PARSIMONIOUS ONLINE LEARNING WITH KERNELS VIA SPARSE PROJECTIONS IN FUNCTION SPACE

Alec Koppel[⋆], *Garrett Warnell*[†], *Ethan Stump*[†], *and Alejandro Ribeiro*[⋆]

[⋆]Department of Electrical and Systems Engineering, University of Pennsylvania
[†]Computational and Information Sciences Directorate, U.S. Army Research Laboratory

ABSTRACT

We consider stochastic nonparametric regression problems in a reproducing kernel Hilbert space (RKHS), an extension of expected risk minimization to nonlinear function estimation. Popular perception is that kernel methods are inapplicable to online settings, since the generalization of stochastic methods to kernelized function spaces require memory storage that is cubic in the iteration index ("the curse of kernelization"). We alleviate this intractability in two ways: (1) we consider the use of functional stochastic gradient method (FSGD) which operates on a subset of training examples at each step; and (2), we extract parsimonious approximations of the resulting stochastic sequence via a greedy sparse subspace projection scheme based on kernel orthogonal matching pursuit (KOMP). We establish that this method converges almost surely in both diminishing and constant algorithm step-size regimes for a specific selection of sparse approximation budget. The method is evaluated on a kernel multi-class support vector machine problem, where data samples are generated from class-dependent Gaussian mixture models.

1. INTRODUCTION

We propose solving expected risk minimization problems, where the goal is to learn a regressor that minimizes a loss function quantifying the merit of a statistical model averaged over a data set. We focus on instances where the number of training examples N is either very large, or training examples arrive sequentially. This setting amounts to assuming a given training example with its associated target variable $(\mathbf{x}_n, \mathbf{y}_n)$ are independent realizations from a stationary joint distribution of the random pair $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$. Further, we focus on the case where our regressor is *not* a vector-valued parameter, but instead is a *function* $f \in \mathcal{H}$ in a function class $\mathcal{H}$. Function estimation allows for learning nonlinear statistical models, and has yielded promising results in applications where linearity of a given statistical model is overly restrictive, e.g., computer vision, object recognition, [1–5], and text processing [6].

This amounts to the minimization of a convex functional $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ which quantifies the merit of the estimator $f(\mathbf{x})$ evaluated at feature vector $\mathbf{x}$ averaged over all possible training examples plus a Tikhonov regularizer $\|f\|_{\mathcal{H}}$ to enforce stability [7, 8], stated as

$$f^* = \underset{f \in \mathcal{H}}{\operatorname{argmin}}\, R(f) := \underset{f \in \mathcal{H}}{\operatorname{argmin}}\, \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), y)] + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \qquad (1)$$

where we define the average loss as $L(f) := \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(f(\mathbf{x}), y)]$. In general, this problem may be intractable, but for the case that $\mathcal{H}$ is equipped with a *reproducing kernel* $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, a nonparametric function estimation problem of the form (1) may be reduced to a parametric form via the well-known Representer Theorem [9, 10]. Function spaces of this type are called Reproducing Kernel Hilbert Spaces (RKHS) [11].

Online kernel learning methods extend techniques from unkernelized (vector-valued) stochastic optimization in terms of optimizing (1) by replacing the objective's descent direction with a stochastic estimate [12]. However, several issues are unique to the kernelized setting: (i) the implementation of stochastic methods for kernel expected risk minimization require storage of kernel matrices and weight vectors whose size is comparable to the iteration index [13]; (ii) the design of kernel sparsification techniques must not break descent properties of the optimization sequences to which they are applied; and (iii) sparse approximation schemes frequently require conditions on input parameters that do not hold for kernel matrices induced by arbitrary data streams [14,15].

Issue (i), the *curse of kernelization*, is a key point of departure between the kernel stochastic optimization setting [cf. (1)] and its vector-valued counterpart, and comes from the fact that functions $f \in \mathcal{H}$ are coupled to the random variable $\mathbf{x}$, a consequence of the Representer Theorem [9]: f is a linear combination of kernel evaluations of elements of the training set of size N. As the sample size grows, i.e., $N \to \infty$, f requires infinite realizations of $\mathbf{x}$ to be represented in the Hilbert space [10]. Works on stochastic optimization in RKHS have either ignored issue (i) [16–19], or have augmented the learned function to reduce the storage issues associated with kernelization through sparsification.

These sparsification schemes are either designed in terms of the underlying optimization problem (supervised) or not (unsupervised). Unsupervised methods ignore issue (ii) and only focus on limiting the kernel dictionary growth [13, 20–23]. Past works that have considered *supervised sparsification* (addressing issues (i)-(ii)) have only been proposed for special cases [24–26]. [24] propose fixing the number of kernel dictionary elements, or *model order*, rather than determining which kernel dictionary elements are essential to represent f^*.

This is the first attempt to solve general kernelized expected risk minimization [cf. (1)] with supervised sparsification (Section 2). To do so, we make use of functional stochastic gradient descent (Section 3.1) together with a projection step onto a subspace of the Hilbert space spanned by a small number of kernel dictionary elements. We construct these instantaneous sparse subspaces by making use of kernel orthogonal matching pursuit (Section 3.2), a greedy search routine which, given a function and an approximation budget ϵ, returns a sparse approximation and guarantees its output to be in a specific error neighborhood of its input [27,28]. This approach mitigates issue (iii) by not requiring any stipulations on the kernel matrices induced by arbitrary data streams, and emphasizing error bounds over exact recovery. We show that the resulting sequence converges almost surely to the optimum of (1) under both attenuating and constant learning rate schemes (Section 4) – see [29] for proofs. In Section 5 we present numerical results for a multi-class kernel support vector machine problem with Gaussian mixture model data.

2. LEARNING IN REPRODUCING KERNEL HILBERT SPACE

In the case of supervised kernel learning [3], the function class $\mathcal{H}$ is taken to be a Hilbert space, whose elements are *functions*, $f : \mathcal{X} \to \mathcal{Y}$, that admit a representation in terms of elements of $\mathcal{X}$ when $\mathcal{H}$ has a special structure. In particular, equip $\mathcal{H}$ with a unique *kernel function*, $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, such that:

$$(i)\ \langle f, \kappa(\mathbf{x}, \cdot)\rangle_{\mathcal{H}} = f(\mathbf{x}),\ (ii)\ \mathcal{H} = \overline{\operatorname{span}\{\kappa(\mathbf{x}, \cdot)\}} \text{ for all } \mathbf{x} \in \mathcal{X}. \quad (2)$$

where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the Hilbert inner product for $\mathcal{H}$. Further assume the kernel is positive semidefinite, i.e., $\kappa(\mathbf{x}, \mathbf{x}') \geq 0$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$.

For kernelized regularized empirical risk minimization, the Representer Theorem [30, 31] states that the optimal f in the function class $\mathcal{H}$ may be written in terms of kernel evaluations *only* of the training set:

Thanks to NSF CCF-0952867, ONR N00014-12-1-0997, and ASEE SMART.

$$f(\mathbf{x}) = \sum_{n=1}^{N} w_n \kappa(\mathbf{x}_n, \mathbf{x}) \ , \tag{3}$$

where $\mathbf{w} = [w_1, \cdots, w_N]^T \in \mathbb{R}^N$ denotes a set of weights. The upper summand index N in (3) is henceforth referred to as the model order. Common choices κ include the polynomial kernel and the radial basis (Gaussian) kernel, i.e., $\kappa(\mathbf{x}, \mathbf{x}') = \left(\mathbf{x}^T\mathbf{x}' + b\right)^c$ and $\kappa(\mathbf{x}, \mathbf{x}') = \exp\left\{-\frac{\|\mathbf{x}-\mathbf{x}'\|_2^2}{2\tilde{\sigma}^2}\right\}$, respectively, where $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. c denotes the order of the polynomial, and $\tilde{\sigma}^2$ denotes the bandwidth of the Gaussian kernel.

We may now formulate the kernel variant of the empirical risk minimization problem as the one that minimizes the loss functional $L : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ plus a complexity-reducing penalty. The loss functional L may be written as an average over instantaneous losses $\ell : \mathcal{H}\times\mathcal{X}\times\mathcal{Y} \to \mathbb{R}$, each of which penalizes the average deviation between $f(\mathbf{x}_n)$ and the associated output y_n over the training set $\mathcal{S} = \{\mathbf{x}_n, y_n\}_{n=1}^N$. We denote the regularized loss as $R : \mathcal{H} \to \mathbb{R}$, and consider the problem

$$f^* = \operatorname*{argmin}_{f\in\mathcal{H}} R(f;\mathcal{S}) = \operatorname*{argmin}_{f\in\mathcal{H}} \frac{1}{N}\sum_{n=1}^{N} \ell(f(\mathbf{x}_n), y_n) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \tag{4}$$

The above problem, kernelized Tikhonov regularization [8], is one in which we aim to learn a general nonlinear relationship between $\mathbf{x}_n$ and y_n through a function f. We assume ℓ is convex with respect to its first argument $f(\mathbf{x})$. By substituting the Representer Theorem expansion in (3) into (4), the problem amounts to finding a coefficient vector $\mathbf{w}$ as

$$f^* = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^N} \frac{1}{N}\sum_{n=1}^{N} \ell(\mathbf{w}^T\boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_n), y_n) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{K}_{\mathbf{X},\mathbf{X}}\mathbf{w} \tag{5}$$

where we define the Gram matrix (also called the *kernel matrix*) $\mathbf{K}_{\mathbf{X},\mathbf{X}} \in \mathbb{R}^{N\times N}$, with entries given by the kernel evaluations between $\mathbf{x}_m$ and $\mathbf{x}_n$ as $[\mathbf{K}_{\mathbf{X},\mathbf{X}}]_{m,n} = \kappa(\mathbf{x}_m, \mathbf{x}_n)$. Further define the vector of kernel evaluations $\boldsymbol{\kappa}_{\mathbf{X}}(\cdot) = [\kappa(\mathbf{x}_1, \cdot)\ldots\kappa(\mathbf{x}_N, \cdot)]^T$, which are related to the kernel matrix as $\mathbf{K}_{\mathbf{X},\mathbf{X}} = [\boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_1)\ldots\boldsymbol{\kappa}_{\mathbf{X}}(\mathbf{x}_N)]$. Lastly, define the dictionary associated with the kernel matrix as $\mathbf{X} = [\mathbf{x}_1, \ \ldots\ , \mathbf{x}_N]$.

Observe that by exploiting the Representer Theorem, we reduce a nonparametric infinite dimensional optimization problem in $\mathcal{H}$ (4) into a finite N-dimensional parametric problem (5). Thus, for empirical risk minimization, the RKHS provides a principled framework to solve nonparametric regression problems as a search for coefficients over $\mathbb{R}^N$.

2.1. Online Kernel Learning

The goal of this paper is to solve problems of the form (4) when training examples $(\mathbf{x}_n, \mathbf{y}_n)$ either become sequentially available or their total number is not necessarily finite. Hence consider the case where $(\mathbf{x}_n, \mathbf{y}_n)$ are independent realizations from a stationary joint distribution of the random pair $(\mathbf{x}, \mathbf{y}) \in \mathcal{X}\times\mathcal{Y}$. In this case, the objective in (4) may be written as an expectation over this random pair as

$$f^* = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^{|\mathcal{I}|}, \{\mathbf{x}_n\}_{n\in\mathcal{I}}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ell(\sum_{n\in\mathcal{I}} w_n\kappa(\mathbf{x}_n,\mathbf{x}), y)] + \frac{\lambda}{2}\|\sum_{n,m\in\mathcal{I}} w_n w_m\kappa(\mathbf{x}_m,\mathbf{x}_n)\|_{\mathcal{H}}^2. \tag{6}$$

where we substitute the expansion of f given by the Representer Theorem [10] into (1), and define $\mathcal{I}$ as some countably infinite indexing set. Our goal is to solve (6) (equivalent to (1)) while ensuring parsimony in function representation by ensuring index set $\mathcal{I}$ has small cardinality.

3. ALGORITHM DEVELOPMENT

We turn to deriving an algorithmic solution to the kernelized expected risk minimization problem stated in (1) by first deriving functional stochastic gradient descent, making use of functional gradient (Frechét derivative) computations (Section 3.1). The resulting parametric updates require memory storage whose complexity is cubic in the iteration index (the curse of kernelization). We alleviate this memory explosion using sparse projections constructed with KOMP (Section 3.2).

3.1. Functional Stochastic Gradient Descent

Following [13], we present the generalization of the stochastic gradient method in RKHS. The result is *functional stochastic gradient descent* (FSGD). Given an independent realization $(\mathbf{x}_t, y_t)$ of the random pair $(\mathbf{x}, y)$, the stochastic functional gradient update associated with (1) is

$$f_{t+1} = (1-\eta_t\lambda)f_t - \eta_t\ell'(f(\mathbf{x}_t), y_t)\kappa(\mathbf{x}_t, \cdot) \ , \tag{7}$$

where $\eta_t > 0$ is learning rate (diminishing as $\mathcal{O}(1/t)$ or a small constant – Section 4), and $\nabla_f\ell(f(\mathbf{x}_t), y_t)(\cdot) = \ell'(f(\mathbf{x}_t), y_t)\kappa(\mathbf{x}_t, \cdot) \in \mathcal{H}$ is the functional gradient of the instantaneous loss. We denote $\ell'(f(\mathbf{x}_t), y_t) := \partial\ell(f(\mathbf{x}_t), y_t)/\partial f(\mathbf{x}_t)$ as the derivative of $\ell(\mathbf{f}(\mathbf{x}_t), y_t)$ with respect to its scalar argument $f(\mathbf{x}_t)$ evaluated at $\mathbf{x}_t$. Derivation details are given in Section 3.1 of [29]. With $\lambda > 0$, step-size $\eta_t < 1/\lambda$, and initialization $f_0 = 0 \in \mathcal{H}$, the Representer Theorem (3) allows us to write f_t as an expansion over past feature vectors $\mathbf{x}_t$

$$f_t(\mathbf{x}) = \sum_{n=1}^{t-1} w_n\kappa(\mathbf{x}_n, \mathbf{x}) = \mathbf{w}_t^T\boldsymbol{\kappa}_{\mathbf{X}_t}(\mathbf{x}) \ . \tag{8}$$

On the right-hand side of (8) we have introduced the notation $\mathbf{X}_t = [\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}] \in \mathbb{R}^{p\times(t-1)}$ and $\boldsymbol{\kappa}_{\mathbf{X}_t}(\cdot) = [\kappa(\mathbf{x}_1, \cdot), \ \ldots\ , \kappa(\mathbf{x}_{t-1}, \cdot)]^T$. The kernel expansion in (8), taken together with the functional update (7), makes explicit that performing the stochastic gradient method in $\mathcal{H}$ amounts to updates on the kernel dictionary $\mathbf{X}$ and coefficient vector $\mathbf{w}$:

$$\mathbf{X}_{t+1} = [\mathbf{X}_t, \ \ \mathbf{x}_t], \ \mathbf{w}_{t+1} = [(1-\eta_t\lambda)\mathbf{w}_t, -\eta_t\ell'(f_t(\mathbf{x}_t), y_t)], \tag{9}$$

This update causes $\mathbf{X}_{t+1}$ to have one more column than $\mathbf{X}_t$. We define the *model order* as number of data points M_t in the dictionary at time t (the number of columns of $\mathbf{X}_t$). SGD is such that $M_t = t-1$, and hence grows unbounded with iteration index t.

3.2. Model Order Control via Stochastic Projection

We propose replacing the update (7), which may be considered a projection onto subspace $\mathcal{H}_{\mathbf{X}_{t+1}} = \text{span}\{\kappa(\mathbf{x}_n, \cdot)\}_{n=1}^{t-1}$, by the stochastic projection of the functional stochastic gradient sequence onto the subspace $\mathcal{H}_{\mathbf{D}_{t+1}} = \text{span}\{\kappa(\mathbf{d}_n, \cdot)\}_{n=1}^{M_t}$ as

$$\begin{aligned} f_{t+1} &= \operatorname*{argmin}_{f\in\mathcal{H}_{\mathbf{D}_{t+1}}} \Big\|f - \Big((1-\eta_t\lambda)f_t - \eta_t\nabla_f\ell(f_t(\mathbf{x}_t), y_t)\Big)\Big\|_{\mathcal{H}}^2 \\ &:= \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}}\Big[(1-\eta_t\lambda)f_t - \eta_t\nabla_f\ell(f_t(\mathbf{x}_t), y_t)\Big]. \end{aligned} \tag{10}$$

where we define the projection $\mathcal{P}$ onto subspace $\mathcal{H}_{\mathbf{D}_{t+1}} \subset \mathcal{H}$ by (10).

Coefficient update The update (10), for a fixed dictionary $\mathbf{D}_{t+1} \in \mathbb{R}^{p\times M_{t+1}}$, may be expressed in terms of a coefficient update only. To do so, define the stochastic gradient update without projection $\tilde{f}_t$, given function f_t parameterized by dictionary $\mathbf{D}_t$ and coefficients $\mathbf{w}_t$,

$$\tilde{f}_{t+1} = (1-\eta_t\lambda)f_t - \eta_t\nabla_f\ell(f_t; \mathbf{x}_t, \mathbf{y}_t). \tag{11}$$

This update may be represented using dictionary and weight vector

$$\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t, \ \ \mathbf{x}_t], \ \tilde{\mathbf{w}}_{t+1} = [(1-\eta_t\lambda)\mathbf{w}_t, \ \ -\eta_t\ell'(f_t(\mathbf{x}_t), y_t)] \ . \tag{12}$$

Then for a fixed dictionary $\mathbf{D}_{t+1}$, the stochastic projection in (10) amounts to a least-squares problem on the coefficient vector (derived via use of the Representer Theorem in [29])

$$\mathbf{w}_{t+1} = \mathbf{K}_{\mathbf{D}_{t+1}\mathbf{D}_{t+1}}[\mathbf{K}_{\mathbf{D}_{t+1}\tilde{\mathbf{D}}_{t+1}}]^{\dagger}\tilde{\mathbf{w}}_{t+1} \ , \tag{13}$$

where $\dagger$ is used to denote the pseudoinverse. Given that the projection of $\tilde{f}_{t+1}$ onto the stochastic subspace $\mathcal{H}_{\mathbf{D}_{t+1}}$, for a fixed dictionary $\mathbf{D}_{t+1}$, is a least-squares projection, we now detail how the kernel dictionary $\mathbf{D}_{t+1}$ is selected from the data sample path $\{\mathbf{x}_u, y_u\}_{u\leq t}$.

Dictionary Update The selection procedure for dictionary $\mathbf{D}_{t+1}$ is based upon greedy sparse approximation [32]. The function $\tilde{f}_{t+1} = (1-\eta_t)f_t - \eta_t\nabla_f\ell(f_t; \mathbf{x}_t, \mathbf{y}_t)$ defined by FSGD without projection is parameterized by dictionary $\tilde{\mathbf{D}}_{t+1}$ [cf. (12)] of model order $\tilde{M} = M_t +$

Algorithm 1 Destructive Kernel Orthogonal Matching Pursuit (KOMP)

```
Require: function $\tilde f$ defined by dict. $\tilde{\mathbf{D}} \in \mathbb{R}^{p\times \tilde M}$, coeffs. $\tilde{\mathbf{w}} \in \mathbb{R}^{\tilde M}$, approx. budget $\epsilon_t > 0$
  initialize $f = \tilde f$, dictionary $\mathbf{D} = \tilde{\mathbf{D}}$ with indices $\mathcal{I}$, model order $M = \tilde M$, coeffs. $\mathbf{w} = \tilde{\mathbf{w}}$.
  while candidate dictionary is non-empty $\mathcal{I} \neq \emptyset$ do
    for $j = 1, \dots, \tilde M$ do
      Find minimal approx. error with dict. element $\mathbf{d}_j$ is removed
        $\gamma_j = \min_{\mathbf{w}_{\mathcal{I}\setminus\{j\}} \in \mathbb{R}^{M-1}} \|\tilde f(\cdot) - \sum_{k\in\mathcal{I}\setminus\{j\}} w_k \kappa(\mathbf{d}_k, \cdot)\|_{\mathcal{H}}$ .
    end for
    Find dict. index minimizing approx. error: $j^* = \mathrm{argmin}_{j\in\mathcal{I}} \gamma_j$
      if minimal approximation error exceeds threshold $\gamma_{j^*} > \epsilon_t$
        stop
      else
        Prune dict. $\mathbf{D} \leftarrow \mathbf{D}_{\mathcal{I}\setminus\{j^*\}}$
        Revise set $\mathcal{I} \leftarrow \mathcal{I} \setminus \{j^*\}$ and model order $M \leftarrow M - 1$.
        Compute updated weights $\mathbf{w}$ defined by current dictionary $\mathbf{D}$
          $\mathbf{w} = \mathrm{argmin}_{\mathbf{w}\in\mathbb{R}^M} \|\tilde f(\cdot) - \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}}$
      end
  end while
  return $f, \mathbf{D}, \mathbf{w}$ of model order $M \le \tilde M$ such that $\|f - \tilde f\|_{\mathcal{H}} \le \epsilon_t$
```

1. We form $\mathbf{D}_{t+1}$ by selecting a subset of $M_{t+1} \le M_t + 1$ columns from $\tilde{\mathbf{D}}_{t+1}$ that approximates $\tilde f_{t+1}$ well in terms of Hilbert-norm error. We make use of *kernel orthogonal matching pursuit* (KOMP) [28] with error tolerance ϵ_t to find a kernel dictionary matrix $\mathbf{D}_{t+1}$ based on the one which adds the latest sample point $\tilde{\mathbf{D}}_{t+1}$. This choice is due to the fact that we can tune its stopping criterion to stochastic descent in contrast to other compressive techniques – see Section 4. We propose using a destructive variant of KOMP with pre-fitting (detailed in Algorithm 1).

With Algorithm 1 stated, we summarize the proposed method in Algorithm 2 for solving (1). The method, Parsimonious Online Learning with Kernels (POLK), executes the stochastic projection of the functional stochastic gradient iterates onto subspaces $\mathcal{H}_{\mathbf{D}_{t+1}}$ [cf. (10)]. The initial function is set to null $f_0 = 0$ (empty kernel dictionary $\mathbf{D}_0 = []$ and coefficient vector $\mathbf{w}_0 = []$). At each step, given training example $(\mathbf{x}_t, y_t)$ and step-size η_t, we compute the *unconstrained* FSGD iterate $\tilde f_{t+1}(\cdot) = (1 - \eta_t\lambda) f_t - \eta_t \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot)$, represented by $\tilde{\mathbf{D}}_{t+1}$ and $\tilde{\mathbf{w}}_{t+1}$ as stated in (12). These parameters are then fed into KOMP with budget ϵ_t, such that $(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \text{KOMP}(\tilde f_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$.

4. CONVERGENCE ANALYSIS

We turn to studying the theoretical performance of Algorithm 2 developed in Section 3. In particular, we establish that the method, when a diminishing step-size is chosen, is guaranteed to converge to the optimum of (1). We further obtain that when a sufficiently small constant step-size is chosen, the limit infimum of the iterate sequence is within a neighborhood of the optimum. In both cases, convergence depends on the approximation budget used in sparsification (Algorithm 1).

First, note an important fact regarding the stochastic gradient and define its modification induced by sparsification. The stochastic functional gradient defined by the update (7), i.e., $\hat\nabla_f \ell(f_t(\mathbf{x}_t), y_t) = \nabla_f \ell(f_t(\mathbf{x}_t), y_t) + \lambda f_t$ is an unbiased estimator of the true functional gradient of $R(f)$ in (1), i.e.

$$\mathbb{E}[\hat\nabla_f \ell(f_t(\mathbf{x}_t), y_t) \,\big|\, \mathcal{F}_t] = \nabla_f R(f_t) \tag{14}$$

Moreover, define the projected stochastic gradient associated with the sparsified stochastic descent direction in (10) as

Algorithm 2 Parsimonious Online Learning with Kernels (POLK)

```
Require: $\{\mathbf{x}_t, \mathbf{y}_t, \eta_t, \epsilon_t\}_{t=0,1,2,\dots}$
  initialize $f_0(\cdot) = 0, \mathbf{D}_0 = [], \mathbf{w}_0 = []$, empty dict. and coeff.
  for $t = 0, 1, 2, \dots$ do
    Receive independent training pair $(\mathbf{x}_t, y_t)$
    Compute functional stochastic gradient step [cf. (11)]
      $\tilde f_{t+1}(\cdot) = (1 - \eta_t\lambda) f_t - \eta_t \ell'(f_t(\mathbf{x}_t), \mathbf{y}_t)\kappa(\mathbf{x}_t, \cdot)$
    Revise dict. $\tilde{\mathbf{D}}_{t+1} = [\mathbf{D}_t, \mathbf{x}_t]$, weights $\tilde{\mathbf{w}}_{t+1} \leftarrow [(1 - \eta_t\lambda)\mathbf{w}_t,\ -\eta_t \ell'(f_t(\mathbf{x}_t), y_t)]$
    Compute sparse function approximation via Algorithm 1
      $(f_{t+1}, \mathbf{D}_{t+1}, \mathbf{w}_{t+1}) = \textbf{KOMP}(\tilde f_{t+1}, \tilde{\mathbf{D}}_{t+1}, \tilde{\mathbf{w}}_{t+1}, \epsilon_t)$
  end for
```

$$\tilde\nabla_f \ell(f_t(\mathbf{x}_t), y_t) = \Big(f_t - \mathcal{P}_{\mathcal{H}_{\mathbf{D}_{t+1}}}\Big[f_t - \eta_t \hat\nabla_f \ell(f_t(\mathbf{x}_t), y_t)\Big]\Big)/\eta_t \tag{15}$$

With these facts noted, we may now clarify the technical setting.

(A1) The feature space $\mathcal{X} \subset \mathbb{R}^p$ and target domain $\mathcal{Y} \subset \mathbb{R}$ are compact, and the reproducing kernel map may be bounded as

$$\sup_{\mathbf{x}\in\mathcal{X}} \sqrt{\kappa(\mathbf{x}, \mathbf{x})} = X < \infty \tag{16}$$

(A2) The instantaneous loss $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is uniformly C-Lipschitz continuous for all $z \in \mathbb{R}$, fixed $y \in \mathcal{Y}$

$$|\ell(z, y) - \ell(z', y)| \le C|z - z'| \tag{17}$$

(A3) The loss function $\ell(f(\mathbf{x}), y)$ is convex and differentiable with respect to its first (scalar) argument $f(\mathbf{x})$ on $\mathbb{R}$ for all $\mathbf{x} \in \mathcal{X}, y \in \mathcal{Y}$.

(A4) Let $\mathcal{F}_t$ denote the sigma algebra which measures the algorithm history for times $u \le t$, i.e. $\mathcal{F}_t = \{\mathbf{x}_u, y_u, u_u\}_{u=1}^t$. The projected functional gradient of the regularized instantaneous risk has finite conditional second moments for each t, that is,

$$\mathbb{E}[\|\tilde\nabla_f \ell(f_t(\mathbf{x}_t), y_t)\|_{\mathcal{H}}^2 \mid \mathcal{F}_t] \le \sigma^2 \tag{18}$$

(A1) is satisfied in most applications by the data domain, and justifies bounding the loss in (A2). These conditions permit bounding the optimal f^* in Hilbert norm. Variants of (A2) appear in kernelized stochastic methods [16, 18]. (A3) is satisfied for most problems. (A4) ensures that the variance of the stochastic approximation error is finite.

We next establish that under a diminishing algorithm step-size scheme, with the sparse approximation budget selection

$$\sum_{t=1}^{\infty} \eta_t = \infty\ , \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty\ , \quad \epsilon_t = \eta_t^2\ , \tag{19}$$

Algorithm 2 converges almost surely exactly to the optimal f^* [cf. (1)].

Theorem 1 *The sequence generated $\{f_t\}$ by Algorithm 2 with $f_0 = 0$, f^* the minimizer of the regularized expected risk stated in* (1)*, under (A1)-(A4), diminishing step-size and approximation budget [cf.* (19)*], and regularizer such that $\eta_t < 1/\lambda$ for all t, yields an objective error sequence converging to null in infimum almost surely as*

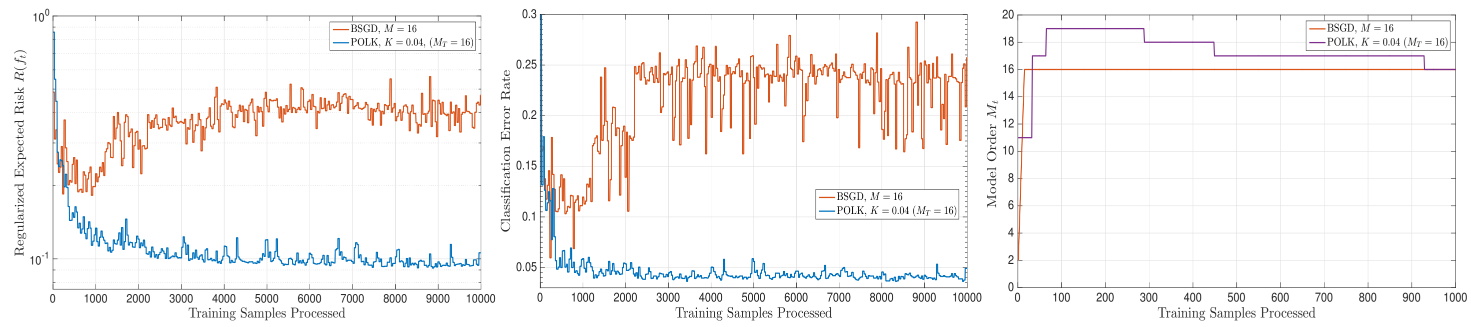
$$\liminf_{t\to\infty} R(f_t) - R(f^*) = 0 \qquad \textit{a.s.} \tag{20}$$

Moreover, the function sequences $\{f_t\}$ converges almost surely as

$$\lim_{t\to\infty} \|f_t - f^*\|_{\mathcal{H}}^2 = 0 \qquad \textit{a.s.} \tag{21}$$

Theorem 1 states that when a diminishing step-size is chosen as, e.g. $\eta_t = \mathcal{O}(1/t)$, with approximation budget selected as $\epsilon_t = \eta_t^2$, we obtain exact convergence to the optimizer of (1) is attained. However, in obtaining exact convergence, the approximation budget approaches null $\epsilon_t = \mathcal{O}(1/t^2)$, which means that the model order may grow arbitrarily.

Instead, consider a constant algorithm step-size $\eta_t = \eta$ with approximation budget chosen as a constant which satisfies $\epsilon_t = \epsilon = \mathcal{O}(\eta^{3/2})$. Then we obtain convergence in infimum to the optimal neighborhood.



(a) Objective $R(f_t)$ vs. no. samples processed

(b) Error rate vs. no. samples processed

(c) Model order M_t vs. no. samples processed

Fig. 1: POLK and BSGD for a large-scale kernel SVM multi-classification problem. Observe that POLK outperforms the competing method by an order of magnitude in terms of objective evaluation (Fig. 1a) and misclassification rate (Fig. 1b) for the same model order (Fig. 1c). POLK *learns* the appropriate model order $M_T = 16$ defined by the data domain (15 total modes of joint data density), suggesting attainment of the optimal f^*.

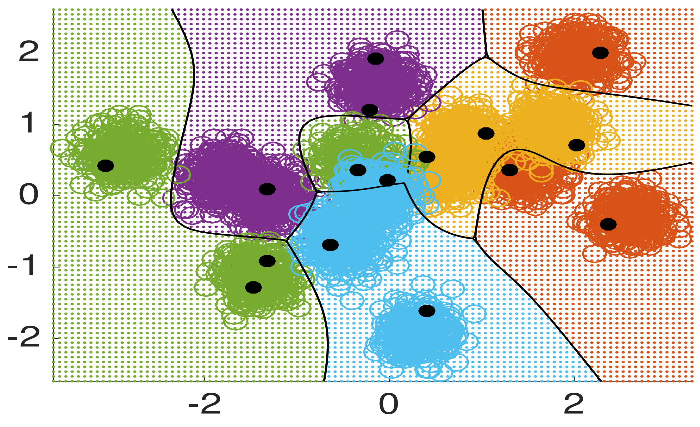


Fig. 2: Planar decision surface of final POLK classifier. Training examples from distinct classes are assigned a unique color. Grid colors represent the classification decision by f_t. Bold black dots are kernel dictionary elements, which concentrate at the modes of the joint data distribution. Curved lines are drawn to denote class label boundaries.

Theorem 2 *The sequence $\{f_t\}$ generated by Algorithm 2 with $f_0 = 0$, with f^* defined by* (1)*, under (A1)-(A4), with regularizer $\lambda > 0$, constant algorithm step-size $\eta_t = \eta$ chosen such that $\eta < 1/\lambda$, and sparse approximation budget satisfying $\epsilon = K\eta^{3/2} = \mathcal{O}(\eta^{3/2})$, with K a positive scaler, converges to a neighborhood almost surely as*

$$\liminf_{t\to\infty} \|f_t - f^*\|_{\mathcal{H}} \leq \frac{\sqrt{\eta}}{\lambda}\left(K + \sqrt{K^2 + \lambda\sigma^2}\right) = \mathcal{O}(\sqrt{\eta}) \ \ a.s. \quad (22)$$

Theorem 2 states that when a small constant step-size is used together with a bias tolerance induced by sparsification chosen as $\epsilon = \mathcal{O}(\eta^{3/2})$, Algorithm 2 converges in infimum to a neighborhood of the optimum which depends on the chosen step-size, the Lipschitz constant C of the instantaneous loss, the regularization parameter λ, as well as the variance of the stochastic gradient σ^2. This result is typical of stochastic methods – the infimum is a consequence of the bias induced by sparsification. However, use of a constant learning rate allows us to guarantee the model order of Algorithm 2 is always bounded ([29], Section 4.2).

Remark 1 *(Sparsity of f^*)* Algorithm 2 provides a method to avoid keeping an unnecessarily large number of kernel dictionary elements along the convergence path towards f^* [cf. (1)], solving the classic scalability problem of kernel methods in stochastic programming. However, if the optimal function admits a low dimensional representation $|\mathcal{I}| << \infty$, then in addition to extracting memory efficient instantaneous iterates, POLK will obtain the optimal function exactly. In Section 5, we illustrate this property via a multi-class classification problem.

5. NUMERICAL ANALYSIS

In this section, we evaluate the performance of Algorithm 2 on a multi-class support vector machine (SVM) classification problem with training examples drawn from distinct Gaussian mixture models for each class. The data generation, comparable to [26], consists of feature-label pairs $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$ drawn from a single distribution. Each $y_n \in \{1, \ldots, C\}$ denotes a class label, and is drawn uniformly at random from the label set. Then, each feature vector $\mathbf{x} \in \mathbb{R}^p$ is drawn from a planar ($p = 2$) equitably weighted Gaussian mixture model, i.e., $\mathbf{x}\,|\,y \sim (1/3)\sum_{j=1}^3 \mathcal{N}(\boldsymbol{\mu}_{y,j}, \sigma_{y,j}^2\mathbf{I})$ with unit variances $\sigma_{y,j}^2 = 1$ and class-dependent mean $\boldsymbol{\mu}_{y,j}$. These means are realizations of a Gaussian distribution with class-dependent parameters, i.e., $\boldsymbol{\mu}_y \sim \mathcal{N}(\boldsymbol{\theta}_y, \sigma_y^2\mathbf{I})$. Here $\boldsymbol{\theta}_y$ are equitably spaced around the unit circle, one for each class label. We fix the number of classes $C = 5$, meaning that the joint density of the data consists of 15 distinct Gaussians. We generate 7500 feature-label pairs: $N = 5000$ for training; 2500 for testing.

We formulate kernel multi-class SVM by defining for each class c a class-specific function $f_c : \mathcal{X} \to \mathbb{R}$. Together these functions classify feature vector $\mathbf{x}$ by maximizing the class-conditional probability $y = \max_{y'} f_{y'}(\mathbf{x})$. Given N labeled training examples, we train the KSVM by finding the set of functions $\mathbf{f}^* = [f_1^*, \ldots, f_C^*] \in \mathcal{H}^C$ that minimize the λ-regularized expected risk via the multi-class hinge loss [24]

$$\mathbf{f}^* = \underset{\mathbf{f}}{\text{argmin}}\ \frac{1}{N}\sum_{n=1}^N \ell(\mathbf{f}(\mathbf{x}_n), y_n) + \lambda\sum_{c'=1}^C \|f_{c'}\|_{\mathcal{H}}^2\ , \quad (23)$$

with $\ell(\mathbf{f}(\mathbf{x}), y) = \max(0, 1+f_r(\mathbf{x})-f_y(\mathbf{x}))$, $r = \text{argmax}_{c'\neq y} f_{c'}(\mathbf{x})$.

To implement Algorithm 2 for the problem (23), several parameters must be specified. We use the Gaussian kernel with bandwidth $\tilde{\sigma}^2 = 0.6$, regularizer $\lambda = 10^{-6}$, constant learning rate $\eta = 6.0$, approximation budget $\epsilon = K\eta^{3/2}$, and parsimony constant [cf. (22)] $K = 0.04$. Further, we initialize the kernel classifier as null, i.e., $\mathbf{f}_0 = 0$. To compare the performance of Algorithm 2 with competitors, we consider another online kernel method with supervised sparsification (projection) developed only for KSVM, called Budgeted Stochastic Gradient Descent (BSGD) [24], for which we used the same $\tilde{\sigma}^2$ and λ, set the budget to $M = 16$ for comparison (since this is the model order our method extracts at steady state), and set $\eta = 1.0$ since it yields the best results for this instance. BSGD fixes the dictionary size, not the Hilbert-norm error.

In Figure 1 we plot the empirical results of this experiment, and observe that POLK outperforms the competing method by an order of magnitude in terms of objective evaluation (Fig. 1a) and test misclassification rate (Fig 1b). Moreover, for our setup, the optimal model order is $M^* = 15$, which is approximately *learned* by POLK $M_T = 16$ (Fig. 1c), whereas BSGD, initialized with this parameter, does not converge. The final decision surface $\mathbf{f}_T$ of POLK is shown in Fig. 2 – kernel dictionary elements concentrate near the modes of the joint data density.

6. REFERENCES

[1] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," in *the IEEE Trans. on Image Processing*. ITIP, 2007, pp. 53–69.

[2] S. Mukherjee and S. K. Nayar, "Automatic generation of rbf networks using wavelets," *Pattern Recognition*, vol. 29, no. 8, pp. 1369–1383, 1996.

[3] J.-B. Li, S.-C. Chu, and J.-S. Pan, *Kernel Learning Algorithms for Face Recognition*. Springer, 2014.

[4] M. Taşan, G. Musso, T. Hao, M. Vidal, C. A. MacRae, and F. P. Roth, "selecting causal genes from genome-wide association studies via functionally coherent subnetworks," *Nature methods*, 2014.

[5] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*, 4th ed. Academic Press, 2008.

[6] G. Sampson, R. Haigh, and E. Atwell, "Natural language analysis by stochastic optimization: A progress report on project april," *J. Exp. Theor. Artif. Intell.*, vol. 1, no. 4, pp. 271–287, Oct. 1990. [Online]. Available: http://dx.doi.org/10.1080/09528138908953710

[7] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan, "Learnability, stability and uniform convergence," *Journal of Machine Learning Research*, vol. 11, no. Oct, pp. 2635–2670, 2010.

[8] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," *Advances in computational mathematics*, vol. 13, no. 1, pp. 1–50, 2000.

[9] R. Wheeden and A. Zygmund, *Measure and Integral: An Introduction to Real Analysis*, ser. Chapman & Hall/CRC Pure and Applied Mathematics. Taylor & Francis, 1977. [Online]. Available: https://books.google.com/books?id=YDkDmQ_hdmcC

[10] V. Norkin and M. Keyzer, "On stochastic optimization and statistical learning in reproducing kernel hilbert spaces by support vector machines (svm)," *Informatica*, vol. 20, no. 2, pp. 273–292, 2009.

[11] K. Müller, T. Adali, K. Fukumizu, J. C. Principe, and S. Theodoridis, "Special issue on advances in kernel-based learning for signal processing [from the guest editors]," *IEEE Signal Process. Mag.*, vol. 30, no. 4, pp. 14–15, 2013. [Online]. Available: http://dx.doi.org/10.1109/MSP.2013.2253031

[12] H. Robbins and S. Monro, "A stochastic approximation method," *Ann. Math. Statist.*, vol. 22, no. 3, pp. 400–407, 09 1951.

[13] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online Learning with Kernels," *IEEE Transactions on Signal Processing*, vol. 52, pp. 2165–2176, August 2004.

[14] E. J. Candes, "The restricted isometry property and its implications for compressed sensing," *Comptes Rendus Mathematique*, vol. 346, no. 9, pp. 589–592, 2008.

[15] T. T. Cai and L. Wang, "Orthogonal matching pursuit for sparse signal recovery with noise," *Information Theory, IEEE Transactions on*, vol. 57, no. 7, pp. 4680–4688, 2011.

[16] Y. Ying and D. X. Zhou, "Online regularized classification algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 11, pp. 4775–4788, Nov 2006.

[17] W. Liu, P. P. Pokharel, and J. C. Principe, "The kernel least-mean-square algorithm," *Signal Processing, IEEE Transactions on*, vol. 56, no. 2, pp. 543–554, 2008.

[18] M. Pontil, Y. Ying, and D. xuan Zhou, "Error analysis for online gradient descent algorithms in reproducing kernel hilbert spaces," Tech. Rep., 2005.

[19] A. Dieuleveut and F. Bach, "Non-parametric stochastic approximation with large step sizes," *arXiv preprint arXiv:1408.0361*, 2014.

[20] L. Zhang, J. Yi, R. Jin, M. Lin, and X. He, "Online kernel learning with a near optimal sparsity bound." in *ICML (3)*, 2013, pp. 621–629.

[21] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2275–2285, Aug 2004.

[22] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, pp. 1058–1067, 2009.

[23] P. Honeine, "Online kernel principal component analysis: A reduced-order model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1814–1826, 2012.

[24] Z. Wang, K. Crammer, and S. Vucetic, "Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale svm training," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 3103–3131, 2012.

[25] T. Joachims and C.-N. J. Yu, "Sparse kernel svms via cutting-plane training," *Machine Learning*, vol. 76, no. 2-3, pp. 179–193, 2009.

[26] J. Zhu and T. Hastie, "Kernel Logistic Regression and the Import Vector Machine," *Journal of Computational and Graphical Statistics*, vol. 14, no. 1, pp. 185–205, 2005.

[27] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition," in *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, 1993.

[28] P. Vincent and Y. Bengio, "Kernel matching pursuit," *Machine Learning*, vol. 48, no. 1, pp. 165–187, 2002.

[29] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "Parsimonious online kernel learning via sparse projections in function space," *Journal of Machine Learning Research (in preparation)*, 2016. [Online]. Available: https://fling.seas.upenn.edu/~akoppel/assets/papers/kernel_report.pdf

[30] G. Kimeldorf and G. Wahba, "Some results on tchebycheffian spline functions," *Journal of mathematical analysis and applications*, vol. 33, no. 1, pp. 82–95, 1971.

[31] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," *Subseries of Lecture Notes in Computer Science Edited by JG Carbonell and J. Siekmann*, p. 416.

[32] D. Needell, J. Tropp, and R. Vershynin, "Greedy signal recovery review," in *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*. IEEE, 2008, pp. 1048–1050.