

Gaussian Kernel Optimization for Pattern Classification

Jie Wang^a, Haiping Lu^{b*}, K. N. Plataniotis^b, Juwei Lu^c

^a *Epson Edge, 3771 Victoria Park Avenue, Toronto, M1W 3Z5, Canada*

^b *The Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto, 10 King's College Road, Toronto, M5A 3G4, Canada*

^c *Vidient Systems, Inc., 4000 Burton Dr., Santa Clara, CA 95054, USA*

Abstract

This paper presents a novel algorithm to optimize the Gaussian kernel for pattern classification tasks, where it is desirable to have well-separated samples in the kernel feature space. We propose to optimize the Gaussian kernel parameters by maximizing a classical class separability criterion, and the problem is solved through a quasi-Newton algorithm by making use of a recently proposed decomposition of the objective criterion. The proposed method is evaluated on five data sets with two kernel-based learning algorithms. The experimental results indicate that it achieves the best overall classification performance, compared with three competing solutions. In particular, the proposed method provides a valuable kernel optimization solution in the severe small sample size scenario.

Key words: Gaussian kernel, kernel optimization, pattern classification, small sample size

1 Introduction

The kernel machine technique has been widely used to tackle complicated classification problems [1–5] by a nonlinear mapping from the original input space to a kernel feature space. Although in general the dimensionality of the kernel feature space could be arbitrarily large or even infinite, which makes direct analysis in this space very difficult, the nonlinear mapping can be specified implicitly by replacing the dot products in the kernel feature space with a

* Corresponding author. Tel:1-416-978-6845; Fax: 1-416-978-4425 Email: haiping@comm.toronto.edu

kernel function defined in the original input space. Therefore, the key task of a kernel-based solution is to generalize the linear representation in the form of dot products. Existing kernel-based algorithms include the kernel principal component analysis (KPCA)[6], generalized discriminant analysis (GDA)[7] and kernel direct discriminant analysis (KDDA)[8]. In kernel-based learning algorithms, choosing an appropriate kernel, which is a model selection problem [4], is crucial to ensure good performance since the geometrical structure of the mapped samples is determined by the selected kernel and its parameters. Thus, this paper focuses on kernel optimization for pattern classification in a supervised setting, where labeled data are available for training.

For classification purposes, it is often anticipated that the linear separability of the mapped samples is enhanced in the kernel feature space so that applying traditional linear algorithms in this space could result in better performance compared to those obtained in the original input space. However, this is not always the case [9]. If an inappropriate kernel is selected, the classification performance of kernel-based methods can be even worse than that of their linear counterparts. Therefore, selecting a proper kernel with good class separability plays a significant role in kernel-based classification algorithms.

In the literature, there are three approaches to kernel optimization. First, cross validation is a commonly used method but it can only select kernel parameters from a set of discrete values defined empirically, which may or may not contain the optimal or even suboptimal parameter values. Furthermore, cross validation is a costly procedure and it can only be performed when sufficient training samples are available. Thus, it may fail to apply when there are only very limited number of training samples available, which is often encountered in realistic applications such as face recognition. The second approach is to directly optimize the kernel matrix. In [10], the kernel matrix is optimized by maximizing the margin in the kernel feature space via a Semi-Definite Programming technique. Weinberger et al. [11] proposed to learn the kernel matrix by maximizing the variances in the kernel feature space, and their method performed well for manifold learning but not for classification. In [12], a measure named as “alignment” is proposed to adapt the kernel matrix to sample labels, and a series of algorithms are derived for two-class classification problems. Their extensions to multi-class classification problems are usually performed by decomposing the problem into a series of two-class problems through the so-called “one-vs-all” or “one-vs-one” schemes [13,14]. These methods usually result in different classifiers/feature extractors for different class pairs. This complicates the implementation and increases the computational demand and memory requirement in both training and testing, especially for a large number of classes. In general, direct optimization of the kernel matrix operates in the so-called transductive setting and requires the availability of both training and test data in the learning process. This is unrealistic in most, if not all, realtime applications, in addition to its high computational demand in testing.

The third approach is to optimize the kernel function rather than the kernel matrix. In [15–18], a radius-margin quotient is used as a criterion to tune kernel parameters for the support vector machine (SVM) classifier, and it is applicable to two-class classification problems only. Xiong et al. [9] proposed to optimize a kernel function in the so-called empirical feature space by maximizing a class separability measure defined as the ratio between the trace of the between-class scatter matrix and the trace of the within-class scatter matrix, which corresponds to the class separability criterion J_4 in [19]. Promising results have been reported on a set of two-class classification problems. However, its performance on more general multi-class classification problems is not studied. Furthermore, as pointed out in [19], the J_4 criterion is dependent on the coordinate system. In addition, it needs a set of samples to form the so-called empirical core set. This limits its use in the severe small sample size (SSS) scenario, where there are only a very small number of (e.g., two) samples per class available for training.

Among the three approaches, the third one is the most principled one since it directly optimizes the kernel function, which defines the mapping from the input space to the kernel feature space. Therefore, in this paper, we take this approach and propose a kernel optimization algorithm by maximizing the J_1 class separability criterion in [19], defined as the trace of the ratio between the between-class scatter matrix and the within-class scatter matrix. This criterion is equivalent to the criterion used in the classical Fisher’s discriminant analysis [19,20], and it is invariant under any non-singular linear transformation. We focus on optimizing the Gaussian kernel since it is a widely used kernel with success in various applications. The optimization is solved using a Newton-based algorithm, based on the decomposition introduced recently in [21]. The proposed solution works for multi-class problems and it is applicable in the severe SSS scenario as well. Furthermore, although an isotropic Gaussian kernel (with a single parameter) is used to present the algorithm, the proposed method can be readily extended to multi-parameter optimization problems as well as other kernels with differentiable kernel functions.

The rest of the paper is organized as follows. Section 2 presents the optimization algorithm in detail, where the optimization criterion is formulated and a Newton-based searching algorithm is then adopted to optimize the criterion. In Section 3, experimental results on five different data sets, including both two-class and multi-class problems, are reported to show that the proposed kernel optimization method improves the classification performance of two kernel-based classification algorithms, especially in the severe SSS scenario. Comparisons with three methods are included in this section. Finally, a conclusion is drawn in Section 4.

2 Gaussian Kernel Optimization for Pattern Classification

Kernel-based learning algorithms are essentially the implementations of the corresponding linear algorithms in the kernel feature space. Let $\mathbf{Z} = \{\mathbf{Z}_i\}_{i=1}^C$ be a training set containing C classes and each class $\mathbf{Z}_i = \{\mathbf{z}_{ij}\}_{j=1}^{N_i}$ consists of N_i samples, where $\mathbf{z}_{ij} \in \mathcal{R}^J$ and \mathcal{R}^J denotes the J -dimensional real space. Let $\phi(\cdot)$ be a nonlinear mapping from the input space \mathcal{R}^J to a kernel feature space \mathcal{F} [5], i.e., $\phi: \mathbf{z} \in \mathcal{R}^J \rightarrow \phi(\mathbf{z}) \in \mathcal{F}$, where \mathcal{F} denotes the F -dimensional kernel feature space [22]. Then, $\Phi = [\phi(\mathbf{z}_{11}), \dots, \phi(\mathbf{z}_{CN_C})]$ is the corresponding training feature set in \mathcal{F} . A kernel-based pattern classification algorithm aims to find a discriminant function in \mathcal{F} using Φ . Since the dimensionality of \mathcal{F} is much larger than that of \mathcal{R}^J and even infinite, direct analysis in \mathcal{F} is very difficult. Instead, the so-called kernel trick is used where the dot (scalar) products in \mathcal{F} are computed in \mathcal{R}^J via a kernel function $k(\cdot)$: $\phi(\mathbf{z}_i) \cdot \phi(\mathbf{z}_j) = k(\mathbf{z}_i, \mathbf{z}_j)$, where \mathbf{z}_i and \mathbf{z}_j are two vectors in \mathcal{R}^J .

In this paper, a commonly used kernel, the Gaussian kernel, is considered. To simplify the presentation, we focus on an isotropic Gaussian kernel function with a single parameter σ , although the proposed method is applicable to more general Gaussian kernel with multiple parameters as well as other kernels with differentiable kernel functions. An isotropic Gaussian kernel is defined as:

$$k(\mathbf{z}_i, \mathbf{z}_j) = \exp\left(-\frac{\|\mathbf{z}_i - \mathbf{z}_j\|^2}{\sigma^2}\right), \quad (1)$$

where $\exp(\cdot)$ is the exponential function, $\|\cdot\|$ is the Euclidean norm for vectors and σ is the kernel parameter determining the geometrical structure of the mapped samples in the kernel space. As previously discussed, the determination of a proper σ is important in pattern classification. When a very small σ is used ($\sigma \rightarrow 0$), $k(\mathbf{z}_i, \mathbf{z}_j) \rightarrow 0$ for $i \neq j$ and all mapped samples tend to be orthogonal to each other, despite their class labels. In this case, both between-class and within-class variations are very large. On the other hand, when a very large σ is chosen ($\sigma^2 \rightarrow \infty$), $k(\mathbf{z}_i, \mathbf{z}_j) \rightarrow 1$ and all mapped samples converge to a single point. This obviously is not desired in a classification task. Therefore, a too large or too small σ will not result in more separable samples in \mathcal{F} . Instead, selecting an appropriate σ that can maximize the between-class variations as well as minimize the within-class variations is important for good classification performance.

In the following, we will formulate the objective criterion for kernel optimization so that good separability of the input samples in \mathcal{F} is enforced. A Newton-based searching algorithm is adopted to solve the problem.

2.1 The objective criterion for kernel optimization

Our objective is to obtain a kernel that results in good separability in the obtained kernel feature space, which can be measured by the linear class separability of the mapped samples. Let $\tilde{\mathbf{S}}_b$ and $\tilde{\mathbf{S}}_w$ denote the within-class scatter matrix and between-class scatter matrix, respectively, defined in the kernel space as follows:

$$\begin{aligned}\tilde{\mathbf{S}}_b &= \frac{1}{N} \sum_{i=1}^C N_i (\bar{\phi}_i - \bar{\phi})(\bar{\phi}_i - \bar{\phi})^T = \Phi_b \Phi_b^T \\ \tilde{\mathbf{S}}_w &= \frac{1}{N} \sum_{i=1}^C \sum_{j=1}^{N_i} (\phi_{ij} - \bar{\phi}_i)(\phi_{ij} - \bar{\phi}_i)^T = \Phi_w \Phi_w^T,\end{aligned}\tag{2}$$

where $N = \sum_{i=1}^C N_i$ is the total number of training samples, $\Phi_b = [\sqrt{N_1/N}(\bar{\phi}_1 - \bar{\phi}), \dots, \sqrt{N_C/N}(\bar{\phi}_C - \bar{\phi})]$, $\Phi_w = [\sqrt{1/N}\phi(\mathbf{z}_{11}) - \bar{\phi}_1, \dots, \sqrt{1/N}\phi(\mathbf{z}_{CN_C}) - \bar{\phi}_C]$, $\bar{\phi}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} \phi(\mathbf{z}_{ij})$ and $\bar{\phi} = \frac{1}{N} \sum_{i=1}^C \sum_{j=1}^{N_i} \phi(\mathbf{z}_{ij})$.

In [9], the criterion for kernel optimization corresponding to the J_4 criterion in [19] is used: $J_4 = \frac{tr(\tilde{\mathbf{S}}_b)}{tr(\tilde{\mathbf{S}}_w)}$. However, this criterion is seldom used in linear discriminant analysis (LDA) because it is dependent on the coordinate system [19]. In contrast, the criterion $J_1 = tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$ in [19] is invariant under any non-singular linear transformation, and it is a criterion commonly used for LDA in the literature [21]. It is equivalent to the well-known Fisher's discrimination criterion $J_{Fisher} = \frac{|\tilde{\mathbf{S}}_b|}{|\tilde{\mathbf{S}}_w|}$ (J_2 in [19]), where $|\cdot|$ denotes the determinant of a matrix. Therefore, in this work, we adopt this criterion (J_1 in [19]) as the separability measure,

$$J = tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b),\tag{3}$$

where $tr(\cdot)$ denotes the trace of a matrix. For the isotropic Gaussian kernel, the optimal σ is thus optimized by maximizing the criterion J :

$$\sigma_{opt} = \arg \max_{\sigma} J(\sigma) = \arg \max_{\sigma} tr(\tilde{\mathbf{S}}_w^{-1}(\sigma)\tilde{\mathbf{S}}_b(\sigma)).\tag{4}$$

However, to the authors' best knowledge, there is currently no systematic solution to this kernel optimization problem since there is no existing method for the direct optimization of a criterion in such a form.

Recently, a decomposition for this criterion is proposed in [21]. By taking advantage of this development, we are able to solve this problem. Let $\tilde{\lambda}_{bi}$, $\tilde{\lambda}_{wj}$ be the nonzero eigenvalues of $\tilde{\mathbf{S}}_b$ and $\tilde{\mathbf{S}}_w$ sorted in a descending order, i.e., $\tilde{\lambda}_{b1} > \tilde{\lambda}_{b2} > \dots > \tilde{\lambda}_{bp}$, $\tilde{\lambda}_{w1} > \tilde{\lambda}_{w2} > \dots > \tilde{\lambda}_{wq}$, where p and q define the ranks of $\tilde{\mathbf{S}}_b$ and $\tilde{\mathbf{S}}_w$, respectively. Let $\tilde{\mathbf{v}}_{bi}$, $\tilde{\mathbf{v}}_{wj}$ be the corresponding eigenvectors. It

was shown in [21] that $tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$ can be expressed as follows:

$$tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b) = \sum_{i=1}^p \sum_{j=1}^q \frac{\tilde{\lambda}_{bi}}{\tilde{\lambda}_{wj}} (\tilde{\mathbf{v}}_{wj}^T \tilde{\mathbf{v}}_{bi})^2. \quad (5)$$

It should be noted here that $tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$ is a valid metric only when $\tilde{\mathbf{S}}_w$ is non-singular. In the Gaussian kernel feature space, however, $\tilde{\mathbf{S}}_w$ is always singular due to the fact that the number of training samples is limited and dimensionality of the feature space is infinite. In such a case, Eq.(5) is equivalent to evaluate $tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$ in the range space of $\tilde{\mathbf{S}}_w$, i.e., $tr\{(\tilde{\mathbf{V}}_w^T \tilde{\mathbf{S}}_w \tilde{\mathbf{V}}_w)^{-1}(\tilde{\mathbf{V}}_w^T \tilde{\mathbf{S}}_b \tilde{\mathbf{V}}_w)\} = \sum_{i=1}^p \sum_{j=1}^q \frac{\tilde{\lambda}_{bi}}{\tilde{\lambda}_{wj}} (\tilde{\mathbf{v}}_{wj}^T \tilde{\mathbf{v}}_{bi})^2$, where $\tilde{\mathbf{V}}_w = [\tilde{\mathbf{v}}_{w1}, \dots, \tilde{\mathbf{v}}_{wq}]$. For simplicity, the same expression $tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$ is used in this paper to denote the criterion evaluated in the range space of $\tilde{\mathbf{S}}_w$. Removing the null space of $\tilde{\mathbf{S}}_w$ is a commonly used scheme in the LDA literature to solve the singularity problem of $\tilde{\mathbf{S}}_w$ under the SSS scenario [20]. Furthermore, in the Gaussian kernel feature space, the SSS problem becomes extremely severe due to its infinite dimensionality. This makes the estimation of zero or small eigenvalues of $\tilde{\mathbf{S}}_w$ very unstable, giving rise to high variance. Therefore, evaluating the criterion J in the range space of $\tilde{\mathbf{S}}_w$ also helps to reduce the estimation variance.

Due to the fact that the dimensionality of the Gaussian kernel space is infinite, a direct calculation of the eigenvectors and eigenvalues of $\tilde{\mathbf{S}}_b = \Phi_b \Phi_b^T$ and $\tilde{\mathbf{S}}_w = \Phi_w \Phi_w^T$ is impossible. Fortunately, the problem can be solved by an algebraic transform from the eigenvectors of a $C \times C$ matrix $\Phi_b^T \Phi_b$ and the eigenvectors of an $N \times N$ matrix $\Phi_w^T \Phi_w$, respectively [8,23]. Let $\tilde{\lambda}_{bi}$ and $\tilde{\mathbf{e}}_{bi}$ be the i^{th} eigenvalue and eigenvector of $\Phi_b^T \Phi_b$, it can be seen that $(\Phi_b \Phi_b^T)(\Phi_b \tilde{\mathbf{e}}_{bi}) = \tilde{\lambda}_{bi}(\Phi_b \tilde{\mathbf{e}}_{bi})$. Therefore, it can be deduced that $(\Phi_b \tilde{\mathbf{e}}_{bi})$ is the i^{th} eigenvector of $\tilde{\mathbf{S}}_b = \Phi_b \Phi_b^T$. In a similar way, $(\Phi_w \tilde{\mathbf{e}}_{wi})$ is the corresponding eigenvector of $\tilde{\mathbf{S}}_w = \Phi_w \Phi_w^T$, where $\tilde{\lambda}_{wi}$ and $\tilde{\mathbf{e}}_{wi}$ are the i^{th} eigenvalue and eigenvector of $\Phi_w^T \Phi_w$. Thus, $tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$ can be further expressed as,

$$tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b) = \sum_{i=1}^p \sum_{j=1}^q \frac{(\tilde{\mathbf{e}}_{bi}^T \Phi_{bw} \tilde{\mathbf{e}}_{wj})^2}{\tilde{\lambda}_{wj}^2}, \quad (6)$$

where $\Phi_{bw} = \Phi_b^T \Phi_w$, $\tilde{\mathbf{e}}_{bi}$ and $\tilde{\mathbf{e}}_{wi}$ are the eigenvectors of $\tilde{\mathbf{S}}_b^T$ and $\tilde{\mathbf{S}}_w^T$, respectively. The detailed derivation can be found in Appendix I.

Thus, the criterion $J(\sigma)$ can be written as,

$$J(\sigma) = tr(\tilde{\mathbf{S}}_w^{-1}(\sigma)\tilde{\mathbf{S}}_b(\sigma)) = \sum_{i=1}^{p(\sigma)} \sum_{j=1}^{q(\sigma)} F_{ij}(\sigma), \quad (7)$$

where

$$F_{ij}(\sigma) = \frac{(\tilde{\mathbf{e}}_{bi}(\sigma)^T \Phi_{bw}(\sigma) \tilde{\mathbf{e}}_{wj}(\sigma))^2}{\tilde{\lambda}_{wj}(\sigma)^2}. \quad (8)$$

A major difficulty in deriving the expression of $J(\sigma)$ is the determination of the relationship of the ranks of $\tilde{\mathbf{S}}_b$ and $\tilde{\mathbf{S}}_w$ with respect to different σ values, i.e., $p(\sigma)$ and $q(\sigma)$, whose analytical expressions can not be obtained. In this work, we propose a scheme to implicitly determine $p(\sigma)$ and $q(\sigma)$. It is well-known that, for a given data set with N samples and C classes, $p(\sigma) \leq C - 1$ and $q(\sigma) \leq N - C$. Considering the numerical problems in the implementation of the eigenvalue calculation procedure, a realistic solution to estimate the effective rank of the scatter matrices is to compare the eigenvalues against a nonzero tolerance value, denoted in this work as T_0 . The effective ranks of $\tilde{\mathbf{S}}_b$ and $\tilde{\mathbf{S}}_w$, therefore, equal to the number of eigenvalues that are larger than T_0 .

Therefore, Eq.(7) can be rewritten as follows

$$J(\sigma) = \sum_{i=1}^{C-1} \sum_{j=1}^{N-C} \hat{F}_{ij}(\sigma), \quad (9)$$

where

$$\hat{F}_{ij}(\sigma) = \begin{cases} F_{ij}(\sigma) & \text{if } \tilde{\lambda}_{wj}(\sigma) > T_0 \text{ and } \tilde{\lambda}_{bi}(\sigma) > T_0 \\ 0 & \text{Otherwise.} \end{cases} \quad (10)$$

We can then express $\hat{F}_{ij}(\sigma)$ as the product of $F_{ij}(\sigma)$ and step functions $H_{stp}(\tilde{\lambda}_{wj}(\sigma))$ and $H_{stp}(\tilde{\lambda}_{bi}(\sigma))$, i.e.,

$$\hat{F}_{ij}(\sigma) = F_{ij}(\sigma)H_{stp}(\tilde{\lambda}_{wj}(\sigma))H_{stp}(\tilde{\lambda}_{bi}(\sigma)), \quad (11)$$

where

$$H_{stp}(\lambda) = \begin{cases} 1 & \text{if } \lambda > T_0 \\ 0 & \text{Otherwise.} \end{cases} \quad (12)$$

The objective function $J(\sigma)$ is thus expressed as,

$$J(\sigma) = \sum_{i=1}^{C-1} \sum_{j=1}^{N-C} F_{ij}(\sigma)H_{stp}(\tilde{\lambda}_{wj}(\sigma))H_{stp}(\tilde{\lambda}_{bi}(\sigma)). \quad (13)$$

However, this $J(\sigma)$ is not continuously differentiable because the step function $H_{stp}(\lambda)$ is discontinuous. To make $J(\sigma)$ differentiable everywhere, we propose to approximate $H_{stp}(\lambda)$ with a smooth function, the frequency response function of a high-pass B^{th} order Butterworth filter [24] defined as follows:

$$H_B(\lambda) = \frac{1}{\sqrt{1 + \left(\frac{T}{\lambda}\right)^{2B}}}, \quad (14)$$

where T is the cutoff frequency and B denotes the order of the filter. It can be observed that, with a nonzero T and $B > 1/2$, $H_B(\lambda) = 0$ at $\lambda = 0$ and

$H_B(\lambda) > 0$ when $\lambda > 0$. This indicates, by applying the Butterworth filter, for all eigenvalues ($\tilde{\lambda}_{wj} \in [0, \infty)$ and $\tilde{\lambda}_{bi} \in [0, \infty)$), only those that are larger than zero ($\tilde{\lambda}_{wj} \in (0, \infty)$ and $\tilde{\lambda}_{bi} \in (0, \infty)$) will contribute to the calculation of $J(\sigma)$. Figure 1 depicts $H_B(\lambda)$ with $T = 10^{-6}$ and a set of B values. The Butterworth filter is designed to have a smooth and maximally flat pass-band frequency response in magnitude. The value of B determines the transition band. If B approaches infinity and $T = T_0$, $H_B(\lambda)$ becomes $H_{stp}(\lambda)$ except for $\lambda = T$. Therefore, with a sufficiently large B value, $H_B(\lambda)$ is an appropriate choice to approximate the step function $H_{stp}(\lambda)$. Parameter T approximates the nonzero tolerance value T_0 , which can also be viewed as a regularization factor. It has an advantage for the particular problem considered here. It can be seen from Eq.(8) that $\tilde{\lambda}_{wj}$ is a divisor. A smaller $\tilde{\lambda}_{wj}$ value, therefore, results in a larger F_{ij} . In the calculation of $J(\sigma)$, the importance of small eigenvalues of $\tilde{\mathbf{S}}_{wj}$ and the corresponding eigenvectors is thus dramatically exaggerated. In the SSS scenario, it is well-known that the estimation for small eigenvalues of $\tilde{\mathbf{S}}_w$ usually exhibits high variance. Thus, T helps to eliminate unstable small eigenvalues of $\tilde{\mathbf{S}}_w$, resulting in a more stable solution.

Now, we have a continuously differentiable $J(\sigma)$

$$J(\sigma) = \sum_{i=1}^{C-1} \sum_{j=1}^{N-C} F_{ij}(\sigma) H_B(\tilde{\lambda}_{wj}(\sigma)) H_B(\tilde{\lambda}_{bi}(\sigma)) \quad (15)$$

that is ready for optimization.

2.2 The Optimization of $J(\sigma)$

The optimal σ , denoted as σ_{opt} , is obtained by solving the equation $\frac{\partial J(\sigma)}{\partial \sigma} = 0$ using an iterative optimization procedure since an analytical solution does not exist. The detailed derivation of $\frac{\partial J(\sigma)}{\partial \sigma}$ can be found in Appendix II and the complete optimization procedure is depicted in Algorithm 1.

First, we convert the maximization problem into a minimization problem by letting $R(\sigma) = -J(\sigma)$. Then, we adopt the well-known quasi-Newton algorithm with a line search minimization [25,26]. Given a starting point σ_0 , the quasi-Newton optimization method defines the sequence of $\sigma(m)$ as follows,

$$\sigma(m+1) = \sigma(m) - \alpha(m) \frac{g(m)}{U(m)}, \quad (16)$$

where $g(m)$ denotes the first derivative of $R(\sigma)$, i.e., $g(m) = \frac{\partial R}{\partial \sigma}|_{\sigma=\sigma(m)}$ and $U(m)$ denotes the second derivative. In the quasi-Newton algorithm, $U(m)$ is approximated iteratively using the difference equation, i.e., $U(m+1) =$

Algorithm 1 The pseudo-code implementation of the proposed kernel optimization method.

1. Given training data, determine d_0 and set five starting points as $\sigma_0(t) = \{\frac{d_0}{10}, \frac{d_0}{5}, d_0, 5d_0, 10d_0\}, t = 1, \dots, 5$.
 2. Set repetition counter $t=1$.
 3. Set the initial kernel parameter $\sigma(1) = \sigma_0(t)$, the initial second derivative $U(1) = 1$, and the iteration counter $m=1$.
 - (3.1) Compute the gradient $g(m) = \frac{\partial R}{\partial \sigma}|_{\sigma=\sigma(m)}$
 - (3.2) Set $s(m) = -\frac{g(m)}{U(m)}$
 - (3.3) Line search $\alpha(m)$ along $s(m)$
 - (3.4) Update σ : $\sigma(m+1) = \sigma(m) + \alpha(m)s(m)$ and compute the gradient: $g(m+1) = \frac{\partial R}{\partial \sigma}|_{\sigma=\sigma(m+1)}$
 - (3.5) Update U : $U(m+1) = \frac{g(m+1)-g(m)}{\sigma(m+1)-\sigma(m)}$
 - (3.6) if $|g(m+1)| \leq L_{stop}$, set $\sigma_t^* = \sigma(m+1)$, calculate $J(\sigma_t^*)$ and go to step 4; otherwise, set $m=m+1$ and go to step 3.2
 4. if $t = 5$, go to step 5. Otherwise, $t = t + 1$ and go to step 3
 5. Output $\sigma_{opt} = \arg \max_{\sigma_t^*} J(\sigma_t^*)$
-

$\frac{g(m+1)-g(m)}{\sigma(m+1)-\sigma(m)}$ since a direct calculation of $U(m)$ is complex and computationally demanding. $\alpha(m) > 0$ defines the step size. It is calculated through a traditional cubic polynomial line search algorithm [27] so that the updated σ value, $\sigma(m+1)$, can satisfy the *decrease condition*, i.e., $R(\sigma(m+1)) < R(\sigma(m))$, and the *curvature condition*, i.e., $|g(m+1)\frac{g(m)}{U(m)}| < |g(m)\frac{g(m)}{U(m)}|$ [25]. The line search algorithm guarantees the global convergence of the algorithm to a local minimum. The iterations stop if $|g(m)| \leq L_{stop}$, where $L_{stop} = 10^{-6}$.

We also need to set the starting point of the iterative procedure. As discussed earlier, σ^2 can not be too large or too small. In other words, the value of $k(\mathbf{z}_i, \mathbf{z}_j) = \exp\left(\frac{-\|\mathbf{z}_i - \mathbf{z}_j\|^2}{\sigma^2}\right)$, $\mathbf{z}_i \neq \mathbf{z}_j$, should be neither too close to 0 nor too close to 1. This indicates that an appropriate initial σ^2 could be set to be comparable to the value of $\|\mathbf{z}_i - \mathbf{z}_j\|^2$. The initial σ value is thus determined automatically as the average pairwise Euclidean distance between the train-

$$ing\ samples, \text{ denoted as } d_0 = \sqrt{\frac{1}{N^2 - N} \sum_{i=1}^C \sum_{j=1}^{N_i} \sum_{m=1}^C \sum_{n=1}^{N_m} (\mathbf{z}_{ij} - \mathbf{z}_{mn})^T (\mathbf{z}_{ij} - \mathbf{z}_{mn})}.$$

Since the Newton-based algorithm only guarantees to converge to a local minimum, we repeat the optimization procedure five times with different starting points $\sigma_0 = \{\frac{d_0}{10}, \frac{d_0}{5}, d_0, d_0 \times 5, d_0 \times 10\}$, resulting in five different ‘‘optimal’’ σ values, denoted as $\sigma_t^*, t = 1, \dots, 5$. The final σ , σ_{opt} , is thus the one with the largest $J(\sigma)$ value, i.e., $\sigma_{opt} = \arg \max_{\sigma} J(\sigma)$.

3 Experimental Evaluation

In order to evaluate the effectiveness of the proposed method, five sets of experiments are conducted, including both simple two-class problems and more complicated multi-class problems. The first experiment evaluates the sensitivity of the proposed method in terms of the influence of the system parameters on the classification performance. The second experiment compares the proposed method with three competing solutions. In the third experiment, two optimization criteria J and J_4 are compared. The fourth experiment illustrates the performance under a practical severe SSS scenario, where the other methods may fail to apply due to the limited number of training samples. In the last experiment, the optimization of a Gaussian kernel with multiple parameters is examined.

The five different data sets used are the FERET face data, the Ionosphere data, the Breast Cancer data, the Vowel data and the Wine data. The FERET face data set is based on the well-known FERET database [28,29], which is widely used in the face recognition community. For the FERET face data, all face images are preprocessed according to the recommendation of the FERET protocol, which includes (1) images are rotated and scaled so that the centers of the eyes are placed on specific pixels and the image size is 150×130 ; (2) a standard mask is applied to remove non-face portions; (3) histogram equalization is applied and images are normalized to have zero mean and unit standard deviation. Then each image is finally represented as a vector of length 17154. The other four data sets are from the UCI benchmark repository [30]. Details on the configuration of the five data sets are provided in Table 1. For all the five data sets, part of the data samples are randomly selected to form the training set while the rest are treated as the test set. The configuration of the training and test sets is in Table 1 as well. The partition of the training and test sets is repeated 30 times and the averaged results are reported.

Two kernel-based feature extraction methods, KDDA and GDA, are applied to each data set followed by a nearest center classifier to perform the recognition. The recognition performance is evaluated by the best (minimum) classification error rate obtained across different number of features.

KDDA and GDA are two well-known kernel-based feature extraction algorithms. They can be viewed as the implementation of the well-known LDA method in the kernel feature space. The key difference between the two algorithms is in the way the SSS problem is handled. It is well-known that LDA-based solutions often suffer from the SSS problem. Under the SSS scenario where the number of training samples per class is smaller than the dimensionality of the input feature space, the within-class scatter matrix is singular. This makes the optimization of the so-called Fisher’s criterion a difficult prob-

lem. When applying LDA techniques in the kernel feature space, the problem becomes even more severe due to the extremely high dimensionality of the kernel space. Therefore, solving the SSS problem is also a demanding task in the implementation of kernel-based LDA solutions. GDA attempts to solve the SSS problem by removing the null space of the within-class scatter matrix with a PCA routine as is done in the so-called Fisherface method [20]. In order to avoid the loss of discriminatory information residing in the null space of the within-class scatter matrix by GDA, KDDA seeks discriminative information in the intersection of the null space of the within-class scatter matrix and the range space of the between-class scatter matrix [31]. A detailed description of GDA and KDDA can be found in [8,7].

3.1 Experiment I: Sensitivity of the Proposed Solution

In this experiment, empirical performance sensitivity analysis with respect to different starting points σ_0 , order B and threshold T in the Butterworth filter is provided.

Table 2 lists the average σ_{opt}^2 (the optimal σ^2 value) along with the corresponding $J(\sigma_{opt})$ obtained with the five starting points used in this work, with $B = 8$ and $T = 10^{-6}$. It can be observed from Table 2 that with different starting points, the algorithm converges to different local maxima. This indicates that repeating the searching procedure with different starting points is beneficial in locating the best local maximum.

Table 3 shows the average σ_{opt}^2 obtained with $B = \{4, 6, 8, 10\}$, fixing $T = 10^{-6}$. It can be observed that the variations in the obtained σ_{opt}^2 are small. This indicates that B does not significantly affect the system performance.

Table 4 lists the σ_{opt}^2 obtained with four different T values $\{10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}$, fixing $B = 8$. It can be observed that a decreasing T value leads to an increasing σ_{opt}^2 . The reason behind this observation can be seen from Eqs.(8) and (14). These equations indicate that by decreasing the T value, smaller eigenvalues of $\tilde{\mathbf{S}}_w$ are allowed to be included in the summation. While as a divisor, a smaller eigenvalue will result in a larger $J(\sigma)$. Thus, a larger σ^2 value that produces a kernel space with smaller eigenvalues of $\tilde{\mathbf{S}}_w$ will be selected.

After studying the effects of the three parameters on the objective criterion, we examine their effects on the recognition performance. Among these parameters, T has the most effects on recognition. Table 5 lists the classification error rates for KDDA and GDA with various T s. Although a different T results in a different σ_{opt}^2 , the difference in the classification error rate is not significant, with variation less than 2% for most data sets. Nevertheless, it can be observed from Table 5 that, for the Breast Cancer data set, a smaller

T value ($T = 10^{-8}$) is preferred for both GDA and KDDA. On the other hand, for the other four data sets, a larger T is preferred (i.e., $T = 10^{-8}$ gives worse performance) except for the Wine data with KDDA and Vowel data with GDA. As discussed earlier, T determines the effective ranks of $\tilde{\mathbf{S}}_w$ and $\tilde{\mathbf{S}}_b$ and reduces the negative influence of small eigenvalues. Since in the SSS scenario, the estimation of small eigenvalues are usually unstable, giving rise to high variance, a larger T value suggests a stronger regularization and vice versa. Compared to the other data sets, the Breast Cancer data set has less severe SSS problem with only two classes and the most number (> 212) of training samples per class. Therefore, a more stable estimation of small eigenvalues is expected. This suggests that a weaker regularization factor, i.e., a smaller T value, is more suitable. This observation provides guidance in the selection of T : a larger T should be chosen when the SSS problem is more severe and vice versa. In addition, it should be noted that for KDDA on the Ionosphere data, the average error rate varies from 21.11 to 21.13 as σ_{opt}^2 varies from 22.44 to 312.85. This indicates that kernel optimization does not always lead to significant performance improvement and the effects of kernel optimization may depend on the data and/or the kernel-based algorithm. Moreover, d_0 is a reasonably good guess when optimization routine is not desirable due to limited computational power.

3.2 Experiment II: Comparison with Other Kernel Optimization Methods

In this experiment, the proposed kernel optimization method is compared with the cross validation method, the scheme suggested by Xiong et al. in [9] and the target alignment method [12].

For cross validation, among all training samples, one third of the samples for each class are used for validation, i.e., a three-fold cross validation is performed. The best σ^2 value is selected from $\{1, 10, 10^2, 10^3, 10^4, 10^5, 10^6, 10^7, 10^8, 10^9\}$. In Xiong’s method, the objective kernel function is defined as: $k(\mathbf{x}, \mathbf{z}) = q(\mathbf{x})q(\mathbf{z})k_0(\mathbf{x}, \mathbf{z})$, where $k_0(\mathbf{x}, \mathbf{z})$ is the basic kernel and $q(\cdot)$ is of the form $q(\mathbf{x}) = \alpha_0 + \sum_{i=1}^n \alpha_i k_1(\mathbf{x}, \mathbf{a}_i)$ in which $k_1(\mathbf{x}, \mathbf{a}_i) = e^{-\gamma \|\mathbf{x} - \mathbf{a}_i\|^2}$. The components of the set $\{\mathbf{a}_i, i = 1, \dots, n\}$, named as the “empirical cores” in [9], are chosen from half of the training samples for each class. The α_i s are the combination coefficients optimized by maximizing the J_4 criterion. As suggested by the authors, the Gaussian kernel is used as the basic kernel and other system parameters such as γ are chosen under the cross validation framework. Those samples used as “empirical cores” are also used as validation samples. The initial learning rate η_0 and the iteration number are set to 0.01 and 400, respectively. In the target alignment method [12], a transductive learning framework is adopted, in which both training and test data are assumed to be available priori to the actual learning procedure. The objective kernel

matrix is defined as $\mathbf{K} = \sum_i \alpha_i \nu_i \nu_i^T$, where α_i are the parameters to be optimized, ν_i are the eigenvectors of a basic kernel matrix \mathbf{K}_c for both training and test data. In this experiment, \mathbf{K}_c is generated by using a Gaussian kernel function with $\sigma = d_0$. The kernel parameter α_i is optimized by maximizing the so-called target alignment criterion defined as $J_{TA} = \frac{\langle \mathbf{K}, \mathbf{y}\mathbf{y}^T \rangle}{\sqrt{\langle \mathbf{K}, \mathbf{K} \rangle \langle \mathbf{y}\mathbf{y}^T, \mathbf{y}\mathbf{y}^T \rangle}}$, where $\mathbf{y} \in \{-1, 1\}^N$ is the vector of labels of data samples from two classes. The target alignment method [12] is only proposed for two-class classification problems and its extension to multi-class classification problems is still under investigation. Therefore, in the comparison, the target alignment solution is only applied to the Breast Cancer and Ionosphere data sets.

Tables 6 and 7 compare the average classification error rates as well as their standard deviation of KDDA and GDA using the kernel function optimized by the proposed method, cross validation, Xiong’s method and the target alignment method, which are denoted as ‘ J ’, ‘CV’, ‘Xiong’ and ‘TA’ in the tables, respectively. The bold font denotes the best performance across the methods compared. From the results on individual data sets, the proposed method results in better performance in most cases. Xiong’s method demonstrates good performance for two-class classification problems (the Breast Cancer data and Ionosphere data), while its deficiency on multi-class classification problems is obvious. In addition, on two-class problems, the proposed method outperforms the target alignment method except for KDDA on the Breast Cancer data set.

3.3 Experiment III: Comparison of Criteria J and J_4

In this experiment, the classification performance obtained using σ values optimized through the J and J_4 criteria is compared.

The same quasi-Newton method described in Section 2.2 is used for determining the optimal σ value with the J_4 criterion. The calculation of $\frac{\partial J_4}{\partial \sigma}$ can be found in Appendix III. The optimal σ values obtained for criteria J and J_4 are denoted as σ_{opt} and σ_{optJ_4} , respectively.

Table 8 lists the average σ_{opt}^2 and $\sigma_{optJ_4}^2$. It can be observed that the J_4 criterion tends to select larger σ^2 values except for the Ionosphere data set. Figures 2 and 3 depict the $J(\sigma)$, $J_4(\sigma)$ and the classification error rate over different σ^2 values on the Vowel and the Ionosphere data sets. $J(\sigma)$ and $J_4(\sigma)$ are evaluated on the training set while the classification error rate is evaluated on the test set. $J(\sigma)$ and $J_4(\sigma)$ values are normalized to [0,1] for a better visual perception. From the figures, for the Ionosphere data set, the maxima of J_4 and J both coincide with the minima (or near minima) of the test error obtained by KDDA and GDA. This indicates that the maximization of both J_4 and J results in a good kernel space in terms of maximizing the recognition

performance. However, for the Vowel data set, $J_4(\sigma)$ is approximately a non-decreasing curve, which indicates that a large σ^2 value is preferred. In contrast, the maximum of $J(\sigma)$ consistently coincides with the minimum of the test error obtained by both KDDA and GDA. This indicates that J is a more effective criterion than J_4 in measuring the class separability in the kernel space.

Table 9 compares the average error rates and their standard deviation of KDDA and GDA with corresponding σ^2 obtained through the J and J_4 criteria. The results demonstrate that the J criterion achieves similar or better overall performance than the J_4 criterion on various data sets. In particular, a substantial improvement can be observed on the Vowel data set.

3.4 Experiment IV: The Severe Small Sample Size Scenario

The severe SSS scenario, where only one or two samples per class are available for training, is frequently encountered in real-world face recognition applications [32,33]. Kernel optimization becomes more difficult in this case. In this experiment, we study this scenario on the face data set. Among all 960 face images, two images per subject ($2 \times 91 = 182$ images) are randomly selected to form the training set and the remaining 778 images form the test set.

As mentioned earlier, cross validation and Xiong’s method can only be applied when sufficient training samples are available. If the number of available training samples is limited, these two methods may fail to apply. For instance, in the severe SSS scenario considered here, if only two training samples are available for each class, both cross validation and Xiong’s method can not be readily applied for kernel-based LDA algorithms. This is because training an LDA-based machine requires at least two samples per subject but there is no extra sample left for validation or construction of the empirical core set. The target alignment method is inapplicable in the face recognition problem considered here either since it handles two-class problem only. In contrast, the proposed method is still applicable in determining the σ^2 value in the kernel function. Since we know that there is severe SSS problem, a larger $T = 10^{-4}$ is used to perform a stronger regularization with $B = 8$. The optimization with the J_4 criterion and a heuristic parameter selection procedure are also performed for illustration. It was shown in [8] that the optimal σ^2 value may be found in the range of $[5 \times 10^6, 1.5 \times 10^8]$. Thus, we demonstrate the performance by heuristic selection of the following ten σ^2 values $\{10^4, 10^5, 10^6, 5 \times 10^6, 10^7, 4 \times 10^7, 8 \times 10^7, 1.2 \times 10^8, 1.5 \times 10^8, 5 \times 10^8\}$.

Tables 10 and 11 compare the average error rates of KDDA and GDA with different σ^2 . From Tables 10 and 11, the proposed method gives the best performance with the smallest average error rate 28.09% and 30.61% for KDDA

and GDA, respectively. In addition, the proposed J criterion also outperforms the J_4 criterion. Therefore, the proposed scheme provides a valuable solution to kernel optimization in the severe SSS scenario.

3.5 Experiment V: Optimizing Kernels with Multiple Parameters

Although the presentation of this paper is focused on the optimization of an isotropic Gaussian kernel with only one single parameter σ , the proposed method can be readily extended to multi-parameter optimization problems. In this experiment, the proposed solution is applied to optimize a Gaussian kernel with multiple parameters.

A generalized Gaussian kernel function is defined as

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\sum_{k=1}^J (x_k - z_k)^2 / \sigma_k^2\right), \quad (17)$$

where $\mathbf{x} = [x_1, \dots, x_J]^T$ and $\mathbf{z} = [z_1, \dots, z_J]^T$ are two vectors defined in \mathcal{R}^J . Let $\hat{\boldsymbol{\sigma}} = [\sigma_1, \dots, \sigma_J]$ be the parameter vector, the problem thus becomes to optimize $\hat{\boldsymbol{\sigma}}$ by maximizing $J(\hat{\boldsymbol{\sigma}})$. The differences between the optimization of $\hat{\boldsymbol{\sigma}}$ and the optimization of a single σ are in the calculation of the derivative of J and the update of the second derivative $U(m)$ (Algorithm 1, step (3.5)). $\frac{\partial J(\hat{\boldsymbol{\sigma}})}{\partial \hat{\boldsymbol{\sigma}}}$ can be calculated as: $\frac{\partial J(\hat{\boldsymbol{\sigma}})}{\partial \hat{\boldsymbol{\sigma}}} = [\frac{\partial J(\sigma)}{\partial \sigma_1}, \dots, \frac{\partial J(\sigma)}{\partial \sigma_J}]^T$. The approximated Hessian matrix $U(m)$ is updated according to the well-known Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula as follows:

$$U(m+1) = U(m) - \frac{U(m)\delta m \delta m^T U(m)}{\delta(m)^T U(m) \delta(m)} + \frac{\gamma(m)\gamma(m)^T}{\delta(m)^T \gamma(m)}, \quad (18)$$

where $\delta(m) = \hat{\boldsymbol{\sigma}}(m) - \hat{\boldsymbol{\sigma}}(m-1)$, $\gamma(m) = g(m) - g(m-1)$ and $g(m) = \frac{\partial J(\hat{\boldsymbol{\sigma}})}{\partial \hat{\boldsymbol{\sigma}}} |_{\hat{\boldsymbol{\sigma}}=\hat{\boldsymbol{\sigma}}(m)}$.

The experiments in this section are performed on all data sets except the FERET face data set, for which it is impractical to optimize the 17154 kernel parameters (since the dimensionality of the sample space is 17154). Due to its high input dimensionality, for face recognition applications, an isotropic Gaussian kernel with a single parameter is a more appropriate choice. In the experiments, we fix the starting point $\sigma_{j0} = d_0$ for $j = 1, \dots, J$, $T = 10^{-6}$ and $B = 8$. Table 12 shows the error rates and their standard deviation of KDDA and GDA using the generalized kernel function optimized by the proposed method, denoted as J_{Multi} . Through the comparison with the isotropic Gaussian kernel function with a single parameter σ (denoted as J in the table), it can be observed that a generalized Gaussian kernel function can further improve the recognition performance in some cases. In comparison with Xiong's

method and the target alignment method, the improvement in terms of average error rate can be observed in most cases. However, it should be noted that the standard deviation of error rates when using a generalized Gaussian kernel is larger than that when using an isotropic one. This is explained as follows. Given the same number of training samples, compared to optimizing an isotropic Gaussian kernel, optimizing a generalized Gaussian kernel is more complicated since more parameters need to be estimated. Thus, a more severe SSS problem is encountered, which results in a larger estimation variance. How to improve the stability in optimizing multiple parameters is still an unsolved problem to be explored, which is left for future research.

4 Conclusions

In this paper, a novel kernel optimization method has been proposed for pattern classification tasks. We propose to use the classical class separability criterion defined as the trace of the scatter ratio for kernel optimization. Based on the recent decomposition proposed in [21], a differentiable version of the criterion is developed for the isotropic Gaussian kernel. Experimental results on five different data sets, including both two-class and multi-class problems, demonstrate that the proposed method achieves the best overall recognition performance with two kernel-based algorithms, KDDA and GDA, compared with the cross validation method, Xiong’s method and the target alignment method. In particular, it provides a valuable kernel optimization solution in the severe SSS scenario. Furthermore, a multi-parameter kernel optimization problem is solved with the proposed method for illustration. Finally, we would like to point out that although only the Gaussian kernel is discussed in this paper, the approach introduced in this paper can be extended to the optimization of other differentiable kernel functions as well.

Acknowledgments The authors would like to thank the anonymous reviewers for their insightful and constructive comments. The authors would also like to thank the FERET Technical Agent, the U.S. National Institute of Standards and Technology for providing the FERET database. This work is partially supported by a Bell University Lab research grant and the CITO Student Internship Program.

Appendix I – Calculation of $J = tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$

Let $\tilde{\mathbf{S}}_b^T = \Phi_b^T \Phi_b$, $\tilde{\mathbf{S}}_w^T = \Phi_w^T \Phi_w$ and let $\tilde{\mathbf{e}}_{bi}, \tilde{\mathbf{e}}_{wj}, i = 1, \dots, p, j = 1, \dots, q$ be the eigenvectors of $\tilde{\mathbf{S}}_b^T$ and $\tilde{\mathbf{S}}_w^T$ corresponding to eigenvalues λ_{bi} and λ_{wj} , respectively. Then $\tilde{\mathbf{v}}_{bi} = \Phi_b \tilde{\mathbf{e}}_{bi} / \sqrt{\lambda_{bi}}$ and $\tilde{\mathbf{v}}_{wj} = \Phi_w \tilde{\mathbf{e}}_{wj} / \sqrt{\lambda_{wj}}$. (The division by $\sqrt{\lambda_{bi}}$ and $\sqrt{\lambda_{wj}}$ is to normalize the eigenvectors $\tilde{\mathbf{v}}_{bi}$ and $\tilde{\mathbf{v}}_{wj}$). $J = tr(\tilde{\mathbf{S}}_w^{-1}\tilde{\mathbf{S}}_b)$ can be

expressed as follows:

$$\begin{aligned}
J = \text{tr}(\tilde{\mathbf{S}}_w^{-1} \tilde{\mathbf{S}}_b) &= \sum_{i=1}^p \sum_{j=1}^q \frac{\tilde{\lambda}_{bi}}{\tilde{\lambda}_{wj}} (\tilde{\mathbf{v}}_{bi}^T \tilde{\mathbf{v}}_{wj})^2 \\
&= \sum_{i=1}^p \sum_{j=1}^q \frac{\tilde{\lambda}_{bi}}{\tilde{\lambda}_{wj}} \left(\frac{\tilde{\mathbf{e}}_{bi}^T \Phi_b^T \tilde{\mathbf{e}}_{wj} \Phi_w}{\sqrt{\tilde{\lambda}_{bi}} \sqrt{\tilde{\lambda}_{wj}}} \right)^2 \\
&= \sum_{i=1}^p \sum_{j=1}^q \frac{(\tilde{\mathbf{e}}_{bi}^T \Phi_b^T \Phi_w \tilde{\mathbf{e}}_{wj})^2}{\tilde{\lambda}_{wj}^2} \\
&= \sum_{i=1}^p \sum_{j=1}^q \frac{(\tilde{\mathbf{e}}_{bi}^T \Phi_{bw} \tilde{\mathbf{e}}_{wj})^2}{\tilde{\lambda}_{wj}^2}
\end{aligned} \tag{19}$$

(1) Express $\tilde{\mathbf{S}}_b^T$:

It was shown in [8] that $\tilde{\mathbf{S}}_b^T$ can be expressed as follows:

$$\begin{aligned}
\tilde{\mathbf{S}}_b^T &= \frac{1}{N} \mathbf{B} \cdot (\mathbf{A}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{A}_{NC} - \frac{1}{N} (\mathbf{A}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{1}_{NC})) \\
&\quad - \frac{1}{N} (\mathbf{1}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{A}_{NC}) + \frac{1}{N^2} (\mathbf{1}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{1}_{NC}) \cdot \mathbf{B}
\end{aligned} \tag{20}$$

where $\mathbf{B} = \text{diag}[\sqrt{N_1}, \dots, \sqrt{N_C}]$, $\mathbf{K} = (K_{lh})_{l=1, \dots, C}^{h=1, \dots, C}$ is the $N \times N$ Gram matrix consisting of $C \times C$ sub-matrix K_{lh} , $l = 1, \dots, C$, $h = 1, \dots, C$, $K_{lh} = (k(\mathbf{z}_{li}, \mathbf{z}_{hj}))_{i=1, \dots, C_l}^{j=1, \dots, C_h}$ is composed from kernels of the samples from classes \mathbf{Z}_l and \mathbf{Z}_h , $\mathbf{A}_{NC} = \text{diag}[\mathbf{a}_{N_1}, \dots, \mathbf{a}_{N_C}]$ is an $N \times C$ block diagonal matrix, $\mathbf{1}_{NC}$ is an $N \times C$ matrix with all elements equal to 1, and \mathbf{a}_{N_i} is an $N_i \times 1$ vector with all elements equal to $\frac{1}{N_i}$.

(2) Express $\tilde{\mathbf{S}}_w^T$:

$$\begin{aligned}
\tilde{\mathbf{S}}_w^T &= \Phi_w^T \Phi_w = \frac{1}{N} [(\phi_{11} - \bar{\phi}_1), \dots, (\phi_{N_C} - \bar{\phi}_C)]^T [(\phi_{11} - \bar{\phi}_1), \dots, (\phi_{N_C} - \bar{\phi}_C)] \\
&= ((\phi_{ij} - \bar{\phi}_i)^T (\phi_{mn} - \bar{\phi}_m))_{i,m=1, \dots, C}^{j=1, \dots, N_i, n=1, \dots, N_m} \\
&= (\phi_{ij}^T \phi_{mn} - \phi_{ij}^T \bar{\phi}_m - \bar{\phi}_i^T \phi_{mn} + \bar{\phi}_i^T \bar{\phi}_m)_{i,m=1, \dots, C}^{j=1, \dots, N_i, n=1, \dots, N_m}
\end{aligned} \tag{21}$$

Each item can be further expressed as follows:

$$\begin{aligned}
(\phi_{ij}^T \phi_{mn})_{i,m=1, \dots, C}^{j=1, \dots, N_i, n=1, \dots, N_m} &= \mathbf{K} \\
\phi_{ij}^T \bar{\phi}_m &= \frac{1}{N_m} \sum_{n=1}^{N_m} \phi_{ij}^T \phi_{mn} \Rightarrow (\phi_{ij}^T \bar{\phi}_m)_{i,m=1, \dots, C}^{j=1, \dots, N_i, n=1, \dots, N_m} = \mathbf{K} \cdot \mathbf{W} \\
(\bar{\phi}_i^T \phi_{mn})_{i,m=1, \dots, C}^{j=1, \dots, N_i, n=1, \dots, N_m} &= [(\phi_{ij}^T \bar{\phi}_m)_{i,m=1, \dots, C}^{j=1, \dots, N_i, n=1, \dots, N_m}]^T = \mathbf{W}^T \mathbf{K} \\
\bar{\phi}_i^T \bar{\phi}_m &= \frac{1}{N_i N_m} \sum_{j=1}^{N_i} \sum_{n=1}^{N_m} \phi_{ij}^T \phi_{mn} \Rightarrow (\bar{\phi}_i^T \bar{\phi}_m)_{i,m=1, \dots, C}^{j=1, \dots, N_i, n=1, \dots, N_m} = \mathbf{W}^T \cdot \mathbf{K} \cdot \mathbf{W}
\end{aligned}$$

where $\mathbf{W} = \text{diag}[\mathbf{w}_{N_1}, \dots, \mathbf{w}_{N_C}]$ is an $N \times N$ block diagonal matrix and

\mathbf{w}_{N_k} is an $N_k \times N_k$ matrix with all elements equal to $\frac{1}{N_k}$. Therefore

$$\tilde{\mathbf{S}}_w^T = \frac{1}{N}(\mathbf{K} - \mathbf{K} \cdot \mathbf{W} - \mathbf{W}^T \cdot \mathbf{K} + \mathbf{W}^T \cdot \mathbf{K} \cdot \mathbf{W}) \quad (22)$$

(3) Express Φ_{bw} :

$$\begin{aligned} \Phi_{bw} &= \Phi_b^T \Phi_w \\ &= [\sqrt{N_1/N}(\bar{\phi}_1 - \bar{\phi}), \dots, \sqrt{N_C/N}(\bar{\phi}_C - \bar{\phi})]^T \frac{1}{\sqrt{N}}[(\phi_{11} - \bar{\phi}_1), \dots, (\phi_{N_C} - \bar{\phi}_C)] \\ &= \frac{1}{N}(\sqrt{N_i}(\bar{\phi}_i - \bar{\phi})^T)(\phi_{mn} - \bar{\phi}_m)_{i,m=1,\dots,C}^{n=1,\dots,N_m} \\ &= \frac{1}{N}(\sqrt{N_i}(\bar{\phi}_i^T \phi_{mn} - \bar{\phi}_i^T \bar{\phi}_m - \bar{\phi}^T \phi_{mn} + \bar{\phi}^T \bar{\phi}_m))_{i,m=1,\dots,C}^{n=1,\dots,N_m} \end{aligned} \quad (23)$$

Each item can be expressed as follows:

$$\begin{aligned} \bar{\phi}_i^T \phi_{mn} &= \frac{1}{N_i} \sum_{j=1}^{N_i} \phi_{ij}^T \phi_{mn} \quad \Rightarrow (\bar{\phi}_i^T \phi_{mn})_{i,m=1,\dots,C}^{n=1,\dots,N_m} = \mathbf{A}_{NC}^T \cdot \mathbf{K} \\ \bar{\phi}_i^T \bar{\phi}_m &= \frac{1}{N_i N_m} \sum_{j=1}^{N_i} \sum_{n=1}^{N_m} \phi_{ij}^T \phi_{mn} \quad \Rightarrow (\bar{\phi}_i^T \bar{\phi}_m)_{i,m=1,\dots,C}^{n=1,\dots,N_m} = \mathbf{A}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{W} \\ \bar{\phi}^T \phi_{mn} &= \frac{1}{N} \sum_{l=1}^C \sum_{k=1}^{C_l} \phi_{lk}^T \phi_{mn} \quad \Rightarrow (\bar{\phi}^T \phi_{mn})_{i,m=1,\dots,C}^{n=1,\dots,N_m} = \frac{1}{N} \mathbf{1}_{NC}^T \cdot \mathbf{K} \\ \bar{\phi}^T \bar{\phi}_m &= \frac{1}{N N_m} \sum_{l=1}^C \sum_{k=1}^{C_l} \sum_{n=1}^{N_m} \phi_{lk}^T \phi_{mn} \quad \Rightarrow (\bar{\phi}^T \bar{\phi}_m)_{i,m=1,\dots,C}^{n=1,\dots,N_m} = \frac{1}{N} \mathbf{A}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{W} \end{aligned}$$

Therefore,

$$\Phi_{bw} = \frac{1}{N} \mathbf{B} \cdot \left(\mathbf{A}_{NC}^T \cdot \mathbf{K} - \mathbf{A}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{W} - \frac{1}{N} \mathbf{1}_{NC}^T \cdot \mathbf{K} + \frac{1}{N} \mathbf{A}_{NC}^T \cdot \mathbf{K} \cdot \mathbf{W} \right) \quad (24)$$

Appendix II – Calculating the Derivative of $J(\sigma)$

$$\frac{\partial J}{\partial \sigma} = \sum_{i=1}^{C-1} \sum_{j=1}^{N-C} \left(\frac{\partial F_{ij}}{\partial \sigma} H_B(\tilde{\lambda}_{wj}) H_B(\tilde{\lambda}_{bi}) + F_{ij} \frac{\partial H_B(\tilde{\lambda}_{wj})}{\partial \sigma} H_B(\tilde{\lambda}_{bi}) + F_{ij} H_B(\tilde{\lambda}_{wj}) \frac{\partial H_B(\tilde{\lambda}_{bi})}{\partial \sigma} \right) \quad (25)$$

(1) Calculate $\frac{\partial F_{ij}}{\partial \sigma}$:

Since $F_{ij} = \frac{(\tilde{\mathbf{e}}_{bi}^T \Phi_{bw} \tilde{\mathbf{e}}_{wj})^2}{\tilde{\lambda}_{wj}^2}$, $\frac{\partial F_{ij}}{\partial \sigma}$ can be derived as follows:

$$\begin{aligned} \frac{\partial F_{ij}}{\partial \sigma} &= \\ \frac{2\tilde{\mathbf{e}}_{bi}^T \Phi_{bw} \tilde{\mathbf{e}}_{wj}}{\tilde{\lambda}_{wj}^2} &\left[\left(\frac{\partial \tilde{\mathbf{e}}_{bi}}{\partial \sigma} \right)^T \Phi_{bw} \tilde{\mathbf{e}}_{wj} + \tilde{\mathbf{e}}_{bi}^T \left(\frac{\partial \Phi_{bw}}{\partial \sigma} \right) \tilde{\mathbf{e}}_{wj} + \tilde{\mathbf{e}}_{bi}^T \Phi_{bw} \left(\frac{\partial \tilde{\mathbf{e}}_{wj}}{\partial \sigma} \right) - \frac{\tilde{\mathbf{e}}_{bi}^T \Phi_{bw} \tilde{\mathbf{e}}_{wj}}{\tilde{\lambda}_{wj}} \frac{\partial \tilde{\lambda}_{wj}}{\partial \sigma} \right] \end{aligned} \quad (26)$$

According to [34] (Chapter 8, pp.159), the derivative of the eigenvector and the eigenvalue of $\tilde{\mathbf{S}}_b^T$ and $\tilde{\mathbf{S}}_w^T$ can be formulated as follows:

$$\begin{aligned} \frac{\partial \mathbf{e}_{wj}}{\partial \sigma} &= (\tilde{\lambda}_{wj} \mathbf{I} - \tilde{\mathbf{S}}_w^T)^+ \frac{\partial \tilde{\mathbf{S}}_w^T}{\partial \sigma} \tilde{\mathbf{e}}_{wj} \\ \frac{\partial \mathbf{e}_{bi}}{\partial \sigma} &= (\tilde{\lambda}_{bi} \mathbf{I} - \tilde{\mathbf{S}}_b^T)^+ \frac{\partial \tilde{\mathbf{S}}_b^T}{\partial \sigma} \tilde{\mathbf{e}}_{bi} \\ \frac{\partial \lambda_{wj}}{\partial \sigma} &= \tilde{\mathbf{e}}_{wj}^T \frac{\partial \tilde{\mathbf{S}}_w^T}{\partial \sigma} \tilde{\mathbf{e}}_{wj}, \end{aligned} \quad (27)$$

where \mathbf{I} is an identity matrix and \mathbf{A}^+ denotes the pseudo-inverse of \mathbf{A} .

In addition,

$$\begin{aligned} \frac{\partial \tilde{\mathbf{S}}_b^T}{\partial \sigma} &= \frac{1}{N} \mathbf{B} \cdot \left(\mathbf{A}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{A}_{NC} - \frac{1}{N} \cdot \mathbf{A}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{1}_{NC} - \frac{1}{N} \cdot \mathbf{1}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{A}_{NC} \right. \\ &\quad \left. + \frac{1}{N^2} \cdot \mathbf{1}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{1}_{NC} \right) \cdot \mathbf{B} \\ \frac{\partial \tilde{\mathbf{S}}_w^T}{\partial \sigma} &= \frac{1}{N} \left(\frac{\partial \mathbf{K}}{\partial \sigma} - \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{W} - \mathbf{W}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} + \mathbf{W}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{W} \right) \\ \frac{\partial \Phi_{bw}}{\partial \sigma} &= \frac{1}{N} \left(\mathbf{A}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} - \mathbf{A}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{W} - \frac{1}{N} \mathbf{1}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} + \frac{1}{N} \mathbf{A}_{NC}^T \cdot \frac{\partial \mathbf{K}}{\partial \sigma} \cdot \mathbf{W} \right) \cdot \mathbf{B} \end{aligned} \quad (28)$$

where $\frac{\partial \mathbf{K}}{\partial \sigma} = \left\{ \frac{\partial k(\mathbf{z}_{ij}, \mathbf{z}_{mn})}{\partial \sigma} \right\}_{i,m=1,\dots,C}^{j=1,\dots,N_i, n=1,\dots,N_m}$ is an $N \times N$ symmetric matrix and $\frac{\partial k(\mathbf{z}_{ij}, \mathbf{z}_{mn})}{\partial \sigma}$ can be further expressed as

$$\frac{\partial k(\mathbf{z}_{ij}, \mathbf{z}_{mn})}{\partial \sigma} = \frac{2\|\mathbf{z}_{ij} - \mathbf{z}_{mn}\|^2}{\sigma^3} \exp\left(-\frac{\|\mathbf{z}_{ij} - \mathbf{z}_{mn}\|^2}{\sigma^2}\right). \quad (29)$$

(2) Calculate $\frac{\partial H_B(\tilde{\lambda}_{wj})}{\partial \sigma}$ and $\frac{\partial H_B(\tilde{\lambda}_{bi})}{\partial \sigma}$:

$$H_B(\tilde{\lambda}) = \frac{1}{\sqrt{1 + \left(\frac{T}{\tilde{\lambda}}\right)^{2B}}}$$

$$\begin{aligned}
\frac{\partial H_B(\tilde{\lambda}_{wj})}{\partial \sigma} &= \frac{B \cdot (T)^{2B} \cdot \frac{\partial \tilde{\lambda}_{wj}}{\partial \sigma}}{\left(1 + \left(\frac{T}{\tilde{\lambda}_{wj}}\right)^{2B}\right)^{3/2} \cdot (\tilde{\lambda}_{wj})^{2B+1}} \\
\frac{\partial H_B(\tilde{\lambda}_{bi})}{\partial \sigma} &= \frac{B \cdot (T)^{2B} \cdot \frac{\partial \tilde{\lambda}_{bi}}{\partial \sigma}}{\left(1 + \left(\frac{T}{\tilde{\lambda}_{bi}}\right)^{2B}\right)^{3/2} \cdot (\tilde{\lambda}_{bi})^{2B+1}}
\end{aligned} \tag{30}$$

Appendix III – Calculating the Derivative of $J_4(\sigma)$

From [9], for the Gaussian kernel, $tr(\tilde{\mathbf{S}}_b)$ and $tr(\tilde{\mathbf{S}}_w)$ can be expressed as:

$$\begin{aligned}
tr(\tilde{\mathbf{S}}_b) &= \frac{1}{N} \sum_{i=1}^C \frac{1}{N_i} \mathbf{1}_{N_i}^T K_{ii} \mathbf{1}_{N_i} - \frac{1}{N^2} \mathbf{1}_N^T \mathbf{K} \mathbf{1}_N \\
&= \frac{1}{N} \sum_{i=1}^C \frac{1}{N_i} \sum_{m=1}^{N_i} \sum_{n=1}^{N_i} k(\mathbf{z}_{im}, \mathbf{z}_{in}) - \frac{1}{N^2} \sum_{i=1}^C \sum_{j=1}^C \sum_{m=1}^{N_i} \sum_{n=1}^{N_j} k(\mathbf{z}_{im}, \mathbf{z}_{jn}) \\
tr(\tilde{\mathbf{S}}_w) &= 1 - \frac{1}{N} \sum_{i=1}^C \frac{1}{N_i} \mathbf{1}_{N_i}^T K_{ii} \mathbf{1}_{N_i} \\
&= 1 - \frac{1}{N} \sum_{i=1}^C \frac{1}{N_i} \sum_{m=1}^{N_i} \sum_{n=1}^{N_i} k(\mathbf{z}_{im}, \mathbf{z}_{in}) \\
&= 1 - \frac{C}{N} - \frac{2}{N} \sum_{i=1}^C \frac{1}{N_i} \sum_{m=1}^{N_i} \sum_{n=m+1}^{N_i} k(\mathbf{z}_{im}, \mathbf{z}_{in})
\end{aligned} \tag{31}$$

where $\mathbf{1}_N$ denotes an $N \times N$ matrix with each element equal to 1.

Therefore, $\frac{\partial J_4}{\partial \sigma}$ can be expressed as:

$$\frac{\partial J_4}{\partial \sigma} = \frac{\frac{\partial tr(\tilde{\mathbf{S}}_b)}{\partial \sigma} tr(\tilde{\mathbf{S}}_w) - tr(\tilde{\mathbf{S}}_b) \frac{\partial tr(\tilde{\mathbf{S}}_w)}{\partial \sigma}}{tr(\tilde{\mathbf{S}}_w)^2} \tag{32}$$

and

$$\begin{aligned}
\frac{\partial tr(\tilde{\mathbf{S}}_b)}{\partial \sigma} &= \frac{1}{N} \sum_{i=1}^C \frac{1}{N_i} \mathbf{1}_{N_i}^T \left(\frac{\partial \mathbf{K}}{\partial \sigma} \right)_{ii} \mathbf{1}_{N_i} - \frac{1}{N^2} \mathbf{1}_N^T \frac{\partial \mathbf{K}}{\partial \sigma} \mathbf{1}_N \\
\frac{\partial tr(\tilde{\mathbf{S}}_w)}{\partial \sigma} &= -\frac{1}{N} \sum_{i=1}^C \frac{1}{N_i} \mathbf{1}_{N_i}^T \left(\frac{\partial \mathbf{K}}{\partial \sigma} \right)_{ii} \mathbf{1}_{N_i}
\end{aligned} \tag{33}$$

where $\left(\frac{\partial \mathbf{K}}{\partial \sigma} \right)_{ii}$ is an $N_i \times N_i$ sub-matrix of $\frac{\partial \mathbf{K}}{\partial \sigma}$ corresponding to the samples from the i^{th} class.

References

- [1] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.
- [2] B. Schölkopf, C. Burges, A. J. Smola, *Advances in Kernel Methods - Support Vector Learning*, MIT Press, Cambridge, MA, 1999.
- [3] A. Ruiz, P. L. de Teruel, Nonlinear kernel-based statistical pattern analysis, *IEEE Transactions on Neural Networks* 12 (1) (2001) 16–32.
- [4] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, An introduction to kernel-based learning algorithms, *IEEE Transactions on Neural Networks* 12 (2) (2001) 181–201.
- [5] J.Shawe-Taylor, N.Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.
- [6] B. Schölkopf, A. Smola, K. Müller, Nonlinear component analysis as a kernel eigenvalue problem, *Neural Computation* 10 (1999) 1299–1319.
- [7] G. Baudat, F. Anouar, Generalized discriminant analysis using a kernel approach, *Neural Computation* 12 (2000) 2385–2404.
- [8] J.Lu, K.N.Plataniotis, A.N.Venetsanopoulos, Face recognition using kernel direct discriminant analysis algorithms, *IEEE Transactions on Neural Networks* 14 (1) (2003) 117–126.
- [9] H.Xiong, M.N.S.Swamy, M.Omar, Optimizing the kernel in the empirical feature space, *IEEE Transactions on Neural Networks* 16 (2) (2005) 460–474.
- [10] G.R.G.Lanckriet, N.Cristianini, P.Bartlett, L.E.Ghaoui, M.I.Jordan, Learning the kernel matrix with semidefinite programming, *Journal of Machine Learning Research* 5 (2004) 27–72.
- [11] K.Q.Weinberger, F.Sha, L.K.Saul, Learning a kernel matrix for nonlinear dimensionality reduction, in: *Proceedings of 21th International Conference on Machine Learning*, 2004, pp. 106–113.
- [12] N.Cristianini, J.Kandola, A.Elisseff, J.Shawe-Taylor, On kernel target alignment, in: *Proceedings of the Neural Information Processing Systems*, 2001, pp. 367–373.
- [13] L. W. L. Bo, L. Jiao, Feature scaling for kernel fisher discriminant analysis using leave-one-out cross validation, *Neural Computation* 18 (2006) 961–978.
- [14] R. Rifkin, A. Klautau, In defense of one-vs-all classification, *Journal of Machine Learning Research* 5 (2004) 101–141.
- [15] T. Glasmachers, C. Igel, Gradient-based adaptation of general gaussian kernels, *Neural Computation* 17 (10) (2005) 2099–2105.

- [16] C. Gold, P. Sollich, Model selection for support vector machine classification, *Neurocomputing* 55 (1-2) (2003) 221–249.
- [17] S. S. Keerthi, Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms, *IEEE Transactions on Neural Networks* 13 (5) (2002) 1225–1229.
- [18] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, *Machine Learning* 46 (1) (2002) 131–159.
- [19] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd Edition, Academic Press, Boston, 1990.
- [20] P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman, Eigenfaces vs. Fisherfaces: recognition using class specific linear projection, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19 (7) (1997) 711–720.
- [21] A.M.Martinez, M.Zhu, Where are linear feature extraction methods applicable?, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27 (12) (2005) 1934–1944.
- [22] B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, A. Smola, Input space versus feature space in kernel-based methods, *IEEE Transactions on Neural Networks* 10 (5) (1999) 1000 –1017.
- [23] M. A. Turk, A. P. Pentland, Eigenfaces for recognition, *Journal of Cognitive Neuroscience* 3 (1) (1991) 71–86.
- [24] A.V.Oppenheim, A.S.Willsky, S.H.Nawab, *Signals and Systems*, 2nd Edition, Prentice Hall, 1997.
- [25] R. Fletcher, An overview of unconstrained optimization, *Numerical Analysis Report NA/149*.
- [26] S.Boyd, L.Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
- [27] R. Fletcher, *Practical Methods of Optimization*, 2nd Edition, John Wiley and Sons, Inc., 1987.
- [28] P.J.Phillips, H.Wechsler, J.Huang, P.Rauss, The feret database and evaluation procedure for face recognition algorithms, *Image and Vision Computing Journal* 16 (5) (1998) 295–306.
- [29] P.J.Phillips, H.Moon, S.A.Rizvi, P.Rauss, The feret evaluation method for face recognition algorithms, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22 (10) (2000) 1090–1104.
- [30] C.Blake, E.Keogh, C.J.Merz, (1998) UCI repository of machine learning databases, Dept.Inform.Compu.Sci.,University of California, Irvine.
URL <http://www.ics.uci.edu/mllearn>

- [31] J.Lu, K.N.Plataniotis, A.N.Venetsanopoulos, Face recognition using LDA based algorithms, *IEEE Transactions on Neural Networks* 14 (1) (2003) 195–200.
- [32] J.Wang, K.N.Plataniotis, J.Lu, A.N.Venetsanopoulos, On solving the face recognition problem with one training sample per subject, *Pattern Recognition* 39 (9) (2006) 1746–1762.
- [33] J.Wang, K.N.Plataniotis, A.N.Venetsanopoulos, Selecting discriminant eigenfaces for face recognition, *Pattern Recognition Letters* 26 (2005) 1470–1482.
- [34] J.R.Magnus, *Matrix Differential Calculus with Applications in Statistics and Econometrics*, John Wiley and Sons, Inc., 1988.

Table 1

Information on the Test Data Sets

	FERET	Ionosphere	Vowel	BreastCancer	Wine
Dimensionality	17154	34	10	30	13
Number of Classes	91	2	11	2	3
Number of Samples per Class	≥ 8	≥ 126	90	≥ 212	≥ 48
Number of Training Samples per Class	3	40	20	80	15
Number of Test Samples	778	271	770	409	133

Table 2

Average σ_{opt}^2 Obtained with Different Starting Points and Corresponding $J(\sigma_{opt})$

σ_0		BreastCancer	Ionosphere	Vowel	Wine	FERET
$d_0/10$	σ_{opt}^2	1.1×10^4	47.6	9.21	2.6×10^4	2.5×10^6
	$J(\sigma_{opt})$	81.7	65.8	565.6	65.7	136.6
$d_0/5$	σ_{opt}^2	1.7×10^4	53.6	9.2	2.1×10^4	3.0×10^6
	$J(\sigma_{opt})$	70.5	59.6	569.3	68.8	157.6
d_0	σ_{opt}^2	1.1×10^5	38.8	8.93	1.0×10^5	3.0×10^6
	$J(\sigma_{opt})$	73.3	65.9	582.0	57.25	157.6
$d_0 \times 5$	σ_{opt}^2	1.4×10^6	70.6	9.1	6.0×10^6	3.0×10^6
	$J(\sigma_{opt})$	70.5	58.1	559.3	54.6	157.6
$d_0 \times 10$	σ_{opt}^2	1.8×10^7	46.5	8.91	7.2×10^4	2.9×10^6
	$J(\sigma_{opt})$	58.8	66.2	578.25	62.7	157.04

Table 3

Average σ_{opt}^2 Obtained with Different B s

B	BreastCancer	Ionosphere	Vowel	Wine	FERET
4	9.87×10^3	38.96	8.21	2.53×10^4	2.95×10^6
6	1.03×10^4	39.62	8.09	2.62×10^4	3.03×10^6
8	1.10×10^4	37.66	8.33	2.75×10^4	3.08×10^6
10	7.83×10^3	41.28	8.09	2.62×10^4	3.21×10^6

Table 4
Average σ_{opt}^2 Obtained with Different T s

T	BreastCancer	Ionosphere	Vowel	Wine	FERET
10^{-5}	7.12×10^3	22.44	5.56	1.02×10^4	4.77×10^5
10^{-6}	1.10×10^4	37.66	8.33	2.75×10^4	3.08×10^6
10^{-7}	1.75×10^4	105.99	14.80	6.09×10^4	2.37×10^7
10^{-8}	3.03×10^4	312.85	28.72	1.35×10^5	2.07×10^8

Table 5
Average Error Rate (%) of KDDA and GDA with Different T s

	KDDA with $T =$				GDA with $T =$			
	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
BreastCancer	13.73	13.77	13.28	12.64	7.52	6.81	6.18	5.95
Ionosphere	21.11	21.02	20.96	21.13	8.56	9.19	11.35	12.39
Vowel	24.70	24.97	25.55	26.70	16.05	15.19	14.70	15.79
Wine	28.25	26.84	26.69	26.97	24.56	23.91	25.74	25.69
FERET	20.45	21.07	21.11	21.12	21.86	21.96	20.96	22.25

Table 6
Average Error Rate \pm Standard Deviation of KDDA Using the σ^2 Obtained by the Proposed Method, Cross Validation, Xiong's Method and the Target Alignment Method

	BreastCancer	Ionosphere	Vowel	Wine	FERET
$J(\%)$	13.77 ± 2.96	21.02 ± 5.14	24.97 ± 2.01	26.84 ± 2.13	21.07 ± 1.81
$CV(\%)$	11.03 ± 2.53	23.78 ± 6.33	25.23 ± 2.23	26.97 ± 4.88	24.54 ± 4.44
Xiong(%)	11.52 ± 2.28	13.44 ± 6.32	30.6 ± 3.02	29.85 ± 4.18	54.92 ± 9.48
TA(%)	10.6 ± 2.09	22.15 ± 3.02	-	-	-

Table 7
Average Error Rate \pm Standard Deviation of GDA Using the σ^2 Obtained by the Proposed Method, Cross Validation, Xiong's Method and the Target Alignment Method

	BreastCancer	Ionosphere	Vowel	Wine	FERET
$J(\%)$	6.81 ± 1.40	9.19 ± 2.26	15.19 ± 2.04	23.91 ± 4.91	21.96 ± 3.69
$CV(\%)$	7.36 ± 1.23	9 ± 3.66	16.19 ± 3.32	25.29 ± 4.25	21.91 ± 3.30
Xiong(%)	7.61 ± 1.56	11.13 ± 3.89	24.57 ± 3.11	28.62 ± 6.96	27.08 ± 2.78
TA(%)	12.31 ± 0.8	25.6 ± 4.18	-	-	-

Table 8

Average Optimal σ^2 Found by the J Criterion and the J_4 Criterion

	BreastCancer	Ionosphere	Vowel	Wine	FERET
σ_{opt}^2	1.1×10^4	37.66	8.33	2.75×10^4	3.08×10^6
$\sigma_{optJ_4}^2$	5.67×10^8	10.97	1.18×10^5	4.85×10^8	3.43×10^7

Table 9

Average Error Rate \pm Standard Deviation with Optimal σ^2 Found by the J Criterion and the J_4 Criterion

		BreastCancer	Ionosphere	Vowel	Wine	FERET
KDDA (%)	J	13.77 ± 2.96	21.02 ± 5.14	24.97 ± 2.01	26.84 ± 2.13	21.07 ± 1.81
	J_4	9.91 ± 1.16	22.12 ± 3.97	41.17 ± 2.35	22.91 ± 2.89	21.23 ± 1.79
GDA (%)	J	6.81 ± 1.40	9.19 ± 2.26	15.19 ± 2.04	23.91 ± 4.91	21.96 ± 3.69
	J_4	8.64 ± 1.26	8.22 ± 2.22	41.58 ± 2.08	25.69 ± 2.98	21.99 ± 3.68

Table 10

Average Error Rate \pm Standard Deviation with Different σ^2 for KDDA in the Severe SSS Scenario ($\sigma_{Opt}^2 = 1.3 \times 10^5$, $\sigma_{optJ_4}^2 = 4.0 \times 10^7$)

σ^2	σ_{opt}^2	$\sigma_{optJ_4}^2$	10^4	10^5	10^6	5×10^6
ErrRate %	28.09 ± 1.41	29.04 ± 1.38	34.64 ± 1.41	28.16 ± 1.43	28.70 ± 1.40	28.98 ± 1.39
σ^2	10^7	4×10^7	8×10^7	1.2×10^8	1.5×10^8	5×10^8
ErrRate %	29.02 ± 1.37	29.04 ± 1.39	29.04 ± 1.42	29.09 ± 1.41	29.06 ± 1.39	29.09 ± 1.39

Table 11

Average Error Rate \pm Standard Deviation with Different σ^2 for GDA in the Severe SSS Scenario ($\sigma_{Opt}^2 = 1.3 \times 10^5$, $\sigma_{optJ_4}^2 = 4.0 \times 10^7$)

σ^2	σ_{opt}^2	$\sigma_{optJ_4}^2$	10^4	10^5	10^6	5×10^6
ErrRate %	30.61 ± 4.04	31.03 ± 4.66	34.59 ± 1.51	30.84 ± 4.16	32.55 ± 4.56	32.66 ± 4.44
σ^2	10^7	4×10^7	8×10^7	1.2×10^8	1.5×10^8	5×10^8
ErrRate %	32.19 ± 5.42	32.22 ± 4.64	32.07 ± 4.67	31.05 ± 4.36	31.61 ± 4.41	31.94 ± 4.85

Table 12

Average Error Rate \pm Standard Deviation of KDDA and GDA Using the Gaussian Kernel Function with a Single Parameter(J) or Multiple Parameters (J_{Multi}) Optimized by the Proposed Method, Xiong's Method (Xiong) and the Target Alignment Method (TA)

		BreastCancer	Ionosphere	Vowel	Wine
KDDA (%)	J_{Multi}	12.84 \pm 6.99	18.68 \pm 6.28	26.62 \pm 2.33	24.79 \pm 9.06
	J	13.77 \pm 2.96	21.02 \pm 5.14	24.97 \pm 2.01	26.84 \pm 2.13
	Xiong	11.52 \pm 2.28	13.44 \pm 6.32	30.6 \pm 3.02	29.85 \pm 4.18
	TA	10.6 \pm 2.09	22.15 \pm 3.02	-	-
GDA (%)	J_{Multi}	10.41 \pm 7.09	9.21 \pm 1.85	14.39 \pm 1.86	22.02 \pm 8.25
	J	6.81 \pm 1.40	9.19 \pm 2.26	15.19 \pm 2.04	23.91 \pm 4.91
	Xiong	7.61 \pm 1.56	11.13 \pm 3.89	24.57 \pm 3.11	28.62 \pm 6.96
	TA	12.31 \pm 0.8	25.6 \pm 4.18	-	-

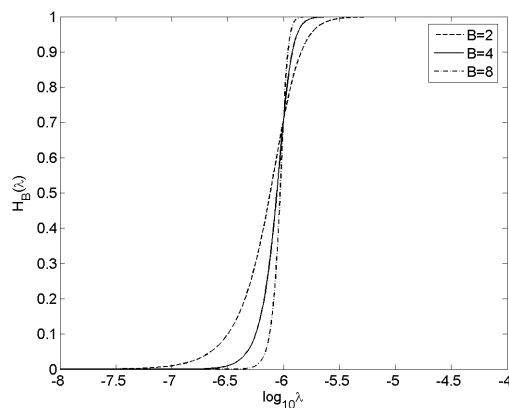
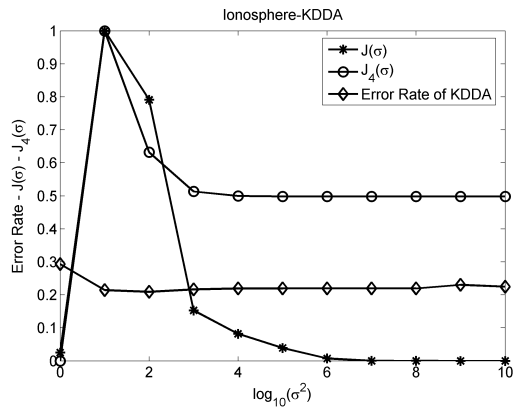
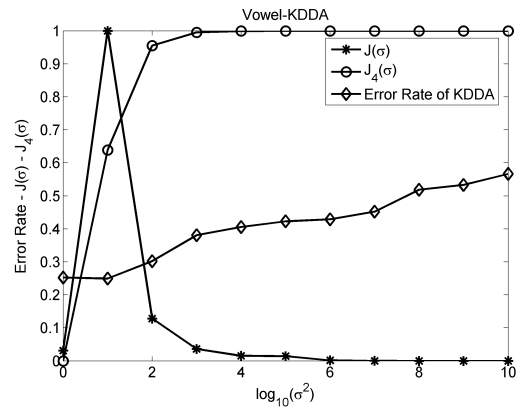


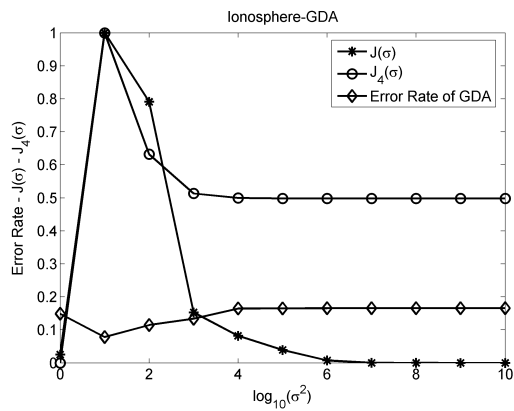
Fig. 1. $H_B(\lambda)$ with $T = 10^{-6}$ and various B s



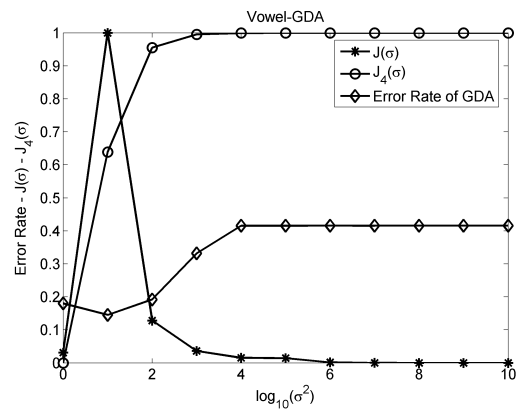
(a)



(b)

Fig. 2. Plots of $J(\sigma)$, $J_4(\sigma)$ and the Average Error Rate of KDDA against σ 

(a)



(b)

Fig. 3. Plots of $J(\sigma)$, $J_4(\sigma)$ and the Average Error Rate of GDA against σ