Using SDP to Parameterize Universal Kernel Functions

Brendon K. Colbert[1] and Matthew M. Peet[2]

Abstract— **We propose a new class of universal kernel functions which admit a linear parametrization using positive semidefinite matrices. We refer to kernels of this class as Tessellated Kernels (TKs) due to the observation that if applied to kernel-based learning algorithms, the resulting discriminants are defined by continuous piecewise-polynomial functions with hyper-rectangular domains whose vertices are determined by the training data. The number of parameters used to define these TKs is determined by the length of an associated monomial basis. However, even for a single monomial basis function the TKs are universal in the sense that the resulting discriminants occupy a hypothesis space which is dense in L_2. This implies that the use of TKs for learning the kernel (aka kernel learning) can obviate the need for Gaussian kernels and associated problem of selecting bandwidth - a conclusion verified through extensive numerical testing on soft margin Support Vector Machine (SVM) problems. Furthermore, our results show that when the ratio of the number of training data to features is high, the proposed method will significantly outperform other algorithms for learning the kernel. Finally, TKs can be integrated efficiently with existing Multiple Kernel Learning (MKL) algorithms such as SimpleMKL.**

I. INTRODUCTION

This paper addresses the problem of automated selection of an optimal kernel function for a given kernel-based machine learning problem (i.e. soft margin SVM). Kernel functions implicitly define a linear parametrization of nonlinear candidate maps $y = f(x)$ from features 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.

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 semidefinite programming (SDP) approach proposed by [10] which directly imposed kernel matrix positivity using a linear subspace of candidate kernel functions (as in MKL). Because of the complexity of semidefinite programming, more recent work has focused on gradient methods for convex and non-convex parameterizations of positive linear combinations of candidate kernels, as in SimpleMKL [15] or the several variations in [19]. These methods rely on kernel set operations (addition, multiplication, convolution) to generate large numbers of parameterized kernel functions as in [5]. When the parameterization is non-convex, gradient-based methods find local minima and include GMKL as introduced in [9]. See, e.g. [8] for a comprehensive review of MKL algorithms.

In this paper, we focus on the class of "Universal Kernels" formalized in [12].

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 a given positive definite kernel, $\mathcal{H}$ exists, is unique, and can be characterized using the Riesz representation theorem [20] as the closure of 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

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

The most well-known example of a universal kernel is the Gaussian (generalized in [22]). However, most other common kernels are not universal, including, significantly, the polynomial class of kernels (this is significant because polynomials admit a linear parameterization).

In this paper, we propose a new class of universal kernel functions which are not polynomials, yet are defined by polynomials and admit a convex parametrization. Specifically, if $\mathcal{X} := \{x \in \mathbb{R}^n \;:\; x_i \in [a_i, b_i]\}$ and the inequality $>$ is defined by the positive orthant, we consider kernels $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ of the form

$$k(x, y) = \int_{\mathcal{X}} I(z, x) Z(z, x)^T P Z(z, y) I(z, y) dz,$$
$$where \quad I(z, x) = \begin{cases} 1, & \text{if } z > x \\ 0, & \text{if } z \not> x \end{cases},$$

and where $Z : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^m$ is a vector of monomials and $P \in \mathbb{S}^m$. We show in Section III that if $P > 0$, then k is a positive kernel, continuous and universal.

To illustrate, we show how this class of kernel can be rigorously incorporated into both the SDP kernel learning

*This work was supported by the National Science Foundation under grant No. 1931270, No. 1935453 and No. 026257-001

[1] Brendon K. Colbert is with the Department of Mechanical Engineering, Arizona State University, Tempe, AZ, 85298 US brendon.colbert@asu.edu

[2] Matthew M. Peet is with Faculty of Mechanical Engineering, Arizona State University Tempe, AZ, 85298 US mpeet@asu.edu

framework and the MKL framework for SVM soft margin problems. In the numerical results we illustrate this improved performance on a number of UCI repository data sets.

II. An overview of the optimal kernel learning problem for the 1-norm SVM 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, for a given linear parameterization of kernel functions, in Subsections A and B, we then present two standard algorithms for solving the kernel learning problem. These algorithms will then be applied in Section III to our class of Tessellated Kernels (TKs).

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 *linear* 1-norm soft margin problem as

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

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

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

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

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

$$\int_{\mathcal{X}}\int_{\mathcal{X}} f(x)k(x,y)f(y)dxdy \geq 0$$

for any function $f \in L_2[\mathcal{X}]$.
In this case, the classifier becomes

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

where α solves the associated dual problem

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

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 [17]. 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 α.

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

Unfortunately optimization problem 2 requires that the kernel function, $k(x,y)$, be chosen a priori. The selection of a kernel function, however, can have a large effect on the accuracy of the resulting classifier f. We therefore consider methods for selecting an optimal kernel function from a convex set of kernel functions $\mathcal{K}$. In this case, we have the following convex optimization problem.

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

In the following two subsections we present two standard approaches to parameterizing $\mathcal{K}$ and solving the resulting convex optimization problem.

A. Formulating the kernel optimization problem for linear combinations of kernel functions

We first consider the method of [10], wherein positive matrices were used to parameterize $\mathcal{K}$ for a given set of candidate kernels $\{k_i\}_{i=1}^l$ as

$$\begin{aligned}\mathcal{K} := \{k(x,y) = \sum_{i=1}^l \mu_i k_i(x,y) :& \\ \mu\in\mathbb{R}^l, K_{ij} = k(x_i,x_j),\ K\geq 0\},&\end{aligned}$$

where the x_i are the training points of the SVM problem and the k_i were chosen a priori to be 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 positive definite - 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 as the following semi-definite program,

$$\begin{aligned}\min_{\mu\in\mathbb{R}^l,G\in\mathbb{R}^{m\times m},t\in\mathbb{R},\gamma\in\mathbb{R},\boldsymbol{\nu}\in\mathbb{R}^m,\boldsymbol{\delta}\in\mathbb{R}^m} \quad & t, \\ \text{subject to:} \quad & \begin{pmatrix} G & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\gamma\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\gamma\mathbf{y})^T & t-\frac{2}{m\lambda}\boldsymbol{\delta}^T\mathbf{e}\end{pmatrix} \geq 0 \\ & \boldsymbol{\nu}\geq 0, \qquad \boldsymbol{\delta}\geq 0, \qquad G_{ij} = k(x_i,x_j)y_iy_j \\ & k(x,y) = \sum_{i=1}^l \mu_i k_i(x,y) \end{aligned} \tag{4}$$

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 [1] with implementations such as MOSEK [2].

In Optimization Problem (4), 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. Our parameterization, introduced in Section III, avoids this computational scaling by proposing kernel positivity tests whose complexity is independent of the amount of training data. Furthermore, our method does not require the a priori selection of a set of basis kernels.

B. Formulating the kernel learning optimization problem for positive linear combinations of kernel functions

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

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

Any element of this set is a positive kernel, replacing the matrix positivity constraint by an 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). It has the disadvantage that it is non-convex in native form. Recently, however, a number of highly efficient two-step methods have been proposed to solve the associated kernel learning problem, including SimpleMKL [15]. These methods first fix μ_i and optimize over α, then fix α and optimize over μ, adding the constraint that $\sum_i \mu_i = 1$ using a projected gradient descent. Other two-step solvers such as [8] solve the second step using LP. 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 III, 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 either directly in SimpleMKL [15] through the use of a randomly generated basis of kernels, or through a new algorithm which modifies the second step to optimize over the set of positive matrices.

III. Positive “tessellated” kernel functions can be parameterized by positive matrices

In this section, we propose a general framework for using positive matrices to parameterize a class of tessellated kernel functions. The following result is based on a parametrization of positive integral operators initially proposed in [16].

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

$$k(x,y) = \int_{\mathcal{X}} N(z,x)^T P N(z,y) dz \tag{5}$$

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 define

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

Then

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

■

Polynomial Kernels Let $X = \mathbb{R}^n$ and $\mathcal{X} = \mathbb{R}^p$ and define $Z_d : \mathbb{R}^n \rightarrow \mathbb{R}^q$ to be the vector of monomials of degree d. In this case, it was shown in [14] that k is a degree $2d$ positive polynomial kernel if and only if there exists some $P \geq 0$ such that

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

This implies that a representation of the form of Equation (5) is necessary and sufficient to represent all positive polynomial kernels. Unfortunately, polynomial kernels are not universal and hence we propose the following universal class of tessellated kernels, each of which is defined by polynomials, but which are not polynomial.

Tessellated Kernels As defined in [7], a kernel $k(x,y)$ is *semi-separable* if there exist functions A_i such that

$$k(x,y) = \begin{cases} A_1(x) A_2(y), & \text{if } x > y \\ A_3(x) A_4(y), & \text{otherwise.} \end{cases}$$

Semi-separable kernels define a broader class of integral operators include, e.g. the Volterra operators. To parameterize such as class of kernels, we first replace $x > y$ with the constraints $x - y \in S_1 \subset \mathbb{R}^n$ and $x - y \in S_2 \subset \mathbb{R}^n$ where the S_1 is the positive orthant and S_2 is the negative orthant. We now define the following indicator function

$$I_S(z,x) = \begin{cases} 1 & z - x \in S \\ 0 & \text{otherwise,} \end{cases}$$

Now let $X = \mathcal{X} = \mathbb{R}^n$ and define $Z_d : \mathcal{X} \times X \rightarrow \mathbb{R}^q$ to be the vector of monomials of degree d in $\mathbb{R}^{2n}$. We propose the following definition for $N : \mathcal{X} \times X \rightarrow \mathbb{R}^{2q}$.

$$N(z,x) = \begin{bmatrix} Z_d(z,x) I_{S_1}(z,x) \\ Z_d(z,x) I_{S_2}(z,x) \end{bmatrix}. \tag{6}$$

Using Eqn. (5), the associated kernel function is,

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

A Partition of the Tessellated Kernel In this part, we partition the domain X into 2^n orthants and by expanding the integral and show that a tessellated kernel is piecewise polynomial, using polynomial k_β indexed to each domain X_β.

Lemma 4: Suppose that for $a < b \in \mathbb{R}^n$, $X = \mathcal{X} = [a,b]$, N is as defined in Eqn. (6)

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

and k is as defined in Eqn. (5). Then if we define the partition of $\mathbb{R}^n \times \mathbb{R}^n$ into 2^n orthants - parameterized as $\{X_\beta\}_{\beta\in\{0,1\}^n}$ where

$$X_\beta := \left\{(x,y) \in \mathbb{R}^n \times \mathbb{R}^n \;:\; \begin{smallmatrix} x_j \geq y_j \text{ for all } j:\beta_j=0, \\ y_i \geq x_i \text{ for all } i:\beta_i=1 \end{smallmatrix}\right\},$$

we have that

$$k(x,y) = \Big\{ k_\beta(x,y) \quad \text{if} \quad (x,y) \in X_\beta. \tag{7}$$

where the k_β are polynomials defined as

$$k_\beta(x,y) =$$

$$\prod_{i:\beta_i=0} \int_{z_i=x_i}^{b_i} \prod_{j:\beta_j=1} \int_{z_j=y_j}^{b_j} Z_d(z,x)^T Q_1 Z_d(z,y) dz + k_0(x,y)$$

$$k_0(x,y) = \int_x^b Z_d(z,x)^T Q_2 Z_d(z,y) dz$$

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

$$Q_1{=}P_{11}-P_{12}-P_{21}-P_{22}, \;\; Q_2{=}P_{12}-P_{22}, \;\; Q_3{=}P_{21}-P_{22}$$

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

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

where

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

From the definition of X_{ij} we have that,

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

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

$$\begin{aligned} k(x,y) = &\int_{p^*(x,y)}^b Z_d(z,x)^T (P_{11} - P_{12} - P_{21} - P_{22}) \, Z_d(z,y) dz \\ &+ \int_x^b Z_d(z,x)^T (P_{12} - P_{22}) \, Z_d(z,y) dz \\ &+ \int_y^b Z_d(z,x)^T \left(P_{21} - P_{22}\right) Z_d(z,y) dz \\ &+ \int_a^b Z_d(z,x)^T P_{22} Z_d(z,y) dz. \end{aligned} \tag{8}$$

■

Note that the number of domains X_β 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 length of $Z_d(x,y) \in \mathbb{R}^q$ only influences the degree of the polynomial defined on each domain.

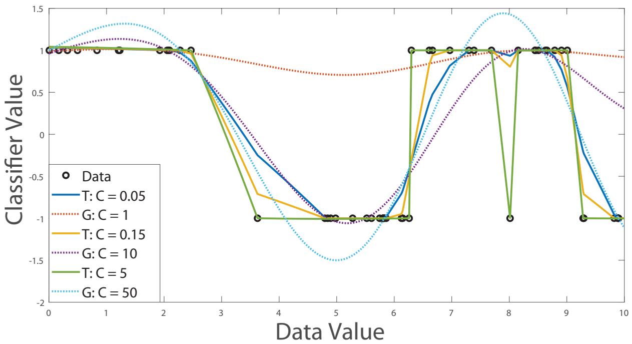


Fig. 1: Optimal classifier, $f(z)$ for labelling a 1 dimensional dataset using a degree one tessellated kernel (solid lines), and a positive combination of Gaussian kernels (dotted lines) with three different penalty weights C. Note that as C increases the maximum slope of $f(z)$ for the tessellated kernel and the maximum value of $f(z)$ for the Gaussian kernel also increases.

The significance of the partition does not lie in the number of domains, however. Rather, the significance lies in the resulting classifier, which it defined by the input data $\{x_i\}_{i=1}^m$ and has the form

$$\begin{aligned} f(z) &= \sum_{i=1}^m \alpha_i y_i k(x_i, z) + b \\ &= \Big\{ \textstyle\sum_{i=1}^m \alpha_i y_i k_\beta(x_i, z) \quad \text{if} \quad (x_i, z) \in X_\beta. \\ &= \Big\{ f_{i,\beta}(z) \quad \text{if} \quad z \in X_{i,\beta}. \end{aligned}$$

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

$$X_{i,\beta} := \left\{ z \;:\; \begin{smallmatrix} (x_i)_j \geq z_j \text{ for all } j:\beta_j=0, \\ z_k \geq (x_i)_k \text{ for all } k:\beta_i=k \end{smallmatrix} \right\}$$

where the $f_{i,\beta}$ are polynomials. 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 grows quickly in the number of training data points m. For instance with n = 2 there are 100 sub-domains for just 9 data points. This growth is what makes tessellated kernels universal - as will be seen in Section IV.

In Figure 1 we see the function, $f(z) = \sum_{i=1}^m \alpha_i y_i k(x_i, z) + b$, for a degree 1 tessellated kernel function trained for a 1-dimensional labeling problem as compared with a Gaussian kernel. We see that the tessellated kernel is continuous, and captures the shape of the generator better than the Gaussian. However, the kernel is not continuously differentiable and this property must be imposed using the inverse regularity weight C in the objective function on Eqn (1). In Figure 1, as C decreases we see that the changes in slope at edges of the domain decrease.

IV. Properties of the Tessellated Class of Kernel Functions

In this section we prove that tessellated kernel functions are both continuous and universal, even in the simplest case of degree $d = 0$.

Theorem 5: Suppose that for $a < b \in \mathbb{R}^n$, $X = \mathcal{X} = [a, b]$, $P \geq 0$, N is as defined in Eqn. (6) for $d \geq 0$ and k is as defined in Eqn. (5). Then for any $\{x_i\}_{i=1}^m$, 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 4 we may define $k(x, y)$ as

$$k(x, y) = \left\{ k_\beta(x, y) \quad \text{if} \quad (x, y) \in X_\beta. \right. \tag{9}$$

where the k_β are polynomials defined as

$$\begin{aligned} & k_\beta(x, y) = \\ & \prod_{i:\beta_i=0} \int_{z_i=x_i}^{b_i} \prod_{j:\beta_j=1} \int_{z_j=y_j}^{b_j} Z_d(z, x)^T Q_1 Z_d(z, y) dz + k_0(x, y) \\ & Q_1 = P_{11} - P_{12} - P_{21} - P_{22} \end{aligned}$$

where $k_0(x, y)$ is a polynomial and thus continuous. 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 $Z_d(x, z)_i = x^{\alpha_i} z^{\gamma_i}$ where $\alpha_i, \gamma_i \in \mathbb{N}^n$ for $i = 1, \cdots, q$. Then

$$\begin{aligned} & \prod_{i:\beta_i=0} \int_{z_i=x_i}^{b_i} \prod_{j:\beta_j=1} \int_{z_j=y_j}^{b_j} Z_d(z, x)^T Q_1 Z_d(z, y) dz \\ & \quad = \sum_{k,l} (Q_1)_{k,l} \prod_{i:\beta_i=0} \int_{z_i=x_i}^{b_i} \prod_{j:\beta_j=1} \int_{z_j=y_j}^{b_j} x^{\alpha_k} z^{\gamma_k} z^{\gamma_l} y^{\alpha_l} dz \\ & \quad = \sum_{k,l} (Q_1)_{k,l} x^{\alpha_k} y^{\alpha_l} \prod_{i:\beta_i=0} \int_{z_i=x_i}^{b_i} \prod_{j:\beta_j=1} \int_{z_j=y_j}^{b_j} z^{\gamma_k + \gamma_l} dz. \end{aligned} \tag{10}$$

Expanding the integrals in (10), each has the form

$$\begin{aligned} & \prod_{i:\beta_i=0} \int_{z_i=x_i}^{b_i} \prod_{j:\beta_j=1} \int_{z_j=y_j}^{b_j} z^{\alpha} dz \\ & = \prod_{i:\beta_i=0} \int_{z_i=x_i}^{b_i} z_i^{\alpha_i} dz_i \prod_{j:\beta_j=1} \int_{z_j=y_j}^{b_j} z_j^{\alpha_j} dz_j \\ & = \prod_{i:\beta_i=0} \frac{1}{\alpha_i + 1} (b_i - x_i^{\alpha_i+1}) \prod_{j:\beta_j=1} \frac{1}{\alpha_j + 1} (b_j - y_j^{\alpha_j+1}) \\ & = \prod_{k=1}^{n} \frac{1}{\alpha_k + 1} \prod_{i:\beta_i=0} (b_i - x_i^{\alpha_i+1}) \prod_{j:\beta_j=1} (b_j - y_j^{\alpha_j+1}) \\ & = \prod_{j=1}^{n} \frac{b_j - (\frac{1}{2}(x_j + y_j + |x_j - y_j|))^{\alpha_j+1}}{\alpha_j + 1}. \end{aligned}$$

where we have used the fact that

$$\frac{1}{2}(x + y + |x - y|) = \begin{cases} x & x > y \\ y & y > x. \end{cases}$$

Therefore $k(x, y)$ is the product and summation of continuous functions and thus $k(x, y)$ can be defined by a single continuous function over every domain. We conclude that k and therefore the resulting classifiers are both continuous. ■

In addition to continuity, we show that any kernel of this form for $P > 0$ has the universal property. Where recall we state the definition of universal kernel in Definition 1 as can be found in, e.g. [12].

Recall that $\mathcal{H}$ can be characterized as the closure of $\text{span}\{k(y, \cdot) \,:\, y \in \mathcal{X}\}$. The following theorem shows that any tessellated kernel with $P > 0$ is necessarily universal.

Theorem 6: Suppose k is as defined in Eqn. (5) for some $P > 0$, $d \in \mathbb{N}$ and N as defined in Eqn. (6). Then k is universal for $X = \mathcal{X} = [a, b]$, $a < b \in \mathbb{R}^n$.

Proof:

Without loss of generality, we assume $X = \mathcal{X} = [0, 1]^n$. If $P > 0$, then there exist ϵ_i such that $P = P_0 + \epsilon P_1$ where $P_0 \geq 0$ and

$$P_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \otimes [e_1, 0, \ldots, 0]$$

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

$$k(x, y) = k_0(x, y) + \epsilon \underbrace{\prod_{i=1}^{n} \min\{x_i, y_i\}}_{k_1(x,y)},$$

where k_0 is a positive kernel. Since the hypothesis space satisfies the additive property [21] [3], if k_1 is a universal kernel, then k is a universal kernel.

Now, consider

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

which consists of all functions of the form

$$f(x) = \sum_j c_j \prod_{i=1}^{n} f_{ij}(x_i)$$

where

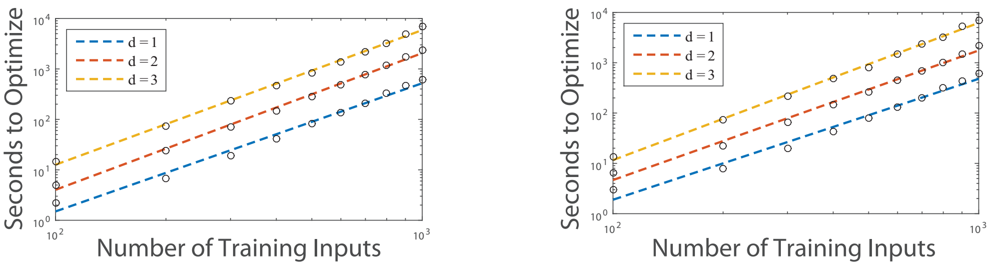
$$f_{ij}(x) = \min\{x, y_{ij}\} = \begin{cases} x, & \text{if } x \leq y_{ij} \\ y_{ij}, & \text{otherwise.} \end{cases}$$

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

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

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

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



(a) Complexity Scaling for Identification of Circle

(b) Complexity Scaling for Identification of Spiral

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

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 [18]. We conclude that k_1 is a universal kernel and hence k is universal.

■

Notation For convenience, we denote the positive Tessellated Kernels by saying $k \in \mathcal{K}_T^d$ if there exists some $P \geq 0$ such that k is as defined in Equation (5) where N is as defines in Eqn (6) using Z_d.

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 $\mathcal{K}_T$ contains only three parameters (elements of P).

V. SDP Formulation of the Kernel Learning Problem

Section II detailed general optimization methods by which we may search for an optimal kernel function, $k \in \mathcal{K}$, given that the set of kernel functions has a linear parameterization. We will now formulate specific methods for learning an optimal tessellated kernel function using either the SDP method of Optimization Problem (4), or using a two-step method like SimpleMKL. Using the representation of Tessellated Kernels ($\mathcal{K} = \mathcal{K}_T$) in Theorem 5 in Section IV, Optimization Problem (3) may be expressed as

$$\begin{aligned}
&\min_{t\in\mathbb{R},\gamma\in\mathbb{R},\boldsymbol{\nu}\in\mathbb{R}^m,\boldsymbol{\delta}\in\mathbb{R}^m} \quad t, \\
&\text{subject to: } \begin{pmatrix} G(P) & \mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\gamma\mathbf{y} \\ (\mathbf{e}+\boldsymbol{\nu}-\boldsymbol{\delta}+\gamma\mathbf{y})^T & t-\frac{2}{m\lambda}\boldsymbol{\delta}^T\mathbf{e} \end{pmatrix} \geq 0 \\
&\boldsymbol{\nu} \geq 0, \qquad \boldsymbol{\delta} \geq 0, \qquad P \geq 0, \qquad \text{trace}(P) \leq 1 \\
&G_{ij}(P) = \sum_{i,j=1,2} \sum_{k,l} (P_{i,j})_{k,l}\, x_i^{\alpha_k} x_j^{\alpha_l} \int_{\mathcal{X}_{ij}} z^{\gamma_k+\gamma_l} dz y_i y_j.
\end{aligned} \tag{11}$$

Optimization Problem (11) is an SDP and can therefore be solved efficiently using standard SDP solver such as [2]. Note that we use the trace constraint to ensure the kernel function is bounded.

Typically SDP problems require roughly p^2n^2 number of operations, where p is the number of decision variables and n is the dimension of the SDP constraint [6]. The number of decision variables in (11) is moderate, increasingly linearly in the number of training data points and the size of P. However, this optimization problem has a semi-definite matrix constraint whose dimension is linear in m. As we will see in Section VII, this limits the amount of training data which can be processed using Optimization Problem (11). To improve the scability of the algorithm, we therefore turn to variations on SimpleMKL.

VI. SimpleMKL Formulation of the Kernel Learning Problem

Recall that SimpleMKL searches for an optimal linear combination of kernel functions, that is it returns a vector of weights μ, on the a priori selected kernel functions. Here we discuss how SimpleMKL can be used to find optimal combinations of tessellated kernel functions that perform well in practice.

Since tessellated kernel functions have a linear parameterization, the positive sum of multiple tessellated kernel functions, parameterized by the positive semi-definite matrices P_i, is equivalent to a single tessellated kernel function, represented by the matrix $P = \sum_{i=1}^{k} P_i$.

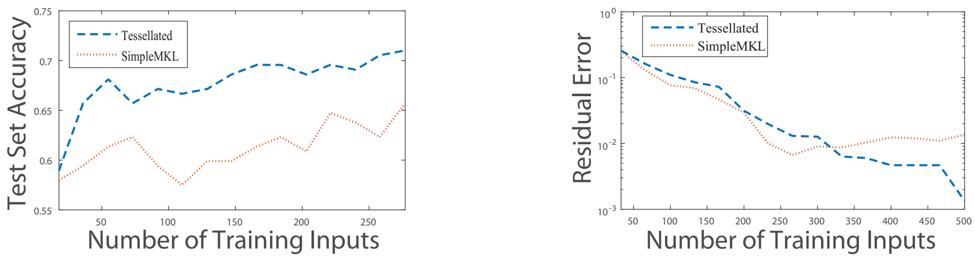
Therefore, by randomly generating a set of l positive semi-definite matrices, P_i for $i = 1, \ldots, l$, we may use SimpleMKL to find the optimal linear combination of tessellated kernels defined by each matrix P_i. The optimal tessellated kernel function may then be approximated as the tessellated kernel function parameterized by the matrix,

$$P = \sum_{i=1}^{k} \mu_i P_i.$$

Where μ is the vector of weights returned by SimpleMKL. In practice, we find that this randomized approach performs well in terms of accuracy on test data sets. Note that the complexity of SimpleMKL approximately increases linearly with the number of kernel functions, and superlinearly with respect to m, the number of data points [15].

Finally, we mention that we may avoid the heuristic use of randomized matrices by noting that SimpleMKL is a two-step method - where the second step fixes α and searched over μ_i. Since our parameterization of tessellated kernels is linear, this second step may be used to search over the entire space of tessellated kernels. However, implementation of this approach is left for future work.

We will next consider an experimental complexity analysis of the SDP method before comparing the accuracy of the two proposed methods.



(a) Average test set accuracy on the Liver dataset vs. the number of training data for the proposed method compared to SimpleMKL

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

Fig. 3: Plots demonstrating the change in accuracy of the tessellated kernel method 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.

VII. Implementation and complexity analysis

In this paper, we have proposed a new class of kernel functions defined by piecewise polynomials. In this section we analyze the complexity of Optimization Problem (11) with respect to the number of training points as well as the selected degree of the tessellated kernel function.

The constraint that the kernel be a positive tessellated kernel can be expressed as an LMI constraint with variables P_{ij}. Using Optimization Problem (11), 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. 2 and is similar to the complexity of other methods such as the hyperkernel approach in [13]. These scaling results are for training data randomly generated by two standard 2-feature example problems (circle and spiral - See Fig. 4) 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 high degree, the size of Z_d will become unmanageably large. Note, however, that, as indicated in the Section IV, even when $d = 0$, the kernels are universal.

VIII. Accuracy and comparison with existing methods

To evaluate 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

TABLE I: TSA comparison for algorithms a), b), c), d), and e). 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 the size of the dataset and n the number of features.

Data Set	Method	Accuracy	Time

Liver
m=346
n=6
	Tessellated	**72.32 ± 4.92**	95.75 ± 2.68

	SimpleMKL	65.51 ± 5.10	2.61 ± 0.42
	SimpleMKL Tess.	70.58 ± 4.69	8.37 ± 0.30
	Combined	70.53 ± 4.79	14.70 ± 0.76
	Neural Net	66.32 ± 7.46	0.14 ± 0.04

Cancer
m=684
n=9
	Tessellated	**97.18 ± 1.48**	636.17 ± 25.43

	SimpleMKL	96.55 ± 1.34	14.74 ± 1.33
	SimpleMKL Tess.	96.89 ± 1.43	45.84 ± 4.28
	Combined	96.89 ± 1.42	65.08 ± 10.52
	Neural Net	96.67 ± 1.30	0.18 ± 0.06

Heart
m=271
n=13
	Tessellated	83.46 ± 4.56	221.67 ± 29.63

	SimpleMKL	83.70 ± 4.77	3.09 ± 0.19
	SimpleMKL Tess.	**84.38 ± 4.34**	55.48 ± 2.67
	Combined	83.64 ± 4.54	13.23 ± 2.70
	Neural Net	78.64 ± 5.19	0.12 ± 0.01

Pima
m=769
n=8
	Tessellated	76.32 ± 3.10	1211.66 ± 27.01

	SimpleMKL	76.00 ± 3.33	19.04 ± 2.33
	SimpleMKL Tess.	**76.75 ± 2.81**	34.65 ± 23.28
	Combined	76.57 ± 2.72	96.20 ± 30.42
	Neural Net	75.35 ± 2.98	0.24 ± 0.19

Ionosphere
m=352
n=34
	Tessellated	**93.24 ± 3.04**	6.69 ± 0.27

	SimpleMKL	92.16 ± 2.78	26.24 ± 2.78
	SimpleMKL Tess.	87.65 ± 2.88	8.28 ± .16
	Combined	92.16 ± 2.78	50.77 ± 2.98
	Neural Net	90.85 ± 3.42	0.16 ± 0.02

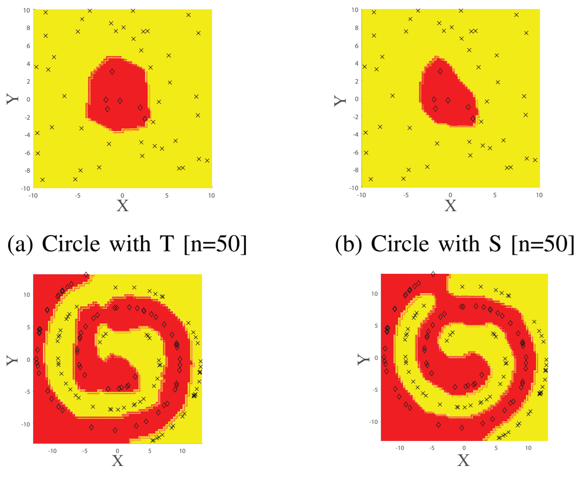
[Tessellated] We use the SDP algorithm in (11) using $d = 1$ (Except Ionosphere, which uses $d = 0$); To determine the integral in (11), we first scaled the data so that $x_i \in [0, 1]^n$, and then set $\mathcal{X} := [0 - \epsilon, 1 + \epsilon]^n$, where $\epsilon > 0$ was chosen by 5-fold cross-validation.

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

[Combined] We combined the libraries in [SimpleMKL] and [SimpleMKL Tess.] into a single SimpleMKL implementation;

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



(a) Circle with T [n=50]

(b) Circle with S [n=50]

(c) Spiral with T [n=150]

(d) Spiral with S [n=150]

Fig. 4: Discriminant Surface for Circle and Spiral Separator using Tessellated kernel [T] as Compared with SimpleMKL [S] for n training data.

In Table I, 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, the 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 tessellated kernel class when the ratio of training data to number of features is high. For this analysis, we use the liver data set (6 features) and the spiral discriminant [11] with 2 features (x and y) (we also briefly examine the unit circle). For the liver data set, in Figure 3, 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 the tessellated class over standard usage of SimpleMKL. For the spiral case, in Figure 3 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 the tessellated kernel showing significant improvement over SimpleMKL only for very large training data sets.

Finally, as illustration, we plotted the discriminant surface for both the spiral and unit circle data sets using both the Tessellated kernel and SimpleMKL using 150 training data points. These 2D surfaces are found in Figure 4.

IX. Conclusion

In this paper, we have proposed a new class of universal kernel functions that can be parameterized directly using positive matrices. Furthermore, any element of this class is universal, yielding comparable performance to and properties of the Gaussian kernels. However, unlike the Gaussian, the tessellated kernel does not require a set of bandwidths to be chosen a priori. Indeed, by increasing the degree of the monomial basis, it may be possible to show that the tessellated kernels can approximate any universal kernel arbitrarily well.

We have demonstrated the effectiveness of the tessellated class of kernel on several datasets 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 tessellated class can be readily integrated with existing multiple kernel learning algorithms such as Simple MKL - yielding similar results with less computational complexity. In most cases, either the optimal tessellated kernel, or the MKL learned suboptimal tessellated kernel will out perform 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 [4].

References

[1] 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.

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

[3] 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.

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

[5] C. Cortes, M. Mohri, and A. Rostamizadeh. Learning non-linear combinations of kernels. In *Advances in neural information processing systems*, 2009.

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

[7] I. Gohberg, S. Goldberg, and M. Kaashoek. *Classes of linear operators*. Birkhäuser, 2013.

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

[9] 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.

[10] 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.

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

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

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

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

[15] A. Rakotomamonjy, F.R. Bach, S. Canu, and Y. Grandvalet. Simplemkl. *Journal of Machine Learning Research*, 2008.

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

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

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

[19] SĆ 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.

[20] H Sun. Mercer theorem for rkhs on noncompact sets. *Journal of Complexity*, 2005.

[21] H. Wang, Q. Xiao, and D. Zhou. An approximation theory approach to learning with 1 regularization. *Journal of Approximation Theory*, 2013.

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