled way to determine the optimal value of δ . In contrast, we optimize the trace quotient function itself and a deterministic bisection search guarantees the optimal δ ; (ii) The optimization in [1] is non-convex (difference of two quadratic functions). Therefore it might become trapped into a local maximum, while our method is globally optimal.

Xing et al. [7] propose a convex programming approach to maximize the distances between classes and simultaneously to clip (but not to minimis) the distances within classes. Unlike our method, in [7] the rank constraint is not considered. Hence [7] is metric learning but not necessary a dimensionality reduction method. Furthermore, although the formulation of Xing et al. is convex, it is not SDP. It is more computationally expensive and general-purpose SDP solvers are not applicable. SDP (or other convex programming) is also used in [8,9] for learning a distance metric.

4 Experiments

In this work, we consider optimizing the LDA criterion using the proposed SDP approach. S_α is the “between classes scatter matrix” and S_β is the “within classes scatter matrix”. However, there are many different ways of constructing S_α and S_β , e.g., the general methods considered in [7].

UCI data Firstly, we test whether the optimal value of the cost function, δ^* obtained by our SDP bisection search, is indeed larger than the one obtained by GEVD (conventional LDA). In all the experiments, the tolerance $\sigma = 0.1$. In this experiment, two datasets (“iris” and “wine”) from UCI machine learning repository [10] are used. We randomly sample 70% of the data each time and run 100 tests for the two datasets. The target low dimension d is set to 2. Figure 1 plots the difference of δ^* obtained by two approaches. Indeed, SDP consistently yields larger δ^* than LDA does.

To see the difference between these two approaches, we project the “wine” data into 2D space and plot the projected data in Figure 2. As can be seen, the SDP algorithm has successfully brought together the points in the same class, while keeping dissimilar ones apart (for the original data and the PCA projected data, different classes are entangled together.) While both of the two discriminant projection algorithms can separately the data successfully, SDP intentionally finds a projection of the data onto a straight line that maintains the separation of the clusters. Xing et al. [7] have reported very similar observations with their convex metric learning algorithm.

To test the influence of the SDP algorithm for classification, again we randomly select 70% of the data for training and the left 30% for testing. Both of the two data are projected to 2D. Results with a k NN classifier ($k = 1$) are collected. We run 100 tests for each data set. For the “iris” data, SDP is slightly better (test error 3.71% ($\pm 2.74\%$)) than LDA (test error 4.16% ($\pm 2.56\%$)). While for the “wine” data, LDA is better than SDP (1.53% ($\pm 1.62\%$) against 8.45% ($\pm 4.30\%$)). It means that a larger LDA cost δ^* does not necessarily produce better classification accuracy because: (i) The LDA criterion is not *directly* connected with the classifier’s performance. It is somewhat a heuristic criterion; (ii) As an example, Figure 2 indicates that SDP might *over-fit* the training data. When LDA already well separates the data, SDP aligns the data into a line for larger δ ; (iii) With noisy training data, LDA can denoise by truncating the eigenvectors corresponding to the smaller eigenvalues, similar to what PCA does. During the learning of SDP, it takes noises into consideration as well, which appears to be harmful. We believe that some regularization to the LDA criterion would be beneficial. Also, other different criteria might perform differently in terms of *over-fitting*. These topics remain future research directions.

Fig. 1. The optimal value of δ obtained by our SDP approach (δ_{SDP}) minus the value obtained by the conventional LDA (δ_{LDA}). For all the runs, δ_{SDP} is larger than δ_{LDA} .

USPS handwritten digits data Experiments are also conducted on the full USPS data set. The US Postal (USPS) handwritten digit data-set is derived from a project on recognizing handwritten digits on envelopes. The digits were down sampled to 16×16 pixels. The training set has 7291 samples, and the test set has 2007 samples. The test set is rather difficult: the error rate achieved by human is 2.5% [11].

In the first experiment, we only use 7291 training digits. 70% are randomly selected for training and the other 30% for testing. The data are linearly mapped from 256D to 55D using PCA such that 90.14% of the energy is preserved. Then for LDA, we map them to 9D (because they are totally ten classes). SDP's target low dimension is 50D. We run the experiments 20 times. The results are somewhat surprising. The 1NN classification (i.e, nearest neighbor) test error for LDA is $6.99\% \pm 0.49\%$. SDP achieves much better performance: a 1NN test error $2.79\% \pm 0.27\%$. Note that if we set the target low dimension to 9D for SDP, SDP performs worse than LDA does.

In the second experiment, we use 7291 training data for training and USPS' 2007 test data for testing. Again at first they are mapped to 55D using PCA. LDA reduces

Fig. 2. (1) Original data (first two dimensions are plotted); (2) Projected to 2D with PCA; (3) Projected to 2D with LDA; (4) Projected to 2D with SDP

the dimensionality to 9D and SDP to 54D. LDA has a 1NN test error 10.36% while our SDP achieves a 5.13% test error. Note that in these experiments we have not tuned all the parameters carefully.

5 Conclusion

We have proposed a new formulation for directly solving the trace quotient problem. It is based on SDP, combining with a bisection search approach for solving a fractional programming, which allow us to derive a guaranteed globally optimal algorithm.

Compared with LDA, the algorithm also relaxes the restriction of linear discriminant analysis that the projection matrix's rank can only be at most $c - 1$. In the USPS classification experiment, it shows that this restriction might significantly affect the LDA's performance. Our experiments have validated the advantages of the proposed algorithm.

Acknowledgements

National ICT Australia (NICTA) is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council.

References

1. Yan, S., Tang, X.: Trace quotient problems revisited. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 232–244. Springer, Heidelberg (2006)
2. Ye, J., Xiong, T.: Null space versus orthogonal linear discriminant analysis. In: Proc. Int. Conf. Mach. Learn., Pittsburgh, Pennsylvania, pp. 1073–1080 (2006)
3. Overton, M.L., Womersley, R.S.: On the sum of the largest eigenvalues of a symmetric matrix. *SIAM J. Matrix Anal. Appl.* 13(1), 41–45 (1992)
4. Overton, M.L., Womersley, R.S.: Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices. *Math. Program* 62, 321–357 (1993)
5. Borchers, B.: CSDP, a C library for semidefinite programming. *Optim. Methods and Software* 11, 613–623 (1999)
6. Sturm, J.F.: Using SeDuMi 1.02, a matlab toolbox for optimization over symmetric cones (updated for version 1.05). *Optim. Methods and Software* 11-12, 625–653 (1999)
7. Xing, E., Ng, A., Jordan, M., Russell, S.: Distance metric learning, with application to clustering with side-information. In: Proc. Adv. Neural Inf. Process. Syst., MIT Press, Cambridge (2002)
8. Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: Proc. Adv. Neural Inf. Process. Syst. (2005)
9. Globerson, A., Roweis, S.: Metric learning by collapsing classes. In: Proc. Adv. Neural Inf. Process. Syst. (2005)
10. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
11. Simard, P., LeCun, Y., Denker, J.S.: Efficient pattern recognition using a new transformation distance. In: Proc. Adv. Neural Inf. Process. Syst., pp. 50–58. MIT Press, Cambridge (1993)