# A DC-Programming Algorithm for Kernel Selection

**Andreas Argyriou** A.ARGYRIOU@CS.UCL.AC.UK
Department of Computer Science, University College London, Gower Street, London WC1E 6BT, UK

**Raphael Hauser** RAPHAEL.HAUSER@COMLAB.OX.AC.UK
Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK

**Charles A. Micchelli**
Department of Mathematics and Statistics, State University of New York, The University at Albany, 1400 Washington Avenue, Albany, NY, 12222, USA

**Massimiliano Pontil** M.PONTIL@CS.UCL.AC.UK
Department of Computer Science, University College London, Gower Street, London WC1E 6BT, UK

## Abstract

We address the problem of learning a kernel for a given supervised learning task. Our approach consists in searching within the convex hull of a prescribed set of basic kernels for one which minimizes a convex regularization functional. A unique feature of this approach compared to others in the literature is that the number of basic kernels can be infinite. We only require that they are continuously parameterized. For example, the basic kernels could be isotropic Gaussians with variance in a prescribed interval or even Gaussians parameterized by multiple continuous parameters. Our work builds upon a formulation involving a minimax optimization problem and a recently proposed greedy algorithm for learning the kernel. Although this optimization problem is not convex, it belongs to the larger class of DC (difference of convex functions) programs. Therefore, we apply recent results from DC optimization theory to create a new algorithm for learning the kernel. Our experimental results on benchmark data sets show that this algorithm outperforms a previously proposed method.

Appearing in *Proceedings of the 23$^{rd}$ International Conference on Machine Learning*, Pittsburgh, PA, 2006. Copyright 2006 by the author(s)/owner(s).

## 1. Introduction

An essential ingredient in a wide variety of machine learning algorithms is the choice of the kernel. The performance of the algorithm is affected by which kernel is used. Commonly used kernels are Gaussian or polynomial ones. However, there are additional possible classes of kernels one can use. Recent interest has focused on the question of learning the kernel from a prescribed family of available kernels, $\mathcal{K}$, which is often required to be convex. Generally, the method is to specify an objective function of the kernel and to optimize it over $\mathcal{K}$. This task has been pursued from different perspectives, see (Argyriou et al., 2005; Bach et al., 2004; Lanckriet et al., 2004; Lin & Zhang, 2003; Micchelli & Pontil, 2005; Ong et al., 2003; Sonnenburg et al., 2006) and references therein. An essential aspect of our perspective is that we consider the convex hull of a continuous parameterized family. For example, the family of Gaussians whose covariance is an arbitrary positive multiple of the identity matrix, or the family of polynomial kernels of arbitrary degree. This point of view avoids deciding in advance which finite set of variances must be chosen to specify a finite set of kernels whose convex hull is then considered, see (Bach et al., 2004; Lanckriet et al., 2004; Lin & Zhang, 2003).

Almost exclusively, Gaussians with isotropic covariance have been considered up to now, that is, the covariance is a multiple of the identity matrix. An important departure from previous work that we take in this paper is to consider the possibility that the covariance is a full matrix, although perhaps constrained appropriately. This leads us to a challenging optimization problem for choosing the covariance matrix as a

function of the available data. In (Argyriou et al., 2005) an algorithm was proposed for automatically choosing the scalar covariance and its effectiveness was demonstrated on a benchmark dataset. Here we propose a new technique for handling convex combinations of basic kernels which are continuously parameterized by multiple parameters. A key ingredient is the use of a DC (difference of convex functions) algorithm from optimization. Such algorithms have seen recent theoretical development (Horst & Thoai, 1999) and have been used in a variety of application environments, see e.g. (Ellis & Nayakkankuppam, 2003; Neumann et al., 2004). We demonstrate here that even in the scalar covariance case we improve upon previous computational performance in (Argyriou et al., 2005).

The paper is organized as follows. In Section 2 we briefly review the convex optimization perspective for learning the kernel. In Section 3 we discuss a greedy algorithm for this problem which motivates the use of DC-programming in Section 4. Finally, in Section 5 we present numerical simulations using our algorithm.

## 2. Background

In this paper, we focus on learning a functional relationship from examples $\{(x_j, y_j) : j \in \mathbb{N}_m\}$ where the $x_j$ are in a prescribed input space $\mathcal{X}$, the $y_j$ are scalars, $y_j \in \{-1, 1\}$ for binary classification, and we use the notation $\mathbb{N}_m := \{1, \ldots, m\}$. The method we employ chooses the minimum of the regularization functional

$$Q(f) = \sum_{j \in \mathbb{N}_m} q(y_j, f(x_j)) + \mu \|f\|_K^2. \qquad (1)$$

The first term on the right hand side of the above equation is the empirical error measured by a prescribed loss function $q : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ which we assume to be convex in its second argument. The second term is the square of the norm induced by a *reproducing kernel Hilbert space* (RKHS) with kernel $K$, see e.g. (Aronszajn, 1950). The norm is a mechanism to ensure smoothness of the desired function form which we seek. This affects the statistical behavior on new data so that the generalization error is well controlled by the positive parameter $\mu$, thereby preventing overfitting.

The theory of RKHS tells us that the solution we seek has the form

$$f = \sum_{j \in \mathbb{N}_m} c_j K(x_j, \cdot), \qquad (2)$$

where the $c_j$ are real numbers. This result is known as the *Representer Theorem*, see, for example, (Shawe-Taylor & Cristianini, 2004). Consequently, the problem of minimizing the functional $Q$ above reduces to a *finite dimensional* optimization problem. Furthermore, when the loss function is convex, such as the square loss or the hinge loss used in support vector machines, the unique minimizer of functional (1) can be found by replacing $f$ by the right hand side of equation (2) in equation (1) and then optimizing with respect to the parameters $c_j$.

Our presentation here has been brief since these issues are well documented throughout the learning theory literature, see e.g. (Shawe-Taylor & Cristianini, 2004) and references therein. Instead, we wish to address now the issue of *how to choose the kernel* $K$. In this regard, we assume a limited collection of prescribed *basic kernels*

$$\mathcal{G} = \{G(\omega) : \omega \in \Omega\},$$

where $\Omega$ is the kernel parameter space. Each choice of a basic kernel leads to a learned functional form (2). Our challenge is to find some convex combination of the basic kernels which "learns better" than any specific choice of the basic kernels.[1] This presents us with a dilemma, namely if, for example, our basic kernels are Gaussians parameterized by a covariance matrix, how do we choose them? Should we fix them in advance and then use convex combinations as in (Lanckriet et al., 2004) or can we learn them from data? Moreover, should our search for these covariances be limited to a scalar multiple of the identity matrix or can we effectively search through full covariance matrices?

An essential aspect of our work is that we optimize over a continuously parameterized set of basic kernels. That is, we consider kernels in the set

$$\mathcal{K}(\mathcal{G}) = \left\{ \int_\Omega G(\omega) dp(\omega) : p \in \mathcal{P}(\Omega) \right\}, \qquad (3)$$

where $\mathcal{P}(\Omega)$ is the set of all probability measures on $\Omega$ and $G(\cdot)(x, t)$ is continuous for all $x, t \in \mathcal{X}$. For example, when $\Omega = \mathbb{R}_+$ and the function $G(\omega)$ is a multivariate Gaussian kernel with covariance which is a scalar multiple of the identity matrix, namely $G(\omega)(x, t) = e^{-\omega \|x-t\|^2}$, $x, t \in \mathbb{R}^d$, then $\mathcal{K}(\mathcal{G})$ is the set of all radial functions which are kernels for *any* finite choice of the input space dimension $d$ (Schoenberg, 1938).

Returning to the issue of optimal kernel selection, we recall that the minimum regularization error, defined

[1] A distinct problem would be that of learning a single basic kernel from a class of parameterized kernels, see (Chapelle et al., 2002; Rasmussen & Williams, 2005) and references therein.

as

$$E(K) = \min\{Q(f) : f \in \mathcal{H}_K\}, \tag{4}$$

is a convex function of the kernel (Micchelli & Pontil, 2005). This important property motivates us to consider minimizing functional $E(K)$ as a criterion for choosing the kernel $K$. Specifically, we search for a global minimum of $E$ over the convex hull $\mathcal{K}(\mathcal{G})$ of basic kernels $\mathcal{G}$, see equation (3). We note that the point of view of learning an optimal kernel within the convex hull of a finite set of kernels was studied in (Bach et al., 2004; Lanckriet et al., 2004) where semidefinite programming was employed for its practical implementation and independently in (Lin & Zhang, 2003) in the context of smoothing splines.

### 2.1. Overfitting Issue

The fact that we consider an infinite set of basic kernels may lead to concerns that our algorithm could overfit the data. This would be the case if any positive multiple of a basic kernel is also a basic kernel. Indeed, recall the following property of the reproducing kernel norm, namely, for every kernel $K$ and $\lambda > 0$, we have that $\|\cdot\|^2_{\lambda K} = \frac{1}{\lambda}\|\cdot\|^2_K$. Therefore, if $K$ interpolates the data, for example in the case of a Gaussian kernel, the minimum of the functional (4) over $K \in \mathcal{K}(\mathcal{G})$ is zero.

This problem is avoided when the basic kernels satisfy the uniform boundedness condition, namely,

$$\kappa := \max\{G(\omega)(x,x) : \omega \in \Omega, x \in \mathcal{X}\} < \infty. \tag{5}$$

For the class of Gaussian kernels we have that $\kappa = 1$. When condition (5) holds true, it is shown in (Micchelli et al., 2005c) that, for a wide class of loss functions, the minimum of the functional (4) over $\mathcal{K}(\mathcal{G})$ is bounded below and away from zero and generalization error bounds for the function (2) learned with the optimal convex combination of kernels in $\mathcal{G}$ are derived.

We note that the bound (5) implies both a bound on the Frobenius norm and the trace of the kernel matrix $K_{\mathbf{x}}$, conditions imposed in (Lanckriet et al., 2004).

### 2.2. Minimax Problem

We now describe in detail the variational problem of minimizing (4) over $K$. To this end, we recall, for each $y \in \mathbb{R}$, that the conjugate function of $q(y,\cdot)$, denoted by $q^*(y,\cdot) : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$, is defined, for every $v \in \mathbb{R}$, as

$$q^*(y,v) = \sup\{wv - q(y,w) : w \in \mathbb{R}\}$$

and it follows, for every $w \in \mathbb{R}$, that

$$q(y,w) = \sup\{wv - q^*(y,v) : v \in \mathbb{R}\}.$$

Using this notion and a version of the von Neumann minimax theorem, see (Micchelli & Pontil, 2005), we can rewrite the variational problem (4) as

$$E(K) = -\min_{c \in \mathbb{R}^m} \left\{ \frac{1}{4\mu}\langle c, K_{\mathbf{x}} c\rangle + \sum_{i \in \mathbb{N}_m} q^*(y, c_i) \right\}, \tag{6}$$

where $\langle\cdot,\cdot\rangle$ is the standard inner product in $\mathbb{R}^m$ and $K_{\mathbf{x}}$ is the $m \times m$ matrix with elements $K(x_i, x_j)$, $i, j \in \mathbb{N}_m$, which we assume to be invertible. Equation (6) reveals that $E$ is a convex function of $K$.

We remark that, generally, $E$ is not strictly convex. In that case, we may add to it a positive multiple of a strictly convex functional of $K$, for example, the Frobenius norm of the matrix $K_{\mathbf{x}}$.

### 2.3. Examples of Kernel Classes

Let $x \in \mathbb{R}^d$ and $\Sigma$ be a $d \times d$ positive definite matrix. A basic kernel can be built from the Gaussian kernel

$$G(\Sigma)(x,t) = e^{-\langle (x-t), \Sigma^{-1}(x-t)\rangle}. \tag{7}$$

The case $\Sigma = \sigma I$ was discussed above. As before, we consider the question of finding the optimal kernel among the convex hull of the kernels defined above where $\Sigma$ is constrained to lie in a compact set. However, to ensure that $\Sigma$ plays the role of a covariance we must bound its determinant away from zero. A special case is provided by a block diagonal covariance

$$\Sigma = \mathrm{diag}(\Sigma_1, \ldots, \Sigma_\ell),$$

where each $\Sigma_i$ is a $d_i \times d_i$ matrix, $i \in \mathbb{N}_\ell$, and $\sum_{i \in \mathbb{N}_\ell} d_i = d$. We write $x$ as $x = (x^i, i \in \mathbb{N}_\ell)$, where $x^i \in \mathbb{R}^{d_i}$. Thus, we obtain that

$$G(\Sigma)(x,t) = \prod_{i \in \mathbb{N}_\ell} e^{-\langle (x^i - t^i), \Sigma_i^{-1}(x^i - t^i)\rangle}.$$

Next, we present another example for the case of two dimensions. For this purpose, we write $\Sigma^{-1} = U \Lambda U^\top$ where $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2)$, $0 < \lambda_1 \le \lambda_2$ and $U$ is unitary, i.e. $U_{11} = U_{22} = \cos\theta$, $U_{12} = -U_{21} = \sin\theta$.

A direct computation gives that

$$\langle x, \Sigma^{-1} x\rangle = \left( \frac{\lambda_1 + \lambda_2}{2} + \frac{\lambda_2 - \lambda_1}{2}\cos(2\theta - \gamma) \right) \|x\|^2,$$

where $\gamma$ depends only on $x$. Using this formula and integrating (7) over the parameters $\theta$, $\lambda_1, \lambda_2$ with the measure $\frac{1}{4} W_1(\frac{\lambda_1+\lambda_2}{2}) W_2(\frac{\lambda_2-\lambda_1}{2}) d\theta d\lambda_1 d\lambda_2$, we obtain, for $x, t \in \mathbb{R}^2$, that

$$K(x,t) = H_1(\|x-t\|^2) H_2(\|x-t\|^2),$$

where

$$H_1(t) = \int_0^\infty e^{-tu} W_1(u)du, \quad H_2(t) = \pi \int_0^\infty I_0(tv) W_2(v)dv$$

and $I_0(t) = \frac{1}{\pi}\int_{-\pi}^{\pi} e^{-t\cos\theta}d\theta$, the modified Bessel function of order zero.

### 2.4. Feature Space Formulation

An alternate point of view for learning the kernel focuses on the feature space representation of kernels. The idea here is to reformulate the variational problem described in detail above in the space associated with the features of basic kernels. This issue is investigated in generality in (Micchelli & Pontil, 2005b). Here, we wish to highlight some of its main observations. To keep the discussion accessible we restrict ourselves to the case that the parameter set $\Omega$ is finite, that is, $\Omega = \mathbb{N}_n$ and each of the basic kernels is determined by a finite number of features. Hence, for each $\ell \in \mathbb{N}_n$ we write $G_\ell(x,t) = \langle \Phi_\ell(x), \Phi_\ell(t)\rangle$, where $\Phi_\ell(t) \in \mathbb{R}^s$ and $s$ is the number of features. With this representation of the basic kernels, we follow (Micchelli & Pontil, 2005b) and consider the variational problem

$$\sum_{j\in\mathbb{N}_m} q\left(y_j, \sum_{\ell\in\mathbb{N}_n} \langle w_\ell, \Phi_\ell(x_j)\rangle\right) + \mu \sum_{\ell\in\mathbb{N}_n} \|w_\ell\|^2. \quad (8)$$

The minimizer $(\hat{w}_\ell : \ell \in \mathbb{N}_n)$ of this variational problem provides an optimal kernel based on the features defined above. In particular, the optimal kernel is given by

$$\hat{K} = \sum_{\ell\in\mathbb{N}_n} \frac{\|w_\ell\|}{\sum_{r\in\mathbb{N}_n}\|w_r\|} G_\ell.$$

A detailed explanation for this fact and its extensions to the continuous case can be found in (Micchelli & Pontil, 2005b, Thm. 2.1). As pointed out in that paper, when there is one feature, namely $s = 1$, equation (8) reduces to the $\ell^1$-regularization problem. There has been renewed interest in this problem because a minimizing solution will often have few nonzero coefficients, in other words will be sparse.

## 3. Learning Algorithm

Let us now turn to the theme which is central in the paper, namely the problem of minimizing the functional $E$ in (6) over the convex hull $\mathcal{K}(\mathcal{G})$ of basic kernels $\mathcal{G}$. This problem is equivalent to a *saddle point problem*, which we rewrite as

$$\max\left\{\min\left\{R(c,K) : c \in \mathbb{R}^m\right\} : K \in \mathcal{K}(\mathcal{G})\right\}, \quad (9)$$

where we introduced the function

$$R(c,K) = \frac{1}{4\mu}\langle c, K_{\mathbf{x}} c\rangle + \sum_{i\in\mathbb{N}_m} q^*(y, c_i).$$

**Algorithm 1** Computing an optimal convex combination of basic kernels in $\mathcal{G} = \{G(\omega) : \omega \in \Omega\}$.

**Notation:** $g(K) = \min_{c\in\mathbb{R}^m} R(c,K)$

**Initialization:** Choose $K^{(1)} \in \mathcal{K}(\mathcal{G})$

**for** $t = 1$ to $T$: **do**

**1.** Compute $c^{(t)} = \operatorname{argmin}\{R(c, K^{(t)}) : c \in \mathbb{R}^m\}$

**2.** Find $\hat{\omega} = \operatorname{argmax}\{R(c^{(t)}, G(\omega)) : \omega \in \Omega\}$.

**if** $R(c^{(t)}, G(\hat{\omega})) > R(c^{(t)}, K^{(t)})$ **then**

go to Step 3

**else**

terminate

**3.** Compute $K^{(t+1)} = \operatorname{argmax}\{g(K) : K \in co(G(\hat{\omega}), K^{(t)})\}$

The analysis in (Argyriou et al., 2005) provides necessary and sufficient conditions for a pair $(\hat{c}, \hat{K}) \in \mathbb{R}^m \times \mathcal{K}(\mathcal{G})$ to be a saddle point. In particular, they show that there always exists an optimal kernel $\hat{K}$ with a *finite representation*, that is,

$$\hat{K} = \int_\Omega G(\omega) d\hat{p}(\omega) = \sum_{i\in\mathbb{N}_{m+1}} \hat{\lambda}_i G(\hat{\omega}_i),$$

where $\hat{\lambda}_i \geq 0$, $\sum_i \hat{\lambda}_i = 1$ and $\hat{\omega}_i \in \Omega$. This result implies, in the case of isotropic Gaussians, that the optimal kernel is a *finite* mixture of at most $m+1$ Gaussians.

These results motivate a greedy algorithm (Algorithm 1) for solving problem (9). The algorithm starts with an initial kernel $K^{(1)} \in \mathcal{K}(\mathcal{G})$ and from it computes the unique vector $c^{(1)} = \operatorname{argmin}\{R(c, K^{(1)}) : c \in \mathbb{R}^m\}$. A sequence of kernels $\{K^{(t)} : t \in \mathbb{N}_T\} \subseteq \mathcal{K}(\mathcal{G})$ is computed iteratively. After $K^{(t)}$ has been chosen, the unique vector $c^{(t)}$ is computed (see Step 1) and, subsequently, the algorithm searches for a global maximizer $\hat{\omega} \in \Omega$ of the function $R(c^{(t)}, G(\omega))$, or, equivalently, of the function $\langle c^{(t)}, G_{\mathbf{x}}(\omega) c^{(t)}\rangle$ (see Step 2). If the inequality

$$\langle c^{(t)}, G_{\mathbf{x}}(\hat{\omega}) c^{(t)}\rangle > \langle c^{(t)}, K^{(t)}_{\mathbf{x}} c^{(t)}\rangle \quad (10)$$

holds, a new kernel $K^{(t+1)}$ is computed to be the optimal convex combination of the kernels $G(\hat{\omega})$ and $K^{(t)}$ (see Step 3). If no $\hat{\omega}$ satisfying inequality (10) can be found, the algorithm terminates. We note that after each iteration the objective function strictly increases or the algorithm terminates, see (Argyriou et al., 2005) for a proof.

Step 1 of the algorithm is a convex optimization problem. In our experiments below, for the square loss $q(y,v) = (y-v)^2$, the vector $c^{(t)}$ is simply determined by solving a linear system of equations. Step 2 is implemented by DC programming which we describe in

the next section. For Step 3, we use a Newton method on the function $g(\lambda G(\hat{\omega}) + (1-\lambda)K^{(t)})$, $\lambda \in [0,1]$, which is concave and whose derivative can be easily computed.

### 3.1. Minimizing Sums of Exponentials

As we pointed out above, a key step in our algorithm is to maximize the objective function

$$f(\omega) = \langle c, G_{\mathbf{x}}(\omega)c\rangle \tag{11}$$

over $\omega \in \Omega$. Here we present some insights into this problem in the case that the basic kernels are exponential functions, namely $G(\omega)(x_i, x_j) = e^{-\omega d(x_i,x_j)}$ and $\Omega \subseteq \mathbb{R}_+$. Recall that $G(\omega)$ is a kernel for all $\omega \in \mathbb{R}_+$ if and only if the matrix $D = (d(x_i, x_j) : i, j \in \mathbb{N}_m)$ is conditionally negative definite, that is, $\langle c, Dc\rangle \leq 0$ whenever $\sum_{i\in\mathbb{N}_m} c_i = 0$. For example, in the Gaussian case, $d(x_i, x_j) = \|x_i - x_j\|^2$ has this property, see e.g. (Schölkopf & Smola, 2002). We wish to bound the number of local extrema on $\mathbb{R}_+$ of the function

$$f(\omega) = \sum_{i\in\mathbb{N}_m} c_i^2 + 2\sum_{i<j} c_i c_j e^{-\omega d(x_i,x_j)}.$$

For this purpose, define the univariate function $g(x) = \sum_{i\in\mathbb{N}_n} a_i e^{\lambda_i x}$, $x \in \mathbb{R}$, where $a_i \in \mathbb{R}, \lambda_1 < \cdots < \lambda_n$. We recall that Laguerre's rule of signs states that the number of nonnegative zeros of $g$ (counting multiplicities) does not exceed the number of sign changes in the sequence $a_1, \cdots, a_n$, which is at most $n-1$. Moreover, this result is sharp, see for example (Steinig, 1986).

In our simulations below we have only noticed between two to five local maxima of $f$. This fact is confirmed by Laguerre's rule of signs. Indeed, in our simulations the inputs $x_i$ clustered well in two groups, that is, $d(x_i, x_j)$ is small when $y_i = y_j$ and larger when $y_i \neq y_j$. Moreover, each $c_i$ usually has the same sign as $y_i$ [2]. Hence, when we order the $d(x_i, x_j)$ in a non-decreasing fashion the corresponding ordering of the coordinates of the vector $(c_i c_j : i, j \in \mathbb{N}_m)$ has only a few sign changes.

**Algorithm 2** Cutting plane algorithm.

**Inputs:** A point $y^0$ in the interior of $\Omega$; a simplex $S^0 \supseteq \Omega$ with vertex set $V(S^0)$; a convex function $\alpha : \mathbb{R}^D \to \mathbb{R}$ such that $\Omega = \{\omega \in \mathbb{R}^D : \alpha(\omega) \leq 0\}$.

**Initialization:** Set $\varphi^0 = g(y^0) - h(y^0)$.
Compute a subgradient $s \in \partial g(y^0)$.
Choose $\bar{t} > \max\{g(\omega) : \omega \in V(S^0)\} - \bar{\varphi}$, where

$$\bar{\varphi} = \min\{g(\omega) : \omega \in V(S^0)\} - \max\{h(\omega) : \omega \in V(S^0)\}.$$

Construct a polytope $P^0$ from $S^0, \bar{t}$ and $y^0$.
Set $k = 0$.
**Iteration** $k$**:** Compute an optimal solution $(\omega^k, t^k)$ of the problem $\min\{-h(\omega) + t : (\omega, t) \in V(P^k)\}$.
**if** $-h(\omega^k) + t^k = 0$ **then**
  Stop. $y^k$ is an optimal solution to (13) with optimal value $\varphi^k$.
**else** $\{-h(\omega^k) + t^k < 0\}$
  Case 1: $\omega^k \in \Omega$
  Compute $s^k \in \partial g(\omega^k)$.
  Case 1a: $g(\omega^k) - h(\omega^k) < \varphi^k$
    Set $y^{k+1} = \omega^k$.
  Case 1b: $g(\omega^k) - h(\omega^k) \geq \varphi^k$
    Set $y^{k+1} = y^k$.
  Case 2: $\omega^k \notin \Omega$
  Compute $s^k \in \partial\beta^k(\omega^k, t^k)$, where $\beta^k(\omega, t) := \max\{\alpha(\omega), g(\omega) - t - \varphi^k\}$.
  Compute the zero $(\zeta^k, \theta^k)$ of $\beta^k(x, t)$ on the line segment joining $(\omega^k, t^k)$ and $(y^0, \bar{t})$.
  Case 2a: $g(\zeta^k) - h(\zeta^k) < \varphi^k$
    Set $y^{k+1} = \zeta^k$.
  Case 2b: $g(\zeta^k) - h(\zeta^k) \geq \varphi^k$
    Set $y^{k+1} = y^k$.
  Construct the cutting plane (affine function)

$$\ell^k(x,t) = \begin{cases} \langle \omega - \omega^k, s^k\rangle + g(\omega^k) - \varphi^{k+1} - t, \\ \text{if } \omega^k \in \Omega \\ \langle (\omega, t) - (\zeta^k, \theta^k), s^k\rangle + \beta^k(\zeta^k, \theta^k), \\ \text{if } \omega^k \notin \Omega. \end{cases}$$

  Set $P^{k+1} = P^k \cap \{(\omega, t) : \ell^k(\omega, t) \leq 0\}$.
  Set $k = k + 1$ and continue to next iteration.

## 4. DC Programming

As we noted above, the objective function $R(c, K)$ of the minimax problem (9) is convex in $c$ and linear in $K$. Unfortunately, in general, $R(c, G(\omega))$ is *not* convex in $\omega$. This makes Step 2 of Algorithm 1 a challenging task. However, as we shall see, for a wide class of basic kernels, the function $R(c, G(\cdot))$ belongs to the class of *DC functions* for which there are available well developed iterative algorithms. We first review a few necessary definitions and results from the theory of DC functions, as presented in (Horst & Thoai, 1999).

Let $\Omega$ be a closed convex subset of $\mathbb{R}^D$. A function $f : \Omega \to \mathbb{R}$ is called DC on $\Omega$ if there exist two *convex* functions $g$ and $h$ such that

$$f(\omega) = g(\omega) - h(\omega), \quad \omega \in \Omega.$$

A remarkable result by (Hartman, 1959) states that

[2] This would always be the case for support vector machines, see e.g. (Shawe-Taylor & Cristianini, 2004).

*locally DC functions* are DC. It also implies that every twice continuously differentiable function on $\Omega$ is DC and every continuous function on $\Omega$ is the limit of a sequence of DC functions that converges uniformly on $\Omega$. Moreover, the class of DC functions is linear and closed under multiplication and the *finite* min/max operations. Optimization problems of the type

$$\inf\{f(\omega) : \omega \in \Omega, f_i(\omega) \leq 0,\ i \in \mathbb{N}_n\}, \qquad (12)$$

where $f$ and $f_i$, $i \in \mathbb{N}_n$, are DC, are called *DC programs*.

We now derive a DC-programming formulation for the problem of maximizing function (11). To this end, we note that, for every $c \in \mathbb{R}^m$, the function $R(c, G(\cdot))$ is the limit of DC functions since, for every $x, t \in \mathcal{X}$, we have assumed continuity of $G(\cdot)(x,t)$. In addition, if $G(\cdot)(x,t)$ is twice continuously differentiable, maximizing (11) is a DC program.

Therefore, for most interesting continuous parameterizations, $G_{\mathbf{x}}(\omega)$ and hence $f(\omega)$ in (11) are DC functions. If, furthermore, the DC decomposition of $G(\cdot)(x,t)$ is available, we obtain an optimization problem of the form

$$\hat{\varphi} = \min\{-f(\omega) = g(\omega) - h(\omega) :\ \omega \in \Omega\}, \qquad (13)$$

where $g, h$ are convex.

In particular, in the case of Gaussian kernels as in (7), $D = d(d+1)/2$, $\omega$ consists of the lower triangular elements of $\Sigma^{-1}$ and the DC decomposition is given by

$$f(\omega) = \sum_{\{i,j:c_i c_j > 0\}} c_i c_j e^{-\langle b_{ij}, \omega\rangle} + \sum_{\{i,j:c_i c_j < 0\}} c_i c_j e^{-\langle b_{ij}, \omega\rangle},$$

where the indices $i, j \in \mathbb{N}_m$ and, for every $i, j$, $b_{ij} := ((2 - \delta(k,\ell))(x_{ik} - x_{jk})(x_{i\ell} - x_{j\ell}) : d \geq k \geq \ell \geq 1)$. Here, $g$ is the first term in the right hand side of the above equation and $h$ is minus the second term.

A necessary and sufficient condition for $\hat{\omega}$ to solve problem (13) is that

$$\min\{-h(\omega) + t : \omega \in \Omega, t \in \mathbb{R}, g(\omega) - t \leq g(\hat{\omega}) - h(\hat{\omega})\}$$

equals zero, see, for example, (Horst & Thoai, 1999, Proposition 4.4). This observation motivated a cutting plane algorithm, a variant of which we have implemented. The details appear in Algorithm 2. The algorithm works by constructing outer polytopes $P^{k+1} \subseteq P^k$ which contain the optimal solution $(\hat{\omega}, \hat{t} = h(\hat{\omega}))$. Subsequent polytopes are defined by cutting out the current vertex $(\omega^k, t^k)$ while keeping the solution inside, namely

$$\ell^k(\omega^k, t^k) > 0,$$
$$\ell^k(\omega, t) \leq 0,\ \text{for } (\omega, t) \in \mathbb{R}^{D+1} : \beta^{k+1}(\omega, t) \leq 0.$$

One can show that the sequence of function values converges to the optimal one from above, that is, $\hat{\varphi} \leq \varphi^{k+1} \leq \varphi^k$, see (Horst & Thoai, 1999) for a proof and a more detailed explanation of the algorithm.

## 5. Experiments

We performed a series of experiments on the MNIST data set of images of handwritten digits (available at `http://yann.lecun.com/exdb/mnist/index.html`). The data are 28×28 images with pixel values ranging between 0 and 255. We compared the output of the DC algorithm with the results in (Argyriou et al., 2005), which were obtained using Gaussian kernels with one variance parameter $\sigma$ in a prescribed interval $[\sigma_\ell, \sigma_u]$. In that paper, a branch-and-bound technique combined with local descent was used for just improving (not fully optimizing) at Step 2 of Algorithm 1.

Using the same set-up, we selected the first 500 training points and 1000 test points corresponding to each task from MNIST. The initial kernel for the greedy algorithm was the average of 5 Gaussian kernels with $\sigma$'s equally spaced in $[\sigma_\ell, \sigma_u]$. We have used the square loss and set the regularization parameter to $10^{-7}$. The maximum number of iterations was set to 100 and 300 for the greedy and DC algorithms, respectively.[3]

In Table 1, we present the results obtained using the DC algorithm alongside the previous greedy algorithm, an algorithm for learning the optimal convex combination of a finite number of basic kernels and SVM-light, see (Argyriou et al., 2005) for more information. The range of $\sigma$ is $[75, 25000]$ in columns 2–5, $[100, 10000]$ in columns 6–9 and $[500, 5000]$ in columns 10–13. It is clear from this table that the DC approach is equally good for selecting a single variance.

Next, we performed experiments with multiple parameters. In such cases, the standard branch-and-bound methods suffer from the curse of dimensionality. DC approaches, however, enable us to tackle optimization problems in several dimensions. In Table 2, we show the performance of the DC algorithm simultaneously learning two isotropic variances, one corresponding to the left and one to the right part of the image. The performance with two variances significantly improves and the algorithm remains robust over different ranges of $\sigma_1$ and $\sigma_2$, which is evidence that the optimization subproblem was successfully solved. In the same table, we present the performance of the finite kernels method mentioned above using grids of $5 \times 5$ and

[3] The code is available at `http://www.cs.ucl.ac.uk/staff/a.argyriou/code/dc`

*Table 1.* Misclassification error percentage for learning one kernel parameter on the MNIST tasks.

| Task | Method | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | DC | standard | finite | SVM | DC | standard | finite | SVM | DC | standard | finite | SVM |
| | $\sigma \in [75, 25000]$ | | | | $\sigma \in [100, 10000]$ | | | | $\sigma \in [500, 5000]$ | | | |
| odd vs. even | 6.5 | 6.6 | 18.0 | 11.8 | 6.5 | 6.6 | 10.9 | 8.6 | 6.5 | 6.5 | 6.7 | 6.9 |
| 3 vs. 8 | 3.7 | 3.8 | 6.9 | 6.0 | 3.9 | 3.8 | 4.9 | 5.1 | 3.6 | 3.8 | 3.7 | 3.8 |
| 4 vs. 7 | 2.7 | 2.5 | 4.2 | 2.8 | 2.4 | 2.5 | 2.7 | 2.6 | 2.3 | 2.5 | 2.6 | 2.3 |

*Table 2.* Misclassification error percentage of DC algorithm vs. finite grid for 2 parameters on the MNIST tasks.

| Task | Number of parameters | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | DC | $5 \times 5$ | $10 \times 10$ | DC | $5 \times 5$ | $10 \times 10$ | DC | $5 \times 5$ | $10 \times 10$ |
| | $\sigma \in [75, 25000]$ | | | $\sigma \in [100, 10000]$ | | | $\sigma \in [500, 5000]$ | | |
| odd vs. even | 5.8 | 15.8 | 11.2 | 5.8 | 10.1 | 6.2 | 5.8 | 6.8 | 5.8 |
| 3 vs. 8 | 2.7 | 6.5 | 5.1 | 2.5 | 4.6 | 2.5 | 2.6 | 3.5 | 2.5 |
| 4 vs. 7 | 1.8 | 3.9 | 2.9 | 1.7 | 2.7 | 2.0 | 1.8 | 2.0 | 1.8 |

$10 \times 10$ kernels with equally spaced $\sigma$'s. This method succeeds only in the smallest range of $\sigma$'s and with the $10 \times 10$ grid. But in the absense of information about $\sigma$, even with the finer grid the finite kernels method is not competitive. We also performed experiments with four isotropic variances (corresponding to the four quadrants of the image), which did not further improve the results.

As regards the computational cost, our method compares favorably to the finite method. We implemented both of them in Matlab and performed the experiments on a 1GHz dual-processor machine running Linux. For the local optimization of $-h(x) + t$ in Algorithm 2, we used Matlab's `fmincon()` routine. We observed that the finite method is much worse in time cost than the DC algorithm (about 1 hour vs. 5 minutes with 2 parameters). The main computational cost of our algorithm is incurred by the aforementioned local optimization, whereas the finite method scales polynomially with the grid size. Still, the running time of our algorithm deteriorates fast with the number of parameters. For example, it takes 1-2 minutes for learning one parameter, about 5 minutes for 2 parameters and about 1 hour for 4 parameters. We speculate that faster local search exploiting linear programming and the special nature of the function can lead to further improvements in efficiency. With respect to memory requirements, our algorithm is clearly more efficient because it only needs to store the linear constraints, whereas the grid method requires all the kernels to be in memory.

We also observed that the greedy algorithm usually required less than 20 iterations to terminate, which is evidence in favor of the DC approach. The cutting plane method usually required less than 100 iterations to converge. Thus, the learned convex combination has a small (usually less than 10) number of kernels.

Finally, in Figure 1 we present the learned kernel coefficients for two isotropic variances (left and right image) in the range $[100, 10000]$. These indicate that for the odd vs. even task it is better to combine several complementary kernels focused on different parts of the images than use a single Gaussian kernel. However, for the 3-8, 4-7 tasks there is a clear winner among the kernels. This conforms with the intuition that odd vs. even is a more complex task than the binary ones. Moreover, augmenting the parameter class for odd vs. even results in a more complex (and more effective) representation for the solution. In order to gain insight into the nature of this solution, we have plotted the corresponding variances for odd vs. even in Figure 2. It is clear that the learned kernels are either focused *exclusively* on each half of the images or operate on the image as a whole.

## 6. Summary

We have studied the problem of selecting a kernel in the convex hull of a prescribed set of continuously parameterized basic kernels. A substantial innovation of our approach is that it avoids deciding in advance a finite subset of basic kernels. We have reviewed recent work in this direction and discussed a greedy algorithm for this problem. The algorithm relies on a DC-programming formulation, a recently developed technique for global optimization. We have provided experimental results where the basic kernels are Gaussians with constrained diagonal covariance matrices. These computational results indicate the advantage of working with a continuous parameterization as opposed to an *a priori* choice of a finite number of basic kernels. The method is robust because the choice of the optimal kernel is insensitive to the range of the continuous parameters.

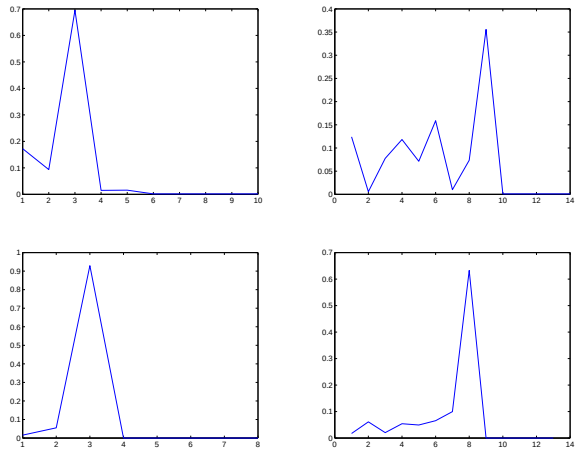

*Figure 1.* Learned kernel coefficients for different classification tasks and kernel parameterizations. Top plots are for odd vs. even (the dimensionality is 1 on the left and 2 on the right). Bottom plots are for the 3-8 task (left) and the 4-7 task (right), with dimensionality 2.

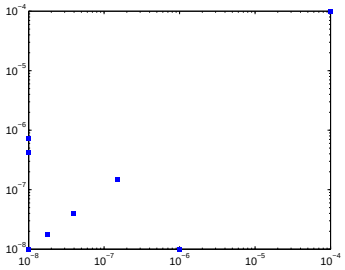

*Figure 2.* Learned $[\sigma_1, \sigma_2]$ parameters of the kernels in the optimal convex combination, for the odd vs. even task. The parameter range was [100,10000].

## References


Argyriou, A., Micchelli, C. A., & Pontil, M. (2005). Learning convex combinations of continuously parameterized basic kernels. *Proc. of the 18th Conference on Learning Theory*, pages 338–352.

Aronszajn, N. (1950). Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, *686*, 337–404.

Bach, F. R., Lanckriet, G. R. G., & Jordan, M. I. (2004). Multiple kernel learning, conic duality, and the SMO algorithm. *Proc. of the 21st International Conference on Machine Learning.*

Chapelle, O., Vapnik, V., Bousquet, O., & Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. Machine Learning, Springer, 2002.

Ellis, S., & Nayakkankuppam, M. (2003). *Phylogenetic analysis via DC programming* (preprint). Department of Mathematics and Statistics, UMBC, Baltimore, MD.

Hartman, P. (1959). On functions representable as a difference of convex functions. *Pacific Journal of Mathematics*, 9: 707–713.

Horst, R., & Thoai, N. V. (1999). DC programming: overview. *Journal of Optimization Theory and Applications*, *103*, 1–41.

Lanckriet, G. R. G., Cristianini, N., Bartlett, P., El-Ghaoui, L., & Jordan, M. I. (2004). Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, *5*, 27–72.

Lin, Y., & Zhang, H. H. (2003). *Component selection and smoothing in smoothing spline analysis of variance models – cosso* (Technical Report 2556). Institute of Statistics Mimeo Series, NCSU.

Micchelli, C. A., & Pontil, M. (2005). Learning the kernel function via regularization. *J. of Machine Learning Research*, 6: 1099–1125, 2005.

Micchelli, C. A., & Pontil, M. (2005b). Feature space perspectives for learning the kernel. To appear in *Machine Learning* (see also: Technical Report RN/05/11, Dept. of Computer Science, UCL).

Micchelli, C. A., Pontil, M., Wu, Q., & Zhou, D-X. (2005c). Error bounds for learning the kernel. Research Note RN/05/09, Dept of Computer Science, UCL, 2005.

Neumann, J., Schnörr, C., Steidl, G. (2004). SVM-based feature selction by direct objective minimisation. C.E. Rasmussen et al (eds.), Proc. of 26th DAGM Symposium.

Ong, C. S., Smola, A. J. & Williamson. R. C. (2003). Hyperkernels. *Advances in Neural Information Processing Systems*, 15, S. Becker et. al (Eds.), MIT Press, Cambridge, MA.

Rasmussen, C. E. & Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, Cambridge, MA.

Schoenberg, I. J. (1938). Metric spaces and completely monotone functions. *Annals of Mathematics*, *39*, 811–841.

Schölkopf, B. & Smola, A. J. (2002). *Learning with Kernels*. The MIT Press, Cambridge, MA.

Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Sonnenburg, S., Raetsch, G. & Schaefer, C. (2006). A General and efficient multiple kernel learning algorithm In *Advances in Neural Information Processing Systems*, 19.

Steinig, J. (1986). A rule of signs for real exponential polynomials. *Acta Math. Hungar.* 47 (1), 187–190.