Journal of Machine Learning Research 21 (2020) 1-29 Submitted 7/19; Revised 10/19; Published 03/20

# A Convex Parametrization of a New Class of Universal Kernel Functions

**Brendon K. Colbert** BRENDON.COLBERT@ASU.EDU
*Department of Mechanical and Aerospace Engineering*
*Arizona State University*
*Tempe, AZ 85281-4322, USA*

**Matthew M. Peet** MPEET@ASU.EDU
*Department of Mechanical and Aerospace Engineering*
*Arizona State University*
*Tempe, AZ 85281-1776, USA*

**Editor:** Mehryar Mohri

## Abstract

The accuracy and complexity of kernel learning algorithms is determined by the set of kernels over which it is able to optimize. An ideal set of kernels should: admit a linear parameterization (tractability); be dense in the set of all kernels (accuracy); and every member should be universal so that the hypothesis space is infinite-dimensional (scalability). Currently, there is no class of kernel that meets all three criteria - e.g. Gaussians are not tractable or accurate; polynomials are not scalable. We propose a new class that meet all three criteria - the Tessellated Kernel (TK) class. Specifically, the TK class: admits a linear parameterization using positive matrices; is dense in all kernels; and every element in the class is universal. This implies that the use of TK kernels for learning the kernel can obviate the need for selecting candidate kernels in algorithms such as SimpleMKL and parameters such as the bandwidth. Numerical testing on soft margin Support Vector Machine (SVM) problems show that algorithms using TK kernels outperform other kernel learning algorithms and neural networks. Furthermore, our results show that when the ratio of the number of training data to features is high, the improvement of TK over MKL increases significantly.

**Keywords:** kernel functions, multiple kernel learning, semi-definite programming, supervised learning, universal kernels

## 1. Introduction

This paper addresses the problem of the automated selection of an optimal kernel function for a given kernel-based machine learning problem (e.g. soft margin SVM). Kernel functions implicitly define a linear parametrization of nonlinear candidate maps $y = f(x)$ from vectors $x$ to scalars $y$. Specifically, for a given kernel, the 'kernel trick' allows optimization over a set of candidate functions in the kernel-associated hypothesis space without explicit representation of the space itself. The kernel selection process, then, is critical for determining the class of hypothesis functions and, as a result, is a well-studied topic with common kernels including polynomials, Gaussians, and many variations of the Radial Basis Function. In addition, specialized kernels include string kernels as in Lodhi et al. (2002); Eskin et al. (2003), graph kernels as in Gärtner et al. (2003), and convolution kernels as in Haussler (1999); Collins and Duffy (2002). The kernel selection process heavily influences the accuracy of the resulting fit and hence significant research has gone into the optimization of these kernel functions in order to select the hypothesis space which most accurately represents the underlying physical process.

Recently, there have been a number of proposed kernel learning algorithms. For support vector machines, the methods proposed in this paper are heavily influenced by the SDP approach proposed by Lanckriet et al. (2004) which directly imposed kernel matrix positivity on a subspace defined by the linear combination of candidate kernel functions. There have been several extensions of the SDP approach, including the hyperkernel method of Ong et al. (2005). However, because of the complexity of semidefinite programming, more recent work has focused on alignment methods for MKL as in, e.g. Cortes et al. (2012) or gradient methods for convex and non-convex parameterizations of positive linear combinations of candidate kernels, such as SimpleMKL in Rakotomamonjy et al. (2008) or the several variations in Sonnenburg et al. (2010). These MKL methods rely on kernel operations (addition, multiplication, convolution) to generate large numbers of parameterized kernel functions as in Cortes et al. (2009). Examples of non-convex parameterizations include GMKL as introduced in Jain et al. (2012), and LMKL as introduced in Gönen and Alpaydin (2008). Work focused on regularization includes the group sparsity metric defined in Subrahmanya and Shin (2010) and the enclosing ball approach in Gai et al. (2010). See, e.g. Gönen and Alpaydın (2011) for a comprehensive review of MKL algorithms.

In this paper, we focus on the class of "Universal Kernels" formalized in Micchelli et al. (2006). For a given compact metric space (input space), $\mathcal{X}$, it is said that a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a Positive Kernel (PK) if for any $N \in \mathbb{N}$ and any $\{x_i\}_{i=1}^N \subset \mathcal{X}$, the matrix defined elementwise by $K_{ij} = k(x_i, x_j)$ is symmetric and Positive SemiDefinite (PSD).

**Definition 1** *A kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be universal on the compact metric space $\mathcal{X}$ if it is continuous and there exists an inner-product space $\mathcal{W}$ and feature map, $\Phi : \mathcal{X} \to \mathcal{W}$ such that $k(x,y) = \langle \Phi(x), \Phi(y) \rangle_{\mathcal{W}}$ and where the unique Reproducing Kernel Hilbert Space (RKHS),*

$$\mathcal{H} := \{f \;:\; f(x) = \langle v, \Phi(x) \rangle,\; v \in \mathcal{W}\}$$

*with associated norm $\|f\|_{\mathcal{H}} := \inf_v \{\|v\|_{\mathcal{W}} \;:\; f(x) = \langle v, \Phi(x) \rangle\}$ is dense in $\mathcal{C}(\mathcal{X}) := \{f : \mathcal{X} \to \mathbb{R} : f \text{ is continuous}\}$ where $\|f\|_{\mathcal{C}} := \sup_{x \in X} |f(x)|$.*

Note that for any given PD kernel, $\mathcal{H}$ exists, is unique, and can be characterized (as described in Sun (2005)) using the Riesz representation theorem as the closure of $\text{span}\{k(y,\cdot) \;:\; y \in \mathcal{X}\}$ with inner product defined for any $f(x) = \sum_{i=1}^n c_i k(y_i, x)$ and $g(x) = \sum_{i=1}^m d_i k(z_i, x)$ as

$$\langle f, g \rangle_{\mathcal{H}} := \sum_{i=1}^n \sum_{j=1}^m c_i d_j k(y_i, z_j).$$

Universal kernels are preferred when large amounts of data are available, due to the fact that the dimension of the hypothesis space increases for every additional data point - resulting in the ability to construct highly specialized and accurate classifiers. The most well-known example of a universal kernel is the Gaussian (generalized in Zanaty and Afifi

(2011)). However, many other common kernels are not universal, including, significantly, the polynomial class of kernels. This is significant because generalized polynomial kernels (Eq. (9)) are dense in all kernels and admit a linear parameterization using a monomial basis, while Gaussian kernels (which are universal, but not dense in all kernels) do not.

**The Class of Tessellated Kernels (TK)** In this paper we propose a new class of kernel functions (called Tessellated Kernels) which are not polynomials, yet which are defined by polynomials and admit a linear parametrization. These kernels define classifiers on a tessellated domain, each sub-domain (or tile) of which is a hyper-rectangle with vertices defined by the *input data* - $\{x_i\}_{i=1}^m$. In this way, each data point further divides any tiles within which any of its features lie, resulting in increasing numbers of disjoint tiles. The classifier itself, then, is piecewise polynomial - being polynomial when restricted to any particular tile.

TK kernels have three important properties which make them uniquely well-suited for kernel learning problems. First, these kernels admit a linear parameterization using positive semidefinite matrices - meaning we can use convex optimization to search over the entire class of such kernels (tractability), which is proven in Corollary 13 and implemented in Optimization Problem (24). This is like the class of generalized polynomial kernels (See Eq. (9)) yet unlike other universal kernel classes such as the Gaussian/RBF, wherein the bandwidth parameter appears in the exponential. Second, the TK class is dense in all kernels (accuracy), meaning there exists a TK kernel that can approximate any given kernel arbitrarily well. This is like the generalized polynomial class yet unlike the Gaussian/RBF class, wherein the resulting kernel matrix is restricted to having all positive elements. Third, any kernel of the TK class has the universal property (scalability). This is like the Gaussian/RBF class and unlike the generalized polynomial kernels, none of which are universal. The TK class is thus unique in that no other currently known class of kernel functions has all three properties of tractability, accuracy, and scalability. Finally, we demonstrated through extensive numerical testing that kernel learning using TK kernels significantly outperforms other kernel learning algorithms in terms of accuracy.

The paper is organized as follows. In Section 2 we provide an overview of the MKL problem. Section 3 proposes a framework by which the class of TK kernels can be parameterized by positive matrices. Section 4 proves general properties such as universality for every member of the class of TK kernels. Sections 5 and 6 show how the class of TK kernels can be rigorously incorporated into the SDP MKL framework and into SimpleMKL's framework respectively. In Section 7 we discuss the complexity of incorporating TK kernels into both the SDP MKL framework and the SimpleMKL framework. Finally in Section 8 we provide numerical results that illustrate improved performance using TK kernels on a number of UCI repository data sets.

## 2. Formulation of the Kernel Learning Problem

We begin this section by posing the kernel-learning problem as a convex optimization problem for the particular case of the 1-norm soft margin support vector machine. Next, in Subsections A and B, we present two standard algorithms for solving the kernel learning problem. These algorithms are general in the sense that they apply to any given linear parameterization of kernel functions. The adaptation of these algorithms to the special case of TK kernels will then be described in Section 3.

Suppose we are given a set of $m$ *training data* points $\{x_i\}_{i=1}^m \subset \mathbb{R}^n$, each with associated *label* $y_i \in \{-1, 1\}$ for $i = 1, \cdots, m$. For a given "penalty" parameter $C \in \mathbb{R}^+$, we define the primal version of the *linear* 1-norm soft margin problem as

$$\begin{aligned} \min_{w\in\mathbb{R}^n,\ \zeta\in\mathbb{R}^m,\ b\in\mathbb{R}} \quad & \frac{1}{2}w^T w + C\sum_{i=1}^m \zeta_i \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0, \end{aligned} \tag{1}$$

where the learned map (*classifier*) from inputs to outputs is then $f : \mathbb{R}^n \to \{-1, 1\}$ where

$$f(x) = \text{sign}(w^T x + b).$$

If we desire the classifier to be a nonlinear function, we may introduce a positive kernel function, $k$.

**Definition 2** *We say a function $k : Y \times Y \to \mathbb{R}$ is a* **positive kernel function** *if*

$$\int_Y \int_Y f(x)k(x, y)f(y)dxdy \geq 0$$

*for any function $f \in L_2[Y]$.*

For any given positive kernel $k$ we may associate a $\Phi$ such that $k(x, y) = \langle\Phi(x), \Phi(y)\rangle$. In this case Optimization Problem (1) might be posed as

$$\begin{aligned} \min_{w\in\mathbb{R}^n,\ \zeta\in\mathbb{R}^m,\ b\in\mathbb{R}} \quad & \frac{1}{2}w^T w + C\sum_{i=1}^m \zeta_i \\ \text{s.t.} \quad & y_i(\langle w, \Phi(x_i)\rangle + b) \geq 1 - \zeta_i, \quad \zeta_i \geq 0. \end{aligned} \tag{2}$$

Given a solution, the classifier would be

$$f(z) = \text{sign}\left(\langle w, \Phi(z)\rangle + b\right).$$

Although the primal form of SVM has certain advantages - see Rahimi and Recht (2008), it is ill-suited to kernel learning. For this reason, we consider the dual formulation,

$$\begin{aligned} \max_{\alpha\in\mathbb{R}^m} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i\alpha_j y_i y_j \langle\Phi(x_i), \Phi(x_j)\rangle \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \ \ \forall \ \ i = 1, ..., m. \end{aligned} \tag{3}$$

In this case we may eliminate $\Phi$ from the optimization problem using $\langle\Phi(x_i), \Phi(x_j)\rangle = k(x_i, x_j)$ where the elements $k(x_i, x_j)$ define the kernel matrix. In this case, the resulting classifier is only a function of $k$ and becomes

$$f(z) = \text{sign}\left(\sum_{i=1}^m \alpha_i y_i k(x_i, z) + b\right).$$

Note that $b$ can be found a posteriori as the average of $y_j - \sum_{i=1}^m \alpha_i y_i k(x_j, x_i)$ for all $j$ such that $0 < \alpha_j < C$ - See Schölkopf et al. (2002). This implies that the primal variable $w$ is not explicitly required for the calculation of $b$, and that the resulting learned classifier, $f$, may be expressed solely in terms of $\alpha$ and the kernel function.

Commonly used positive kernel functions include the gaussian kernel $k_1(x,y) = \mathrm{e}^{(-\beta||x-y||^2)}$, where $\beta$ is the bandwidth (and must be chosen a priori) and the polynomial kernel $k_2(x,y) = (1 + x^T y)^d$ where $d$ is the degree of the polynomial.

Unfortunately Optimization Problem 3 requires that the kernel function, $k(x,y)$, be chosen a priori, a choice which significantly influences the accuracy of the resulting classifier $f$. We therefore alter the optimization problem by considering the kernel itself to be an optimization variable, constrained to lie in a given convex set of candidate positive kernel functions, $\mathcal{K}$. In this case, we have the following convex optimization problem.

$$\begin{aligned} \min_{k\in\mathcal{K}} \max_{\alpha\in\mathbb{R}^m} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{s.t.} \quad & \sum_{i=1}^m \alpha_i y_i = 0, \quad 0 \le \alpha_i \le C \;\; \forall \;\; i = 1, ..., m \end{aligned} \tag{4}$$

Having formulated the kernel learning problem, we now present two standard approaches to parameterizing the set of candidate kernels, $\mathcal{K}$, and solving the resulting convex optimization problem.

### 2.1. SDP-Based Kernel Learning Using Positive Kernel Matrices

We first consider the method of Lanckriet et al. (2004), wherein positive matrices were used to parameterize $\mathcal{K}$ for a given set of candidate kernels $\{k_i\}_{i=1}^l$ as

$$\mathcal{K} := \left\{ k(x,y) = \sum_{i=1}^l \mu_i k_i(x,y) : \; \mu \in \mathbb{R}^l, \; K_{ij} = k(x_i, x_j), \; K \ge 0 \right\},$$

where the $\{x_i\}_{i=1}^m \subset \mathbb{R}^n$ are the training points of the SVM problem and the $k_i$ were chosen a priori to be, for instance, Gaussian and polynomial kernels. It is significant to note that the PSD constraint on the kernel matrix $K$, enforces that the kernel matrix is PSD for the set of training data, but does not necessarily enforce that the kernel function itself is PD - meaning that kernels in $\mathcal{K}$ are not necessarily positive kernels.

Using this parameterized $\mathcal{K}$, the kernel optimization problem for the 1-norm soft margin support vector machine was formulated in Lanckriet et al. (2004) as the following semi-definite program, where $e$ is the vector of all ones.

$$\begin{aligned} \min_{\mu\in\mathbb{R}^l,\; t\in\mathbb{R},\; \gamma\in\mathbb{R},\; \nu\in\mathbb{R}^m,\; \delta\in\mathbb{R}^m} \quad & t \\ \text{subject to:} \quad & \begin{pmatrix} G & e + \nu - \delta + \gamma y \\ (e + \nu - \delta + \gamma y)^T & t - \frac{2}{m\lambda}\delta^T e \end{pmatrix} \ge 0 \\ & \nu \ge 0, \qquad \delta \ge 0, \qquad G_{ij} = k(x_i, x_j) y_i y_j \\ & k(x,y) = \sum_{i=1}^l \mu_i k_i(x,y) \end{aligned} \tag{5}$$

Note that here the original constraint $K \geq 0$ in $\mathcal{K}$ has been replaced by an equivalent constraint on $G$. This problem can now be solved using well-developed interior-point methods as in Alizadeh et al. (1998) with implementations such as MOSEK in ApS (2015).

In Optimization Problem (5), the size of the SDP constraint is $(m+1) \times (m+1)$ which is problematic in that the complexity of the resulting SDP grows as a polynomial in the number of training data. Methods that do not require this large semi-definite matrix constraint are explored next.

### 2.2. Kernel Learning Using MKL

In this subsection, we again take a set of basis kernels $\{k_i\}_{i=1}^l$ and consider the set of positive linear combinations,

$$\mathcal{K} := \left\{ k \;:\; k(x,y) = \sum_{i=1}^{l} \mu_i k_i(x,y),\; \mu_i \geq 0 \right\}. \tag{6}$$

Any element of this set is a positive kernel, replacing the matrix positivity constraint by a LP constraint.

$$\begin{aligned} \min_{\mu \geq 0} \max_{\alpha \in \mathbb{R}^m} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \sum_{k=1}^{l} \mu_k \alpha_i \alpha_j y_i y_j k_k(x_i, x_j) \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \;\; \forall \;\; i = 1, ..., m \end{aligned}$$

Use of this formulation is generally referred to as Multiple Kernel Learning (MKL). This formulation is a LP in $\mu$ for fixed $\alpha$ and a QP in $\alpha$ for fixed $\mu$. Recently, a number of highly efficient two-step methods have been proposed which exploit this formulation, including SimpleMKL as in Rakotomamonjy et al. (2008). These methods alternate between fixing $\mu$ and optimizing $\alpha$, then fixing $\alpha$ and optimizing $\mu$, adding the constraint that $\sum_i \mu_i = 1$ using a projected gradient descent. Other two-step solvers include Gönen and Alpaydın (2011). Two-step MKL solvers typically have a significantly reduced computational complexity compared with SDP-based approaches and can typically handle thousands of data points and thousands of basis kernels.

In Section 3, we propose a parameterization of kernels using positive matrices which avoids the need for the selection of basis kernels. Moreover, we show that this parameterization can be combined with MKL algorithms directly in SimpleMKL through the use of a randomly generated basis of kernels.

## 3. Positive Matrices Parameterize Positive Kernels

To begin this section, we propose a framework for using positive matrices to parameterize positive kernels. This is a generalization of a result initially proposed in Recht (2006). In Subsection 3.1 we apply this framework to obtain generalized polynomial kernels. In Subsection 3.2, we use the framework to obtain the TK class.

**Proposition 3** *Let $N$ be any bounded measurable function $N : \mathcal{X} \times Y \to \mathbb{R}^q$ on compact $\mathcal{X}$ and $Y$ and $P \in \mathbb{R}^{q\times q}$ be a positive semidefinite matrix $P \geq 0$. Then*

$$k(x,y) = \int_{\mathcal{X}} N(z,x)^T P N(z,y) dz \tag{7}$$

*is a positive kernel function.*

**Proof** Since $N$ is bounded and measurable, $k(x,y)$ is bounded and measurable. Since $P \geq 0$, there exists $P^{\frac{1}{2}}$ such that $P = (P^{\frac{1}{2}})^T P^{\frac{1}{2}}$. Now for any $f \in L_2[Y]$ define

$$g(z) = \int_Y P^{\frac{1}{2}} N(z,x) f(x) dx.$$

Then

$$\begin{aligned}\int_Y \int_Y f(x)k(x,y)f(y)dxdy &= \int_Y \int_Y \int_{\mathcal{X}} f(x)N(z,x)^T P N(z,y) f(y) dz dx dy \\ &= \int_{\mathcal{X}} \left( \int_Y P^{\frac{1}{2}} N(z,x) f(x) dx \right)^T \left( \int_Y N(z,y) P^{\frac{1}{2}} f(y) dy \right) dz \\ &= \int_{\mathcal{X}} g(z)^T g(z) dz \geq 0.\end{aligned}$$

■

For a given $N$, the map $P \mapsto k$ in Proposition 3 is linear. Specifically,

$$k(x,y) = \sum_{i,j} P_{i,j} G_{i,j}(x,y),$$

where,

$$G_{i,j}(x,y) = \int_{\mathcal{X}} N_i(z,x) N_j(z,y) dz.$$

### 3.1. Generalized Polynomial Kernels (GPK)

Let $Y = \mathbb{R}^n$ and define $Z_d : \mathbb{R}^n \to \mathbb{R}^q$ to be the vector of monomials of degree $d$. If we now define $N_P(z,y) = Z_d(y)$, then $k$ as defined in Proposition 3 is a polynomial of degree $2d$. The following result is from Peet et al. (2009).

**Lemma 4** *A polynomial $k$ of degree $2d$ is a positive polynomial kernel if and only if there exists some $P \geq 0$ such that*

$$k(x,y) = Z_d(x)^T P Z_d(y). \tag{8}$$

This lemma implies that a representation of the form of Equation (7) is necessary and sufficient for a generalized polynomial kernel to be positive. For convenience, we denote the set of generalized polynomial kernels of degree $d$ as follows.

$$\mathcal{K}_P^d := \{k \ : \ k(x,y) = Z_d(x)^T P Z_d(y) \ : \ P \geq 0\} \tag{9}$$

Unfortunately, however, polynomial kernels are never universal and hence we propose the following universal class of TK kernels, each of which is defined by polynomials, but which are not polynomial.

### 3.2. Tessellated Kernels

To begin, we define the indicator function for the positive orthant as

$$I_+(z) = \begin{cases} 1 & z \geq 0 \\ 0 & \text{otherwise,} \end{cases}$$

where $z \geq 0$ means $z_i \geq 0$ for all $i$. Now define $Z_d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^q$ to be the vector of monomials of degree $d$ in $\mathbb{R}^{2n}$. We now propose the following choice of $N : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{2q}$.

$$N_T^d(z,x) = \begin{bmatrix} Z_d(z,x)I_+(z-x) \\ Z_d(z,x)I_+(x-z) \end{bmatrix} = \begin{cases} \begin{bmatrix} Z_d(z,x) \\ 0 \end{bmatrix} & z \geq x \\ \begin{bmatrix} 0 \\ Z_d(z,x) \end{bmatrix} & x \geq z \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Equipped with this definition, we define the class of Tessellated Kernels as follows.

$$\mathcal{K}_T^d := \left\{ k \ : \ k(x,y) = \int_{\mathcal{X}} N_T^d(z,x)^T P N_T^d(z,y) dz, \ P \geq 0 \right\}, \qquad \mathcal{K}_T := \{k \ : \ k \in \mathcal{K}_T^d, \ d \in \mathbb{N}\}$$

### 3.3. Representation of TK Kernels Using Polynomials

The following result shows that any $k \in \mathcal{K}_T$ is piecewise polynomial. Specifically, if we define the partition of $\mathbb{R}^n$ into $2^n$ orthants - parameterized by $\beta \in \{0,1\}^n$ as $\{X_\beta\}_{\beta \in \{0,1\}^n}$ where

$$X_\beta := \left\{ x \in \mathbb{R}^n \ : \ \substack{x_j \geq 0 \text{ for all } j:\beta_j = 0, \\ x_i \leq 0 \text{ for all } i:\beta_i = 1} \right\}, \tag{11}$$

then for any $k \in \mathcal{K}_T$, there exist $k_\beta$ such that

$$k(x,y) = \Big\{ k_\beta(x,y), \quad \text{if } x - y \in X_\beta.$$

**Lemma 5** *Suppose that for $a < b \in \mathbb{R}^n$, $Y = \mathcal{X} = [a,b]$, $N$ is as defined in Eqn.* (10),

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} > 0,$$

*$k$ is as defined in Eqn.* (7) *and $\{X_\beta\}_{\beta \in \{0,1\}^n}$ is defined in Eqn.* (11). *Then*

$$k(x,y) = \Big\{ k_\beta(x,y) \quad \textit{if } \ x - y \in X_\beta. \tag{12}$$

*where the $k_\beta$ are polynomials defined as*

$$k_\beta(x,y) = \int\limits_{\beta_1 y_1 + (1-\beta_1)x_1}^{b_1} \cdots \int\limits_{\beta_n y_n + (1-\beta_n)x_n}^{b_n} Z_d(z,x)^T Q_1 Z_d(z,y) dz + k_0(x,y),$$

*where*

$$k_0(x,y) =\int_x^b Z_d(z,x)^T Q_2 Z_d(z,y)dz + \int_y^b Z_d(z,x)^T Q_3 Z_d(z,y)dz + \int_a^b Z_d(z,x)^T P_{22} Z_d(z,y)dz,$$

*and*

$$Q_1{=}P_{11} - P_{12} - P_{21} + P_{22}, \quad Q_2{=}P_{12} - P_{22}, \quad Q_3{=}P_{21} - P_{22}.$$

**Proof**

Given $N$ as defined above, if we partition $P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix}$ into equal-sized blocks, we have

$$k(x,y) = \int_{\mathcal{X}} N(z,x)^T P N(z,y) dz = \sum_{i,j=1}^{2} \int_{(x,y,z)\in\mathcal{X}_{ij}} Z_d(z,x)^T P_{i,j} Z_d(z,y)dz$$

where

$$\mathcal{X}_{ij} := \{(x,y,z) \in \mathbb{R}^{3n} \ : \ I_+((-1)^j(z-x))I_+((-1)^j(z-y)) = 1\}.$$

From the definition of $\mathcal{X}_{ij}$ we have that,

$$\begin{aligned} \mathcal{X}_{11} =& \{z \in \mathcal{X} \ : \ z_i \geq p_i^*(x,y), \ i = 1, \cdots, n\} \\ \mathcal{X}_{12} =& \{z \in \mathcal{X} \ : \ z_i \geq x_i, \ i = 1, \cdots, n\}/\mathcal{X}_{11} \\ \mathcal{X}_{21} =& \{z \in \mathcal{X} \ : \ z_i \geq y_i, \ i = 1, \cdots, n\}/\mathcal{X}_{11} \\ \mathcal{X}_{22} =& \mathcal{X}/\left(\mathcal{X}_{11} \cup \mathcal{X}_{12} \cup \mathcal{X}_{21}\right). \end{aligned}$$

where $p_i^*(x,y) = \max\{x_i, y_i\}$ and $p_i^*(x,y) = \beta_i y_i + (1-\beta_i)x_i$. By the definitions of $\mathcal{X}_{11}, \mathcal{X}_{12}, \mathcal{X}_{21}$, and $\mathcal{X}_{22}$ we have that,

$$\begin{aligned} k(x,y) =& \int_{p^*(x,y)}^b Z_d(z,x)^T (P_{11} - P_{12} - P_{21} + P_{22})\, Z_d(z,y)dz + \int_x^b Z_d(z,x)^T (P_{12} - P_{22})\, Z_d(z,y)dz \\ &+ \int_y^b Z_d(z,x)^T \left(P_{21} - P_{22}\right) Z_d(z,y)dz + \int_a^b Z_d(z,x)^T P_{22} Z_d(z,y)dz. \qquad (13) \end{aligned}$$

■

Note that the number of domains $X_\beta$ used to define the piecewise polynomial $k$ is $2^n$, which does not depend on $q$ (the dimension of $P_{ij}$). Thus, even if $Z_d = 1$, the resulting kernel is partitioned into $2^n$ domains. The size of $Z_d(x,y) \in \mathbb{R}^q$ only influences the degree of the polynomial defined on each domain.

The significance of the partition does not lie in the number of domains of the kernel, however. Rather, the significance of the partition lies in the resulting classifier, which, for a given set of training data $\{x_i\}_{i=1}^m$, has a domain tessellated into $(m+1)^n$ tiles, $X_\gamma$, where $\gamma \in \{0, \cdots, m\}^n$. Although the training data is unordered, we create an ordering using $\Gamma(i,j) : \{0, \cdots, m\} \times \{1, n\} \to \{1, \cdots, m\}$ where $\Gamma(i,j)$ indicates that among the $j$th elements of the training data, $x_{\Gamma(i,j)}$ has the $i$th largest value. That is,

$$[x_{\Gamma(i-1,j)}]_j \leq [x_{\Gamma(i,j)}]_j \leq [x_{\Gamma(i+1,j)}]_j \qquad \forall i = 1, \cdots, m-1, \quad j = 1, \cdots, n.$$

Now, for any $\gamma \in \{0, \cdots, m\}^n$, we may define an associated tile

$$X_\gamma := \left\{ z \;:\; [x_{\Gamma(\gamma_j, j)}]_j \le z_j \le [x_{\Gamma(\gamma_j + 1, j)}]_j,\; j = 1, \cdots, n \right\}.$$

The classifier may now be represented as

$$\begin{aligned} f(z) &= \sum_{i=1}^m \alpha_i y_i k(x_i, z) + b \\ &= f_\gamma(z) \;\; \forall z \in X_\gamma. \end{aligned}$$

To define the $f_\gamma$, we associate with every tile $\gamma$ and datum $i$ an orthant $\beta(i, \gamma)$ which denotes the position of tile $T_\gamma$ relative to datum $x_i$ - i.e. $T_\gamma$ is in the orthant $\beta(i, \gamma)$ centered at the point $x_i$. Specifically,

$$\beta(i, \gamma)_j = \begin{cases} 0 & [x_{\Gamma(\gamma_j, j)}]_j \ge [x_i]_j \\ 1 & \text{otherwise.} \end{cases}$$

Now we may define

$$f_\gamma(z) = \sum_{i=1}^m \alpha_i y_i k_{\beta(i,\gamma)}(x_i, z)$$

which is a polynomial for every $\gamma$. In this way, each data point further divides the domains which it intersects, resulting in $(m+1)^n$ disjoint sub-domains, each with associated polynomial classifier.

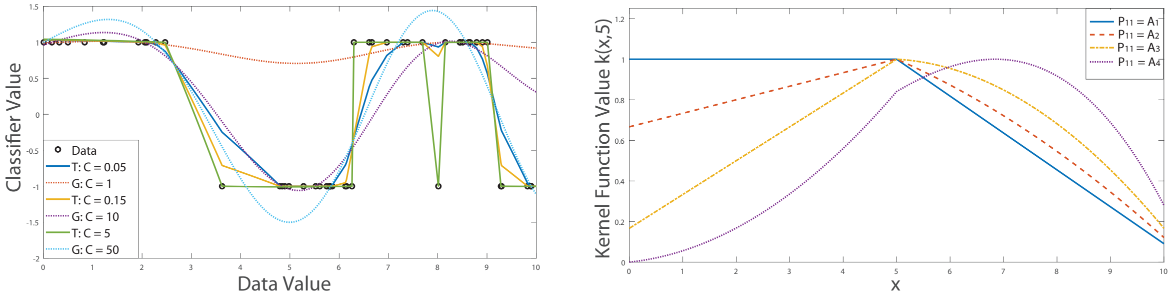
Thus we see that the number of domains of definition of the classifier grows quickly in $m$, the number of training data points. For instance, with n = 2 there are 100 tiles for just 9 data points. This growth is what makes TK kernels universal - as will be seen in Section IV.

In Figure 1(a) we see the function, $f(z) = \sum_{i=1}^m \alpha_i y_i k(x_i, z) + b$, for a degree 1 TK kernel trained for a 1-dimensional labeling problem as compared with a Gaussian kernel. We see that the TK classifier is continuous, and captures the shape of the generator better than the Gaussian. Note that the TK classifier is not continuously differentiable and the derivative can change precipitously at the edges of the tiles. However, if we decrease the inverse regularity weight $C$ in the objective function of Optimization Problem (2), then this has the effect of smoothing the resulting classifier. In Figure 1(a), as $C$ decreases we see that the changes in slope at edges of the tiles decrease.

To illustrate that the function $k(x_i, z)$ is a piecewise polynomial tessellated by the training datum, we plot the value of an assortment of TK kernels in one dimension in Figure 1(b). We use training datum $x_i = 5$, and a selection of different positive matrices where $P_{1,2} = P_{2,1} = P_{2,2} = 0$ and $P_{1,1} = A_i$ for $i = 1, \ldots, 4$ where

$$A_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & 0 \\ 0 & .1 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, A_4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{14}$$

In the first three cases the monomial basis is of degree 1, while in the fourth case the monomial basis is of degree 2 - for simplicity we exclude monomials with $z$. These different matrices all illustrate changes in slope which occur at the training datum.



(a) Optimal classifier, $f(z)$ for labelling a 1 dimensional data set using a degree one TK (solid lines), and a positive combination of Gaussian kernels (dotted lines) with three different penalty weights $C$.

(b) Normalized kernel function $k(5, z)$ using $P_{1,1} = A_i$ from (14) and $P_{1,2} = P_{2,1} = P_{2,2} = 0$.

Figure 1: This figure depicts the optimal classifier for labelling a 1-dimensional data set compared to Gaussian classifiers as well as the normalized kernel function, $k(5, z)$, using different $P_{11}$ matrices matrices and $\mathcal{X} = [0, 10]$.

## 4. Properties of the Tessellated Class of Kernel Functions

In this section, we prove that all TK kernels are continuous and universal and that the TK class is pointwise dense in all kernels.

### 4.1. TK Kernels Are Continuous

Let us begin by recalling that for any $P \geq 0$ and $N(z, x)$,

$$k(x, y) = \int_{\mathcal{X}} N(z, x)^T P N(z, y) dz$$

is a positive kernel and recall that for the TK kernels, we have

$$N(z, x) = \begin{bmatrix} Z_d(z, x) I_+(z - x) \\ Z_d(z, x) I_+(x - z) \end{bmatrix}.$$

By the representer theorem this implies that the classifiers consist of functions of the form

$$f(y) = \sum_{i=1}^{m} \alpha_i \int_{\mathcal{X}} N(x_i, z)^T P N(y, z) dz.$$

The following theorem establishes that such functions are necessarily continuous.

**Theorem 6** *Suppose that for $a < b \in \mathbb{R}^n$, $Y = \mathcal{X} = [a, b]$, $P \geq 0$, $N$ is as defined in Eqn.* (10) *for some $d \geq 0$ and $k$ is as defined in Eqn.* (7)*. Then $k$ is continuous and for any $\{x_i\}_{i=1}^m$ and $\alpha \in \mathbb{R}^m$, the function*

$$f(z) = \sum_{i=1}^{m} \alpha_i k(x_i, z),$$

*is continuous.*

**Proof** Partition $P$ as follows

$$P = \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} > 0.$$

To prove that $f(z)$ is continuous we need only prove that $k(x, y)$ is continuous. Applying Lemma 3 we may define $k(x, y)$ as

$$k(x, y) = \Big\{ k_\beta(x, y) \quad \text{if } \; x - y \in X_\beta. \tag{15}$$

where the $k_\beta$ are polynomials defined as

$$k_\beta(x, y) = \int_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int_{\theta_{\beta,n}(x,y)}^{b_n} Z_d(z, x)^T Q_1 Z_d(z, y) dz + k_0(x, y),$$

where $x - y \in X_\beta$, $\theta_{\beta,i}(x, y) = \beta_i y_i + (1 - \beta_i)x_i$, $Q_1 = P_{11} - P_{12} - P_{21} + P_{22}$, and $k_0(x, y)$ is a polynomial. To expand $k_\beta(x, y)$, we use multinomial notation for the monomials in $Z_d$. Specifically, we index the elements of $Z_d$ as $Z_d(z, x)_i = z^{\gamma_i} x^{\delta_i}$ where $\gamma_i, \delta_i \in \mathbb{N}^n$ for $i = 1, \cdots, q$ and where therefore $z^{\gamma_i} x^{\delta_i} = \prod_{j=1}^n z_j^{\gamma_{i,j}} x_j^{\delta_{i,j}}$. Then

$$\begin{aligned} \int\limits_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int\limits_{\theta_{\beta,n}(x,y)}^{b_n} Z_d(z, x)^T Q_1 Z_d(z, y) dz &= \int\limits_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int\limits_{\theta_{\beta,n}(x,y)}^{b_n} \sum_{k,l} (Q_1)_{k,l} \, x^{\delta_k} z^{\gamma_k} z^{\gamma_l} y^{\delta_l} dz \\ &= \sum_{k,l} (Q_1)_{k,l} \, x^{\delta_k} y^{\delta_l} \int\limits_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int\limits_{\theta_{\beta,n}(x,y)}^{b_n} z^{\gamma_k + \gamma_l} dz. \end{aligned} \tag{16}$$

Expanding the integrals in (16), each has the form

$$\begin{aligned} \int_{\theta_{\beta,1}(x,y)}^{b_1} \cdots \int_{\theta_{\beta,n}(x,y)}^{b_n} z^{\gamma_k + \gamma_l} \, dz &= \prod_{j=1}^n \; \int_{\theta_{\beta,j}(x,y)}^{b_j} z_j^{\gamma_{k,j} + \gamma_{l,j}} \, dz_j \\ &= \prod_{j=1}^n \; \frac{z_j^{\gamma_{k,j} + \gamma_{l,j} + 1}}{\gamma_{k,j} + \gamma_{l,j} + 1} \Bigg|_{\theta_{\beta,j}(x,y)}^{b_j} \\ &= \prod_{j=1}^n \; \frac{b_j^{\gamma_{k,j} + \gamma_{l,j} + 1}}{\gamma_{k,j} + \gamma_{l,j} + 1} - \frac{\theta_{\beta,j}(x, y)^{\gamma_{k,j} + \gamma_{l,j} + 1}}{\gamma_{k,j} + \gamma_{l,j} + 1}. \end{aligned}$$

Since $\theta_{\beta,j}(x, y)$ is equivalent to $\max(x_j, y_j)$, and can be written as the continuous function,

$$\theta_{\beta,j}(x, y) = \frac{1}{2}(x_j + y_j + |x_j - y_j|),$$

we conclude that $k(x, y)$ is the product and summation of continuous functions and therefore $k$ and the resulting classifiers are both continuous. ■

### 4.2. TK Kernels Are Universal

In addition to continuity, we show that any TK kernel with $P > 0$ has the universal property. Recall the following definition of universality.

**Definition 7** *A kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is said to be universal on the compact metric space $\mathcal{X}$ if it is continuous and there exists an inner-product space $\mathcal{W}$ and feature map, $\Phi : \mathcal{X} \to \mathcal{W}$ such that $k(x,y) = \langle \Phi(x), \Phi(y) \rangle_{\mathcal{W}}$ and where the unique Reproducing Kernel Hilbert Space (RKHS),*

$$\mathcal{H} := \{f \ : \ f(x) = \langle v, \Phi(x) \rangle, \ v \in \mathcal{W}\}$$

*with associated norm $\|f\|_{\mathcal{H}} := \inf_v \{\|v\|_{\mathcal{W}} \ : \ f(x) = \langle v, \Phi(x) \rangle\}$ is dense in $\mathcal{C}(\mathcal{X}) := \{f : \mathcal{X} \to \mathbb{R} \ : f \text{ is continuous}\}$ where $\|f\|_{\mathcal{C}} := \sup_{x \in X} |f(x)|$.*

The following theorem shows that any TK kernel with $P > 0$ is necessarily universal.

**Theorem 8** *Suppose $k$ is as defined in Eqn. (7) for some $P > 0$, $d \in \mathbb{N}$ and $N$ as defined in Eqn. (10). Then $k$ is universal for $Y = \mathcal{X} = [a,b]$, $a < b \in \mathbb{R}^n$.*

**Proof** Without loss of generality, we assume $Y = \mathcal{X} = [0,1]^n$. If $P > 0$, then there exist $\epsilon_i$ such that $P = P_0 + \sum_i \epsilon_i P_i$ where $P_0 > 0$ and

$$P_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \otimes \begin{bmatrix} e_1, 0, \dots, 0 \end{bmatrix}$$

where $\{e_1\}$ is the first canonical basis of $\mathbb{R}^n$. In this case

$$k(x,y) = \hat{k}(x,y) + \underbrace{\prod_{i=1}^{n} \epsilon_i \min\{x_i, y_i\}}_{k_1(x,y)},$$

where $\hat{k}$ is a positive kernel. Since the hypothesis space satisfies the additive property (See Wang et al. (2013) and Borgwardt et al. (2006)), if $k_1$ is a universal kernel, then $k$ is a universal kernel.

Recall that for a given kernel, the hypothesis space, $\mathcal{H}$, can be characterized as the closure of $\text{span}\{k(y, \cdot) \ : \ y \in \mathcal{X}\}$. Now, consider

$$\text{span}\{k_1(y, \cdot) \ : \ y \in \mathcal{X}\},$$

which consists of all functions of the form

$$f(x) = \sum_j c_j \prod_{i=1}^{n} \min\{[y_j]_i, x_i\}.$$

Now

$$\min\{[y_j]_i, x_i\} = \begin{cases} x_i, & \text{if } x_i \le [y_j]_i \\ [y_j]_i, & \text{otherwise.} \end{cases}$$

For $n = 1$, we may construct a triangle function of height 1 centered at $y_2$ as

$$f(x) = \sum_{i=1}^{3} \frac{\alpha_i}{\epsilon} k_1(y_i, x) = \begin{cases} 0, & \text{if } x < y_1 \\ \delta(x - y_1), & \text{if } y_1 \leq x < y_2 \\ 1 - \delta(x - y_2), & \text{if } y_2 \leq x < y_3 \\ 0, & \text{if } y_3 < x, \end{cases}$$

where $\delta = y_1 - y_2 = y_2 - y_3$, and

$$\alpha_1 = -\delta, \quad \alpha_2 = 2\delta, \quad \alpha_3 = -\delta.$$

By taking the product of triangle functions in each dimension, we obtain the pyramid functions which are known to be dense in the space of continuous functions on a compact domain (See Shekhtman (1982)). We conclude that $k_1$ is a universal kernel and hence $k$ is universal. ■

This theorem implies that even if the degree of the polynomials is small, the kernel is still universal. Specifically, in the case when $n = 1$ and $d = 0$, the set $\mathcal{K}_T^0$ is universal yet contains only three parameters (the elements of the symmetric $P \in \mathbb{R}^{2\times 2}$).

### 4.3. TK Kernels Are Pointwise Dense in All Kernels

In the previous two subsections, we have shown that TK kernels are continuous and universal. Furthermore, as shown in Section 3, the TK class admits a linear parameterization. The remaining question, then, is whether TK kernels are superior in some *performance* metric to other classes of universal kernels such as Gaussian kernels. First, note that the universal property is of the kernel itself and which is extended to a class of kernels by requiring all kernels in that class to satisfy the property. However, although a kernel may be universal, it may not be well-suited to SVM. Expanding on this point, although it is known that any universal kernel may be used to separate a given set of data, it can be shown that for any given set of normalized data, $\{x_i, y_i\}$, there exists a universal kernel, $k$, for which the solution to Optimization Problems (2) and (3) is arbitrarily suboptimal - e.g. by increasing the bandwidth of the Gaussian kernel.

To address the question of performance, we propose the *pointwise density* property. This property is defined on a set of kernels and guarantees that there is some kernel in the set of kernels for which the solution to Optimization Problems (2) and (3) is optimal. Specifically, we have the following.

**Definition 9** *The set of kernels $\mathcal{K}$ is said to be* ***pointwise dense*** *if for any positive kernel, $k^*$, any set of data $\{x_i\}_{i=1}^m$, and any $\epsilon > 0$, there exists $k \in \mathcal{K}$ such that $\|k(x_i, x_j) - k^*(x_i, x_j)\| \leq \epsilon$.*

This definition implies that a set of kernels can approximate any given positive kernel arbitrarily well. To illustrate the importance of the pointwise density property, in Subsubsection 4.3.1 we show that for a large class of kernel learning problems, the value of the optimal kernel is not pointwise positive - i.e. $k(x, y) \ngeq 0$ for all $x, y \in \mathcal{X}$. This is significant because almost all commonly used kernels are pointwise non-negative. Indeed we find that the elements of the optimal kernel matrix are negative as frequently as they are positive.

### 4.3.1. Optimal Kernels Are Not Pointwise Positive

To demonstrate the necessity of negative values in optimal kernel matrices, we analytically solve the following SDP derived from Optimization Problem (5) which determines the optimal kernel matrix ($K^*$) given the labels $y$ of a problem and a "penalty" parameter $C$, but with no constraint on the form of the kernel function (other than it be PD).

$$\begin{aligned} &\min_{t\in\mathbb{R},K\in\mathbb{R}^{m\times m},\gamma\in\mathbb{R},\nu\in\mathbb{R}^m,\delta\in\mathbb{R}^m} \quad t, \qquad (17)\\ &\text{subject to:} \begin{pmatrix} G & e+\nu-\delta+\gamma y \\ (e+\nu-\delta+\gamma y)^T & t-C\delta^T e \end{pmatrix} \geq 0 \\ &\nu\geq 0, \qquad \delta \geq 0, \qquad K\geq 0, \qquad \text{trace}(K)=m, \qquad G_{i,j}=y_iK_{i,j}y_j \end{aligned}$$

The following theorem finds an analytic solution of this optimization problem.

**Theorem 10** *Let $y_i \in \{1,-1\}$ for $i=1,\cdots,m$ and $C\geq \frac{2}{m}$, then the solution to Optimization Problem 17 is,*

$$\nu^*=0, \quad \gamma^*=-\sum_{i=1}^m \frac{y_i}{m}, \quad \delta^*=0, \quad t^*=\frac{\|e-\gamma^* y\|_2}{m},$$

*and $K^*=\frac{m}{\|e+\gamma^* y\|_2^2}\mathcal{Y}(e+\gamma^*y)(e+\gamma^*y)^T\mathcal{Y}$ where $\mathcal{Y}=diag(y)$.*

**Proof** We first show that $K^*=U\Sigma U^T$, where

$$U=\mathcal{Y}\begin{bmatrix}\frac{(e+\nu^*-\delta^*+\gamma^*y)}{\|(e+\nu^*-\delta^*+\gamma^*y)\|_2} & \cdots\end{bmatrix}, \Sigma=\begin{bmatrix} m & 0\\ 0 & 0\end{bmatrix}.$$

Optimization Problem (17) is equivalent to

$$\begin{aligned} &\min_{K\in\mathbb{R}^{m\times m},\gamma\in\mathbb{R},\nu\in\mathbb{R}^m,\delta\in\mathbb{R}^m} \quad (e+\nu-\delta+\gamma y)^T(\mathcal{Y}K\mathcal{Y})^{-1}(e+\nu-\delta+\gamma y)+2C\delta^T e \qquad (18)\\ &\text{subject to:} \quad \nu\geq 0, \qquad \delta\geq 0, \qquad K\geq 0, \qquad \text{trace}(K)=m. \end{aligned}$$

This problem can be separated into subproblems as

$$\min_{\substack{\gamma\in\mathbb{R},\nu\in\mathbb{R}^m,\delta\in\mathbb{R}^m\\ \nu\geq 0,\ \delta\geq 0}} \quad \min_{\substack{K\in\mathbb{R}^{m\times m},\\ K\geq 0,\ \text{trace}(K)=m}} \quad (e+\nu-\delta+\gamma y)^T(\mathcal{Y}K\mathcal{Y})^{-1}(e+\nu-\delta+\gamma y)+2C\delta^T e.$$

Now, for any feasible $K$, we have that $K\geq 0$ and $\bar{\sigma}(K)\leq m$ and hence

$$\begin{aligned} (e+\nu-\delta+\gamma y)^T(\mathcal{Y}K\mathcal{Y})^{-1}(e+\nu-\delta+\gamma y) &\geq \frac{1}{\bar{\sigma}(K)}\|e+\nu-\delta+\gamma y\|_2^2 \\ &\geq \frac{1}{m}\|e+\nu-\delta+\gamma y\|_2^2. \end{aligned}$$

Now, we propose $K = U\Sigma U^T$ and show that it is optimal, where

$$U = \mathcal{Y}\begin{bmatrix} \frac{(e+\nu^*-\delta^*+\gamma^* y)}{\|(e+\nu^*-\delta^*+\gamma^* y)\|_2} & V \end{bmatrix}, \quad \Sigma = \begin{bmatrix} m & 0 \\ 0 & 0 \end{bmatrix}.$$

and $V$ is any unitary completion of the matrix $U$. Then $K \geq 0$, trace$(K) = m$, and

$$\begin{aligned} &(e+\nu-\delta+\gamma y)^T\,(\mathcal{Y}K\mathcal{Y})^{-1}(e+\nu-\delta+\gamma y) \\ &= (e+\nu-\delta+\gamma y)^T\left(\begin{bmatrix} \frac{(e+\nu-\delta+\gamma y)}{\|(e+\nu-\delta+\gamma y)\|_2} \, V \end{bmatrix}\begin{bmatrix} m & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \frac{(e+\nu-\delta+\gamma y)}{\|(e+\nu-\delta+\gamma y)\|_2} \, V \end{bmatrix}^T\right)^{-1}(e+\nu-\delta+\gamma y) \\ &= \frac{\|(e+\nu-\delta+\gamma y)\|_2^2}{m}. \end{aligned}$$

We conclude that this $K$ solves the first sub-problem and hence Optimization Problem (17) reduces to

$$\min_{\substack{\gamma\in\mathbb{R},\nu\in\mathbb{R}^m,\delta\in\mathbb{R}^m \\ \nu\geq 0,\ \delta\geq 0,}} \quad \frac{\|(e+\nu-\delta+\gamma y)\|_2^2}{m} + 2C\delta^T e. \tag{19}$$

Now let $\nu^*, \delta^*, \gamma^*$ be as defined in the theorem statement. For the convex objective

$$f(\delta,\nu,\gamma) = \frac{\|e+\nu-\delta+\gamma y\|_2^2}{m} + 2C\delta^T e$$

let $\bar{y} = \frac{1}{m}\sum_{i=1}^m y_i$ and we have that

$$\frac{\partial f}{\partial \nu_i}(\nu^*,\delta^*,\gamma^*) = \frac{2+2\bar{y}y_i}{m} \geq \frac{2+-2(1)(1)}{m} \geq 0,$$

and for $C \geq \frac{2}{m}$

$$\frac{\partial f}{\partial \delta_i}(\nu^*,\delta^*,\gamma^*) = \frac{2mC-2-2\bar{y}y_i}{m} \geq \frac{4-2-2(1)(1)}{m} \geq 0.$$

Finally

$$\frac{\partial f}{\partial \gamma}(\nu^*,\delta^*,\gamma^*) = \frac{1}{m}\sum_{i=1}^m 2y_i - 2\bar{y} = -2\bar{y} + 2\frac{1}{m}\sum_{i=1}^m y_i = 0.$$

Hence the KKT conditions are satisfied and since the optimization problem is convex, $(\nu^*,\delta^*,\gamma^*)$ is optimal. ■

This result shows that for binary labels, the optimal kernel matrix has an analytic solution. Furthermore, if we consider the case where $\sum_{i=1}^m y_i = 0$, then $\lambda^* = 0$ and hence $K^* = yy^T$ and $K^*_{i,j} = y_i y_j$. This implies that the optimal kernel matrix consists of an equal number of positive and negative entries - meaning that kernels functions with globally positive values will not be able to approximate the optimal kernel matrix well. Furthermore, for values of $C$ less than $\frac{2}{m}$, we find numerically that the same kernel matrix is still optimal - only the values of $\delta^*$ and $\gamma^*$ are different.

### 4.3.2. The GPK and TK Classes Are Pointwise Dense in All Kernels

Having demonstrated the significance of pointwise density, we now establish that both the GPK and TK kernel sets satisfy this property. For this subsubsection, we relax the strict positivity constraint $P > 0$ in the definition of the TK class. In this case, the GPK class becomes a subset of the TK class. We prove pointwise density of the GPK class - a property which is then inherited by the TK class. The following lemma shows that the GPK class is a subset of the TK class.

**Lemma 11** $\mathcal{K}_P^d \subset \mathcal{K}_T^d$

**Proof** If $k_p \in \mathcal{K}_P^d$, there exists a $P_1 \geq 0$ such that $k_p(x,y) = Z_d(x)^T P_1 Z_d(y)$. Now let $J$ be the matrix such that $JZ_d(z,x) = Z_d(x)$ and define

$$P = \frac{1}{\prod_{j=1}^n (b_j - a_j)} \begin{bmatrix} J^T P_1 J & J^T P_1 J \\ J^T P_1 J & J^T P_1 J \end{bmatrix} \geq 0.$$

Now let $k$ be as defined in Equation (7). Then $k \in \mathcal{K}_T^d$ and

$$\begin{aligned} k(x,y) =& \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_{p^*(x,y)}^b Z_d(z,x)^T J^T \left(P_1 - P_1 - P_1 + P_1\right) J Z_d(z,y) dz \\ &+ \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_x^b Z_d(z,x)^T J^T \left(P_1 - P_1\right) J Z_d(z,y) dz \\ &+ \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_y^b Z_d(z,x)^T J^T \left(P_1 - P_1\right) J Z_d(z,y) dz \\ &+ \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_a^b Z_d(z,x)^T J^T P_1 J Z_d(z,y) dz \\ =& \frac{1}{\prod_{j=1}^n (b_j - a_j)} \int_a^b Z_d(x)^T P_1 Z_d(y) dz \\ =& k_p(x,y). \end{aligned}$$

We conclude that $k_p = k \in \mathcal{K}_T^d$. ■

We now use polynomial interpolation to prove that GPK kernels are pointwise dense.

**Theorem 12** *For any kernel matrix $K^*$ and any finite set $\{x_i\}_{i=1}^m$, there exists a $d \in \mathbb{N}$ and $k \in \mathcal{K}_P^d$ such that if $K_{i,j} = k(x_i, x_j)$, then $K = K^*$.*

**Proof** Since $K^* \geq 0$, $K^* = M^T M$ for some $M$. Using multivariate polynomial interpolation (as in Gasca and Sauer (2001)), for sufficiently large $d$, we may choose $Q$ such that

$$Q \begin{bmatrix} Z_d(x_1) & \cdots & Z_d(x_m) \end{bmatrix} = M.$$

Now let

$$k(x,y) = Z_d(x)^T P Z_d(y)$$

where $P = Q^T Q \geq 0$. Now partition $M$ as

$$M = \begin{bmatrix} m_1 & \cdots & m_m \end{bmatrix}.$$

Then $QZ_d(x_i) = m_i$ and hence

$$\begin{aligned} K_{ij} &= Z_d(x_i)^T Q^T Q Z_d(x_j) \\ &= m_i^T m_j \\ &= K^*_{ij}. \end{aligned}$$

■

### 4.3.3. GPK and TK Kernels Converge Quickly to the Optimal Kernel

In Subsubsection 4.3.1, we obtained an analytical solution to the optimal kernel matrix. In Subsubsection 4.3.2, we used polynomial interpolation to prove that GPK and TK kernels are pointwise dense in the set of all kernels. However, the degree of the polynomials used in this proof increases with the number of interpolation points. In this subsubsection, we show that in practice, a degree of only 4 or 5 can be sufficient to approximate the optimal kernel matrix with minimal error using either GPK and TK kernels.

Specifically, we consider the problem of approximating the optimal kernel matrix for a given set of data $\{x_i\}$ and given set of kernels, $\mathcal{K}$, using both the element-wise matrix $\|\cdot\|_1$ and $\|\cdot\|_\infty$ norms.

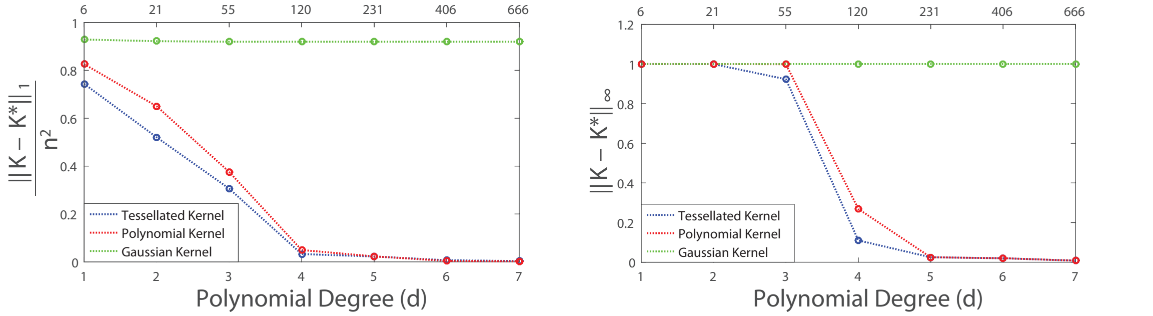
$$\min_{k \in \mathcal{K}} \frac{\|K - K^*\|_1}{n^2} \quad s.t. \;\; K_{i,j} = k(x_i, x_j) \tag{20}$$

$$\min_{k \in \mathcal{K}} \|K - K^*\|_\infty \quad s.t. \;\; K_{i,j} = k(x_i, x_j) \tag{21}$$

The sets of kernels functions we consider are: $\mathcal{K}_G^\gamma$ - the sum of $K$ Gaussians with bandwiths $\gamma_i$; $\mathcal{K}_P^d$ - the GPKs of degree $d$; and $\mathcal{K}_T^d$ - the TK kernels of degree $d$. That is, we choose $\mathcal{K} \in \{\mathcal{K}_G^\gamma,\ \mathcal{K}^d,\ \mathcal{K}_T^d\}$ where for convenience, we define the class of sums of Gaussian kernels of bandwidths $\gamma \in \mathbb{R}^K$ as follows.

$$\mathcal{K}_G^\gamma := \left\{ k \;:\; k(x,y) = \sum_{i=1}^K \mu_i \mathrm{e}^{\frac{||x-y||_2^2}{\gamma_i}} \;:\; \mu_i > 0 \right\} \tag{22}$$

We now solve Optimization Problems (20) and (21) for $\mathcal{K}_G^\gamma$, $\mathcal{K}_P^d$, and $\mathcal{K}_T^d$ as a function of the degree of the polynomials, $d$ and the number of bandwidths selected ($K$). For this test, we use the spiral data set with 20 samples and corresponding labels such that $\sum_{i=1}^m y_i = 0$. Since half of the entries in $K^*$ are $-1$, and since the Gaussian kernel is globally positive, it is easy to see that for $\mathcal{K} = \mathcal{K}_G^\gamma$ the minimum objective values of Optimization Problems (20) and (21) are lower bounded by 0.5 and 1 respectively, irrespective of the choice of bandwidths, $\gamma_i$ and number of data points. In Figs. 2(a) and 2(b) we numerically show the change in the objective value of Optimization Problems (20) and (21) for the optimal Gaussian, GPK, and TK kernels as we increase the complexity of the kernel function. For the TK and GPK kernel



(a) $\frac{\|K-K^*\|_1}{n^2}$ for the TK and GPK classes of degree $d$ and for a positive combination of $m$ Gaussian kernels.

(b) $\|K-K^*\|_\infty$ for the TK and GPK classes of degree $d$ and for a positive combination of $m$ Gaussian kernels.

Figure 2: The objective of Optimization Problem 20 and 21 for the TK and GPK classes of degree $d$ and for a positive combination of $m$ Gaussian kernels with bandwidths ranging from .01 to 10. The number of bandwidths is selected so that the number of decision variables match in the Gaussian and in the TK kernel case.

functions, we increase the complexity of the kernel function by increasing the degree of the monomial basis while scaling the $x$-axis to ensure equivalent computational complexity.

The results demonstrate that, as expected, the Gaussian kernel saturates with an objective value significantly larger than the lower bound of 0.5 for the 1-norm and exactly at 1 for the $\infty$-norm (the projected lower bound). Meanwhile, as the degree increases, both the GPK and TK kernels are able to approximate the kernel matrix arbitrarily well, with almost no error at degree $d = 7$. Furthermore, the TK kernels converge somewhat faster.

Note that the optimization problem considered in this subsubsection only concerns the approximation of a kernel function. In Section 8 we will show that TK kernels also outperform Gaussians when solving the kernel learning SVM Problem (4) for standard data sets.

## 5. SDP Formulation of the TK Kernel Learning Algorithm

Section 2 gave a convex formulation of the kernel learning problem using the convex constraint $k \in \mathcal{K}$. Having now defined the TK class of kernels, we now address specific implementations of the TK kernel learning problem using both an SDP method based on Optimization Problem (5) and a method based on the SimpleMKL toolbox. In both cases, our goal for this section is to define an explicit linear map from the elements of the positive matrix variable, $P$, to the values of the kernel function $k(x_i, x_j)$.

To construct our mapping, we first create an index of the elements in the basis $Z_d(z, x)$ which is used in $N_T^d(z, x)$ as defined in Eqn. (10). Recall $Z_d(z, x)$ is a vector of all monomials of degree $d$ or less of length $q := \binom{d+2n}{d}$. We now specify that the elements of $Z_d$ are ordered, and by default we use lexicographical ordering on the exponents of the variables of the monomials. Specifically, we denote the jth monomial in $Z_d(z, x)$ where $z, x \in \mathbb{R}^n$ as $z^{\gamma_j} x^{\delta_j} := \prod_{i=1}^n z_i^{\gamma_{j,i}} x_i^{\delta_{j,i}}$ where $\gamma_j, \delta_j \in \mathbb{N}^n$ and $\{[\gamma_j, \delta_j]\}_{j=1}^q$ is ordered lexicographically. Note that $\{[\gamma_j, \delta_j]\}_{j=1}^q = \{x \in \mathbb{N}^{2n} : \|x\|_1 \leq d\}$. Using this notation, we have the following representation of the TK kernel $k$.

**Corollary 13** *Suppose that for $a < b \in \mathbb{R}^n$, $Y = \mathcal{X} = [a,b]$, and $d \in \mathbb{N}$ we define the finite set $D_d := \{(\gamma,\delta) \in \mathbb{N}^{2n} : \|(\gamma,\delta)\|_1 \le d\}$. Let $\{[\gamma_i,\delta_i]\}_{i=1}^q \subseteq D_d$ be some ordering of $D_d$ and define $Z_d(z,x)_j = z^{\gamma_j} x^{\delta_j}$. Now let $k$ be as defined in Eqn. (7) for some $P > 0$ and where $N$ is as defined in Eqn. (10). If we partition $P = \begin{bmatrix} Q & R \\ R^T & S \end{bmatrix}$ then we have,*

$$k(x,y) = \sum_{i,j=1}^{q} Q_{i,j} g_{i,j}(x,y) + R_{i,j} t_{i,j}(x,y) + R_{i,j}^T t_{i,j}(y,x) + S_{i,j} h_{i,j}(x,y)$$

*where $g_{i,j}, t_{i,j}, h_{i,j} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ are defined as*

$$\begin{aligned} g_{i,j}(x,y) &:= x^{\delta_i} y^{\delta_j} T(p^*(x,y), b, \gamma_i + \gamma_j + \mathbf{1}) \\ t_{i,j}(x,y) &:= x^{\delta_i} y^{\delta_j} T(x, b, \gamma_i + \gamma_j + \mathbf{1}) - g_{i,j}(x,y) \\ h_{i,j}(x,y) &:= x^{\delta_i} y^{\delta_j} T(a, b, \gamma_i + \gamma_j + \mathbf{1}) - g_{i,j}(x,y) - t_{i,j}(x,y) - t_{i,j}(y,x), \end{aligned} \tag{23}$$

*where $\mathbf{1} \in \mathbb{N}^n$ is the vector of ones, $p^* : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is defined elementwise as $p^*(x,y)_i = \max\{x_i, y_i\}$, and $T : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{N}^n \to \mathbb{R}$ is defined as*

$$T(x,y,\zeta) = \prod_{j=1}^{n} \left( \frac{y_j^{\zeta_j}}{\zeta_j} - \frac{x_j^{\zeta_j}}{\zeta_j} \right).$$

**Proof** The proof follows from Theorem 6. ■

An illustration of this map using lexicographical indexing is given in the appendix. Using a linear map from the elements of $P$ to the value of $k(x,y)$, we may now write the SDP version of the TK kernel learning problem as follows.

$$\min_{t\in\mathbb{R},\ \gamma\in\mathbb{R},\ \nu\in\mathbb{R}^m,\ \delta\in\mathbb{R}^m,\ Q,R,S\in\mathbb{R}^{q\times q}} \quad t \tag{24}$$

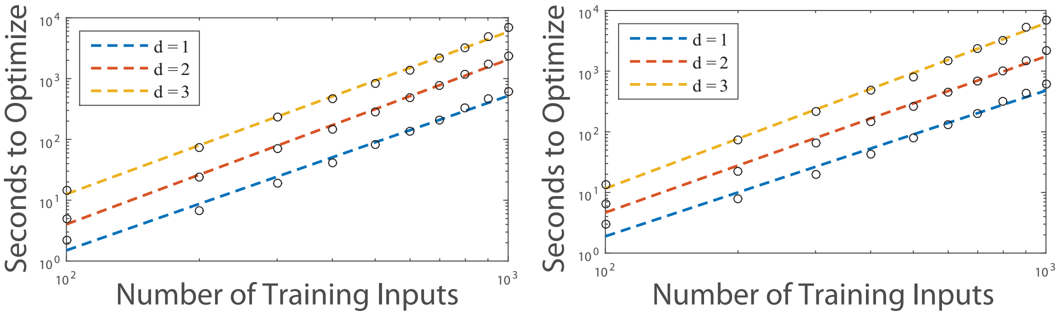
$$\text{subject to: } \begin{pmatrix} G(P) & e + \nu - \delta + \gamma y \\ (e + \nu - \delta + \gamma y)^T & t - \frac{2}{m\lambda}\delta^T e \end{pmatrix} \ge 0,$$

$$\nu \ge 0, \qquad \delta \ge 0, \qquad P = \begin{bmatrix} Q & R \\ R^T & S \end{bmatrix} > 0, \qquad \text{trace}(P) \le 1,$$

$$G_{k,l}(P) = y_k y_l \sum_{i,j=1}^{q} Q_{i,j} g_{i,j}(x_k,x_l) + R_{i,j} t_{i,j}(x_k,x_l) + R_{i,j}^T t_{i,j}(x_l,x_k) + S_{i,j} h_{i,j}(x_k,x_l)$$

Optimization Problem (24), then, is an SDP and can, therefore, be solved efficiently using standard SDP solvers such as MOSEK in ApS (2015). Note that we use the trace constraint to ensure the kernel function is bounded.

Typically SDP problems require roughly $p^2 n^2$ number of operations, where $p$ is the number of decision variables and $n$ is the dimension of the SDP constraint (See Doherty et al. (2004)). The number of decision variables in (24) is moderate, increasingly linearly in the number of training data points and the number of elements of $P$. However, this optimization problem has a semi-definite matrix constraint whose dimension is linear in $m$, the number of training data. As we will see in Section 7, the increase in training data increases $n$ and limits the amount of training data that can be processed using Optimization Problem (24). To improve the scalability of the algorithm, we consider a variation on SimpleMKL.



(a) Complexity Scaling for Identification of Circle

(b) Complexity Scaling for Identification of Spiral

Figure 3: Log-Log Plot of Computation Time vs number of training data for 2-feature kernel learning.

## 6. SimpleMKL Formulation of the TK Kernel Learning Algorithm

Recall that SimpleMKL searches for an optimal positive linear combination of kernel functions from the set (6). The algorithm returns a vector of positive weights $\mu$, corresponding to each kernel in the set of a priori selected kernel functions $k_s(x, y)$. Here we discuss how SimpleMKL as implemented in Rakotomamonjy et al. (2008) can be used to find optimal combinations of TK kernels.

To create a basis set of TK kernels, we randomly generate a set of $L$ positive semi-definite matrices, $P^s$ for $s = 1, \dots, L$ and use SimpleMKL to find the optimal linear combination of the TK kernels defined by each matrix $P^s = \begin{bmatrix} Q^s & R^s \\ (R^s)^T & S^s \end{bmatrix}$. Using the basis kernels

$$k_s(x, y) = \sum_{i,j}^{q} Q^s_{i,j} g_{i,j}(x, y) + R^s_{i,j} t_{i,j}(x, y) + R^s_{j,i} t_{i,j}(y, x) + S^s_{i,j} h_{i,j}(x, y)$$

where $g(x, y),\ t(x, y),$ and $h(x, y)$ are as defined in Eq. (23), we now have

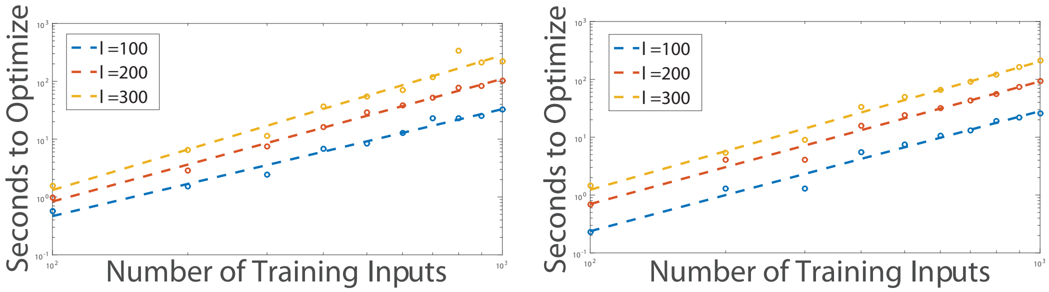
$$\begin{aligned} \min_{\mu \geq 0} \max_{\alpha \in \mathbb{R}^m} \quad & \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \sum_{s=1}^{L} \sum_{l,m=1}^{2q} \mu_s \alpha_i \alpha_j y_i y_j k_s(x_i, x_j) \qquad (25) \\ \text{s.t.} \quad & \sum_{i=1}^{m} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C \ \ \forall \ \ i = 1, ..., m. \end{aligned}$$

While the current use of randomly generated matrices is somewhat heuristic, it may be avoided through the development of a dedicated two-step algorithm - wherein the first step optimizes $\alpha$ for a fixed $P$ and the second step fixes $\alpha$ and searches over the positive matrices.

In Section 7 we will perform a numerical analysis of the complexity of both the SDP and SimpleMKL implementations.

## 7. Implementation and Complexity Analysis

In this section we first analyze the complexity of Optimization Problem (24) with respect to the number of training points as well as the selected degree of the TK - $\mathcal{K}^d_T$. We then



(a) Complexity Scaling for Identification of Circle using SimpleMKL with TK kernels.

(b) Complexity Scaling for Identification of Spiral using SimpleMKL with TK kernels.

Figure 4: Log-Log Plot of Computation Time vs number of training data for 2-feature kernel learning using SimpleMKL with TK kernels.

perform the same analysis on Optimization Problem (25) with respect to the number of training points and the number of random matrices selected.

**Analysis of the SDP Approach:** In Optimization Problem (24) the constraint that the kernel be a positive TK kernel can be expressed as an LMI constraint with variables $P_{ij}$. Using Optimization Problem (24), if $P \in \mathbb{R}^{q\times q}$, and $m$ is the number of training data, with a Mosek implementation, we find experimentally that the complexity of the resulting SDP scales as approximately $m^{2.6} + q^{1.9}$ as can be seen in Fig. 3 and is similar to the complexity of other methods such as the hyperkernel approach in Ong et al. (2005). These scaling results are for training data randomly generated by two standard 2-feature example problems (circle and spiral - See Fig. 6) for degrees $d = 1, 2, 3$ and where $d$ defines the length of $Z_d$ (and hence $q$) which is the vector of all monomials in 2 variables of degree $d$ or less.

Note that the length of $Z_d$ scales with the degree and number of features, $n$, as $q = \frac{(n+d-1)!}{n!d!}$. For a large number of features and a high degree, the size of $Z_d$ will become unmanageably large. Note, however, that, as indicated in Section 4, even when $d = 0$, every TK kernel is universal.

**Analysis of the SimpleMKL Approach:** Solving Optimization Problem (25) with SimpleMKL we first need to generate a set of random matrices. If we have $L$ random positive semi-definite matrices and $m$ training data points then we find experimentally that the complexity of the resulting SDP scales as approximately $m^{2.1} + L^{1.6}$ as can be seen in Fig. 4. These scaling results are, as in the results for the SDP method, for training data randomly generated by two standard 2-feature example problems (circle and spiral - See Fig. 6). We select the number of training data $m$, to vary between 100 and 1000 points and select the number of random matrices to be $L = 100, 200, 300$.

Note that the complexity of the SimpleMKL version is largely independent of the selected degree of the polynomial. However, a larger degree means that the matrices $P$ are larger, and therefore a larger number of random positive semi-definite matrices, $L$, should be selected.

## 8. Accuracy and Comparison With Existing Methods

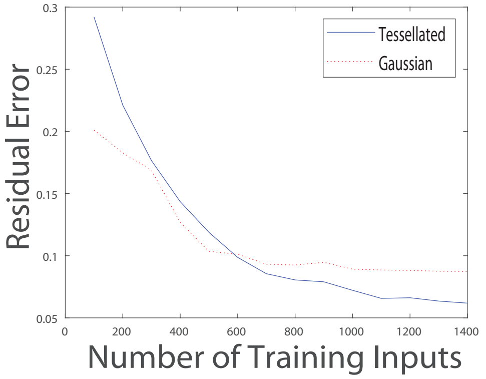


Figure 5: TSA compared with SimpleMKL for spiral data set with artificial additive noise.

In this section, we evaluate the relative accuracy of Optimization Problem (24) using SDP and Optimization Problem (25) using SimpleMKL. To evaluate the accuracy, we applied 5 variations of the kernel learning problem to 5 randomly selected benchmark data sets from the UCI Machine learning Data Repository - Liver, Cancer, Heart, Pima, and Ionosphere. In all evaluations of Test Set Accuracy (TSA), the data is partitioned into 80% training data and 20% testing and this partition is repeated 30 times to obtain 30 sets of training and testing data. For all numerical tests we use the soft-margin problem with regularization parameter $C$, where $C$ is selected from a set of values picked a priori by 5-fold cross-validation. To perform 5-fold cross-validation we split the training data set into five groups, solve the optimization problem using each potential value of $C$ on four of the five groups and test the optimal classifier performance on the remaining group. We repeat this process using each of the five groups as the test set and select the value of $C$ which led to the best average performance.

The 5 variations on the kernel learning problem are

[**TK**] We use the SDP algorithm in (24) using $d = 1$ (Except Ionosphere, which uses $d = 0$) but set $\gamma = 0$ to decrease numerical complexity. To determine the integral in (24), we first scaled the data so that $x_i \in [0, 1]^n$, and then set $\mathcal{X} := [0 - \epsilon, 1 + \epsilon]^n$, where $\epsilon > 0$ was chosen by 5-fold cross-validation.

[**SimpleMKL**] We use SimpleMKL with a standard selection of Gaussian and polynomial kernels with bandwidths arbitrarily chosen between .5 and 10 and polynomial degrees one through three - yielding approximately $13(n + 1)$ kernels;

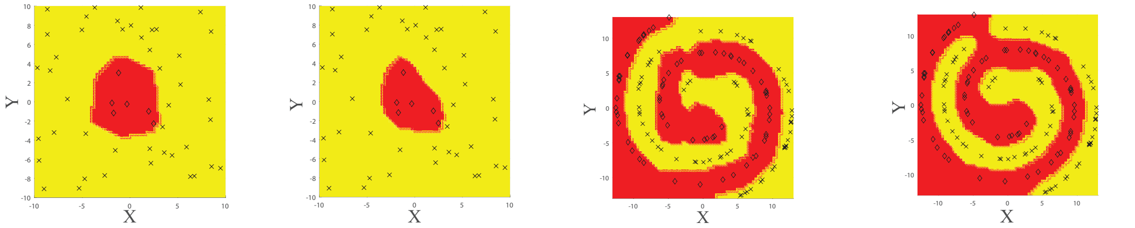
[**SimpleMKL TK**] We randomly generated a sequence of 300 positive semidefinite matrices and use these as the SimpleMKL library of kernels;

[**SimpleMKL TK+**] We combined the libraries in [SimpleMKL] and [SimpleMKL TK] into a single SimpleMKL implementation;

[**Neural Net**] We use 3 layer neural network with 50 hidden layers using MATLABs (`patternnet`) implementation.

In Table 1, we see the average TSA for these four approaches as applied to several randomly selected benchmark data sets from the UCI Machine learning Data Repository. In all cases, either [TK] or [SimpleMKL TK] met or in some cases significantly exceeded the accuracy of [SimpleMKL].

In addition to the standard battery of tests, we performed a secondary analysis to demonstrate the advantages of the TK class when the ratio of training data to the number of features is high. For this analysis, we use the liver data set (6 features) and the spiral discriminant (2 features) from Lang (1988) (we also briefly examine the unit circle). For the liver data set, in Figure 8, we see a semilog plot of the residual error (i.e. 1-TSA) as the size of the training data increases as compared with SimpleMKL. This figure shows consistent improvement of [TK] over standard usage of [SimpleMKL]. For the spiral case, in Figure 8



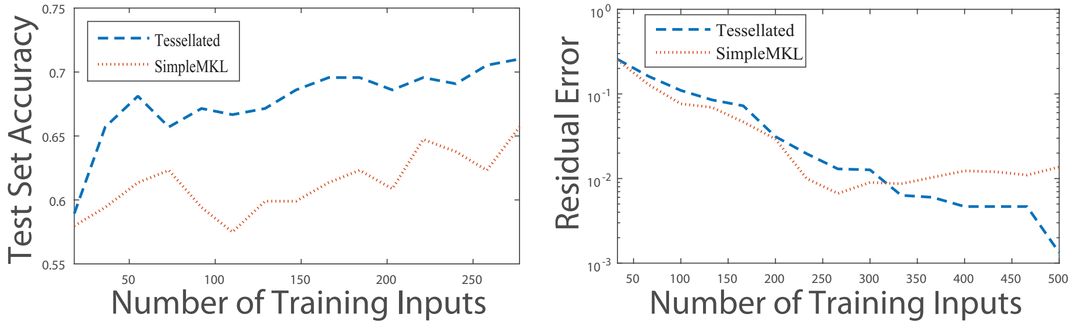
(a) Circle data set classified with [TK] (n=50)

(b) Circle data set classified with [SimpleMKL] (n=50)

(c) Spiral data set classified with [TK] (n=150)

(d) Circle data set classified with [SimpleMKL] (n=150)

Figure 6: Discriminant Surface for Circle and Spiral Separator using method [TK] as Compared with [SimpleMKL] for $n$ training data.



(a) Average test set accuracy on the Liver data set vs. the number of training data for [TK] compared to [SimpleMKL]

(b) Semilog plot of residual error on generated 2D spiral data vs. number of training data for [TK] compared to [SimpleMKL].

Figure 7: Plots demonstrating the change in accuracy of [TK] and [SimpleMKL] with respect to the number of training inputs. The residual error is defined as 1-TSA where TSA is the test set accuracy.

we again see a semilog plot of the residual error as the size of the training data increases as compared with [SimpleMKL]. In this case, both methods converge well with [TK] showing significant improvement over [SimpleMKL] only for very large training data sets.

To explore how increasing the number of training data affects the classifier, we generated a new 1400 point training data set with additive noise of zero mean and $\sigma = .1$. The results are seen in Figure 5. In this case, we see that [TK] significantly outperform [SimpleMKL] beginning at 600 data points.

Finally, as an illustration, we plotted the discriminant surface for both the spiral and unit circle data sets using both the [TK] and [SimpleMKL] methods using 150 training data points. These 2D surfaces are found in Figure 6.

## 9. Conclusion

In this paper, we have proposed a new class of universal kernel functions. This set of kernels can be parameterized directly using positive matrices or indirectly using positive coefficients combined with randomly generated positive matrices. Furthermore, any element of this

Table 1: TSA comparison for algorithms [TK], [SimpleMKL], [SimpleMKL TK], [SimpleMKL TK+], and [Neural Net]. The maximum TSA for each data set is bold. The average TSA, standard deviation of TSA and time to compute are shown below. $m$ is size of data set and $n$ the number of features.

| Data Set | Method | Accuracy | Time | Data Features |
|---|---|---|---|---|
| Liver | TK | **72.32 ± 4.92** | 95.75 ± 2.68 | m = 346<br>n = 6 |
| | SimpleMKL | 65.51 ± 5.10 | 2.61 ± 0.42 | |
| | SimpleMKL TK | 70.58 ± 4.69 | 8.37 ± 0.30 | |
| | SimpleMKL TK+ | 70.53 ± 4.79 | 14.70 ± 0.76 | |
| | Neural Net | 66.32 ± 7.46 | 0.14 ± 0.04 | |
| Cancer | TK | **97.18 ± 1.48** | 636.17 ± 25.43 | m = 684<br>n = 9 |
| | SimpleMKL | 96.55 ± 1.34 | 14.74 ± 1.33 | |
| | SimpleMKL TK | 96.89 ± 1.43 | 45.84 ± 4.28 | |
| | SimpleMKL TK+ | 96.89 ± 1.42 | 65.08 ± 10.52 | |
| | Neural Net | 96.67 ± 1.30 | 0.18 ± 0.06 | |
| Heart | TK | 83.46 ± 4.56 | 221.67 ± 29.63 | m = 271<br>n = 13 |
| | SimpleMKL | 83.70 ± 4.77 | 3.09 ± 0.19 | |
| | SimpleMKL TK | **84.38 ± 4.34** | 55.48 ± 2.67 | |
| | SimpleMKL TK+ | 83.64 ± 4.54 | 13.23 ± 2.70 | |
| | Neural Net | 78.64 ± 5.19 | 0.12 ± 0.01 | |
| Pima | TK | 76.32 ± 3.10 | 1211.66 ± 27.01 | m=769<br>n = 8 |
| | SimpleMKL | 76.00 ± 3.33 | 19.04 ± 2.33 | |
| | SimpleMKL TK | **76.75 ± 2.81** | 34.65 ± 23.28 | |
| | SimpleMKL TK+ | 76.57 ± 2.72 | 96.20 ± 30.42 | |
| | Neural Net | 75.35 ± 2.98 | 0.24 ± 0.19 | |
| Ionosphere | TK | **93.24 ± 3.04** | 6.69 ± 0.27 | m = 352<br>n = 34 |
| | SimpleMKL | 92.16 ± 2.78 | 26.24 ± 2.78 | |
| | SimpleMKL TK | 87.65 ± 2.88 | 8.28 ± .16 | |
| | SimpleMKL TK+ | 92.16 ± 2.78 | 50.77 ± 2.98 | |
| | Neural Net | 90.85 ± 3.42 | 0.16 ± 0.02 | |

class is universal in the sense that the hypothesis space is dense in $L_2$, giving it comparable performance to and properties of the Gaussian kernels. However, unlike the Gaussian or RBFs, the TK class does not require a set of bandwidths to be chosen a priori. Furthermore, by increasing the degree of the monomial basis, we have shown that the TK class can approximate any kernel matrix arbitrarily well.

We have demonstrated the effectiveness of the TK class on several data sets from the UCI repository. We have shown that the computational complexity is comparable to other SDP-based kernel learning methods. Furthermore, by using a randomized basis for the positive matrices, we have shown that the TK class can be readily integrated with existing multiple kernel learning algorithms such as SimpleMKL - yielding similar results with less computational complexity. In most cases, either the optimal TK kernel or the MKL learned sub-optimal TK kernel will outperform or match an MKL approach using Gaussian and polynomial kernels with respect to the Test Set Accuracy. Finally, we note that this universal class of kernels can be trivially extended to matrix-valued kernels for use in, e.g. multi-task learning as in Caponnetto et al. (2008).

## Acknowledgments

This work was supported by NSF Grant 026257-001.

## Appendix A. Example Calculation of g, h, and t

In this appendix, we illustrate the lexicographical ordering $\{[\gamma_i, \delta_i]\}$ and the maps $g_{i,j}(x, y)$, $t_{i,j}(x, y)$, $t_{i,j}(y, x)$ and $h_{i,j}(x, y)$ for the special case of $a = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $b = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$. We choose $d = 1$ which implies $Z_d$ is length $q = 5$. This yields

$$Z_1(z, x) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ z_1 \\ z_2 \end{bmatrix},$$

The associated $\delta$ and $\gamma$ are then given by

$$\delta = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \gamma = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

For brevity we will assemble the matrices $g_{i,j}(x, y)$, $t_{i,j}(x, y)$, $t_{i,j}(y, x)$ and $h_{i,j}(x, y)$ for the first, second and fourth elements of $Z_1(z, x)$.

Assembling $g_{i,j}(x, y)$, $t_{i,j}(x, y)$, $t_{i,j}(y, x)$ and $h_{i,j}(x, y)$ for $i, j = 1, 2, 4$ into matrices of size $\mathbb{R}^{3\times 3}$, and by using Equation (23) in Corollary 13, we have that,

$$\begin{aligned}
g(x,y) &= \begin{bmatrix} (3-p_1^*)(4-p_2^*) & y_1(3-p_1^*)(4-p_2^*) & \frac{1}{2}(9-(p_1^*)^2)(4-p_2^*) \\ x_1(3-p_1^*)(4-p_2^*) & x_1y_1(3-p_1^*)(4-p_2^*) & x_1\frac{1}{2}(9-(p_1^*)^2)(4-p_2^*) \\ \frac{1}{2}(9-(p_1^*)^2)(4-p_2^*) & y_1\frac{1}{2}(9-(p_1^*)^2)(4-p_2^*) & \frac{1}{3}(27-(p_1^*)^3)(4-p_2^*) \end{bmatrix}, \\
t(x,y) &= \begin{bmatrix} (3-x_1)(4-x_2) & y_1(3-x_1)(4-x_2) & \frac{1}{2}(9-x_1^2)(4-x_2) \\ x_1(3-x_1)(4-x_2) & x_1y_1(3-x_1)(4-x_2) & x_1\frac{1}{2}(9-x_1^2)(4-x_2) \\ \frac{1}{2}(9-x_1^2)(4-x_2) & y_1\frac{1}{2}(9-x_1^2)(4-x_2) & \frac{1}{3}(27-x_1^3)(4-x_2) \end{bmatrix} - g(x,y), \\
t(y,x) &= \begin{bmatrix} (3-y_1)(4-y_2) & x_1(3-y_1)(4-y_2) & \frac{1}{2}(9-y_1^2)(4-y_2) \\ y_1(3-y_1)(4-y_2) & x_1y_1(3-y_1)(4-y_2) & y_1\frac{1}{2}(9-y_1^2)(4-y_2) \\ \frac{1}{2}(9-y_1^2)(4-y_2) & x_1\frac{1}{2}(9-y_1^2)(4-y_2) & \frac{1}{3}(27-y_1^3)(4-y_2) \end{bmatrix} - g(x,y), \\
h(x,y) &= \begin{bmatrix} 12 & 12y_1 & 36 \\ 12x_1 & 12x_1y_1 & 36x_1 \\ 36 & 36y_1 & 108 \end{bmatrix} - g(x,y) - t(x,y) - t(y,x),
\end{aligned}$$

where we have defined $p^*$ element wise as $p_i^* = \max\{x_i, y_i\}$.

## References

F. Alizadeh, J.-P. Haeberly, and M. Overton. Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM Journal on Optimization*, 1998.

MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).*, 2015.

K. M. Borgwardt, A. Gretton, M. J. Rasch, H. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 2006.

A. Caponnetto, C. Micchelli, M. Pontil, and Y. Ying. Universal multi-task kernels. *Journal of Machine Learning Research*, 2008.

M. Collins and N. Duffy. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems*, 2002.

C. Cortes, M. Mohri, and A. Rostamizadeh. Learning non-linear combinations of kernels. In *Advances in Neural Information Processing Systems*, 2009.

C. Cortes, M. Mohri, and A. Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 2012.

A. C. Doherty, P. A. Parrilo, and F. M. Spedalieri. Complete family of separability criteria. *Physical Review A*, 2004.

E. Eskin, J. Weston, W. Noble, and C. Leslie. Mismatch string kernels for SVM protein classification. In *Advances in Neural Information Processing Systems*, 2003.

K. Gai, G. Chen, and C. S. Zhang. Learning kernels with radiuses of minimum enclosing balls. In *Advances in Neural Information Processing Systems*, 2010.

T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Learning Theory and Kernel Machines*. 2003.

M. Gasca and T. Sauer. On the history of multivariate polynomial interpolation. In *Numerical Analysis: Historical Developments in the 20th Century*. 2001.

M. Gönen and E. Alpaydin. Localized multiple kernel learning. In *Proceedings of the International Conference on Machine learning*, 2008.

M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 2011.

D. Haussler. Convolution kernels on discrete structures. Technical report, University of California in Santa Cruz, 1999.

A. Jain, S. Vishwanathan, and M. Varma. SPF-GMKL: generalized multiple kernel learning with a million kernels. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining*, 2012.

G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 2004.

K. Lang. Learning to tell two spirals apart. In *Proceedings of the Connectionist Models Summer School*, 1988.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2002.

C. Micchelli, Y. Xu, and H. Zhang. Universal kernels. *Journal of Machine Learning Research*, 2006.

C. S. Ong, A. J. Smola, and R. C. Williamson. Learning the kernel with hyperkernels. *Journal of Machine Learning Research*, 2005.

M. M. Peet, A. Papachristodoulou, and S. Lall. Positive forms and stability of linear time-delay systems. *SIAM Journal on Control and Optimization*, 2009.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 2008.

B. Recht. *Convex Modeling with Priors.* PhD thesis, Massachusetts Institute of Technology, 2006.

B. Schölkopf, A. J. Smola, and F. Bach. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

B. Shekhtman. Why piecewise linear functions are dense in C [0, 1]. *Journal of Approximation Theory*, 1982.

S. Ć. Sonnenburg, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, V. Franc, et al. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 2010.

N. Subrahmanya and Y. Shin. Sparse multiple kernel learning for signal processing applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

H Sun. Mercer theorem for RKHS on noncompact sets. *Journal of Complexity*, 2005.

H. Wang, Q. Xiao, and D. Zhou. An approximation theory approach to learning with $\ell_1$ regularization. *Journal of Approximation Theory*, 2013.

E. Zanaty and A. Afifi. Support vector machines (SVMs) with universal kernels. *Applied Artificial Intelligence*, 2011.