# DECENTRALIZED EFFICIENT NONPARAMETRIC STOCHASTIC OPTIMIZATION

*Alec Koppel*[⋆]*, Santiago Paternain*[⋆]*, Cédric Richard*[†] *and Alejandro Ribeiro*[⋆]

[⋆]Department of Electrical and Systems Engineering, University of Pennsylvania
[†]Laboratory Lagrange, Observatory of the French Riviera, University of Nice Sophia-Antipolis

## ABSTRACT

We consider stochastic optimization problems defined over reproducing kernel Hilbert spaces (RKHS), where a multi-agent network aims to learn decision functions, i.e., nonlinear statistical models, that are optimal in terms of a global convex functional that aggregates data across the network, while only having access to locally observed sequentially available training examples. We address this problem by allowing each agent to learn a local regression function while enforcing consensus constraints. We use a penalized variant of functional stochastic gradient descent operating simultaneously with low-dimensional subspace projections. The resulting algorithm allows each individual agent to learn, based upon its locally observed data stream and message passing with its neighbors, a function that is provably close to globally optimal and satisfies the consensus constraints. Moreover, when constant learning rates are used, the complexity of the learned regression functions is guaranteed to be finite. For a multi-class kernel logistic regression task with Gaussian mixtures data, we observe stable function estimation and state of the art accuracy for distributed online multi-class classification.

## 1. INTRODUCTION

We consider decentralized online optimization problems: a network $\mathcal{G} = (V, \mathcal{E})$ of agents aims to minimize an objective that is a sum of local objectives available only to each node. The problem is online because data samples upon which local convex objectives depend are sequentially and locally observed by each agent. In this setting, agents aim to make inferences as well as one which has access to all data at a centralized location in advance. Rather than assuming agents seek a common vector $\mathbf{w} \in \mathbb{R}^p$, we address the case where agents seek to learn a common *decision function* $f(\mathbf{x})$ belonging to a reproducing kernel Hilbert space (RKHS), which represents, e.g., a nonlinear statistical model [1] or a continuous trajectory [2]. Nonlinear interpolators perform far better than their linear counterparts induced by the vector-valued convex problems [3], but few works [4] extend them to streaming decentralized settings that underlie Internet of Things [5, 6] and multi-robot [7, 8] applications.

Consider centralized vector-valued stochastic convex programming, which has been classically solved with stochastic gradient descent (SGD) [9]. SGD involves descending along the negative of the stochastic gradient rather than the true gradient to avoid the fact that computing the gradient of the average objective has complexity comparable to the training sample size, which could be infinite. In contrast, a stochastic program defined over a function space is an intractable variational inference problem in general, but when the function space is a RKHS [10], the Representer Theorem allows us to reduce an infinite space into a parameterization of weights and data samples [11]. Unfortunately, the resulting feasible set has complexity comparable to sample size $N$ (intractable for $N \to \infty$ [12]).

Efforts to mitigate the complexity of the function representation ("the curse of kernelization") have been developed that combine functional extensions of stochastic gradient method with compressions of the function sequence parameterization [13–17]. Mostly, such methods compress the function representation independent of the iterative sequence to which they are applied. In contrast, a method was recently proposed that combines greedily constructed [18] sparse subspace projections with functional SGD, and tailors the compression to preserve descent properties of the RKHS-valued stochastic process [19].

In this work, we extend [19] to multi-agent settings. Multiple distributed optimization tools may be used to develop such an extension; however, the Representer Theorem [11] has not been established for stochastic saddle point problems in RKHSs. Thus, we adopt an approximate primal-only approach via penalty methods [20, 21], which in decentralized optimization is called distributed gradient descent (DGD). Using functional stochastic extensions of DGD, together with the greedy projections designed in [19], we develop a method such that each agent, through its local data stream and neighborhood message passing, learns a memory-efficient approximation to the globally optimal function almost surely. Such global stability contrasts with nonlinear function estimation techniques such as dictionary learning [22–24] or neural networks [25] which exhibit instability due to non-convexity.

Subsequently, in Section 2 we clarify the problem of stochastic programming in the RKHS. In Section 3, we propose a penalty method that yields a decentralized online method for kernel regression without any complexity bottleneck based on functional stochastic gradient method combined with greedy subspace projections (Section 3.2). We establish that each agent's function sequence converges to a neighborhood of the globally optimal regression function with probability 1 in Section 4. In Section 5, we present an example of a decentralized online multi-class kernel logistic regression problem with Gaussian mixture data [17], and observe a state of the art trade-off between stability and accuracy.

## 2. FUNCTIONAL STOCHASTIC PROGRAMMING

Consider the problem of distributed expected risk minimization, 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 scattered across an interconnected network that represents, for instance, robotic teams [24], communication systems [26], or sensor networks [27]. To do so, we define a symmetric, connected, and directed network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = V$ nodes and $|\mathcal{E}| = E$ edges and denote as $n_i := \{j : (i, j) \in \mathcal{E}\}$ the neighborhood of agent $i$. Each agent $i \in \mathcal{V}$ observes a local data sequence as realizations $(\mathbf{x}_{i,n}, y_{i,n})$ from random pair $(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y} \subset \mathbb{R}^p \times \mathbb{R}$ and seeks to learn a common globally optimal regression function $f$ from the class function $\mathcal{H}$. This setting may be mathematically captured by associating to each node $i$ is a convex loss functional $\ell_i : \mathcal{H} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that quantifies the merit of the estimator $\tilde{f}(\mathbf{x}_i)$ evaluated at feature vector $\mathbf{x}_i$. This loss averaged over all possible $\mathbf{x}_i$ defines the statistical loss $L(\tilde{f}) := \sum_{i \in \mathcal{V}} \mathbb{E}_{\mathbf{x}_i, y_i}\left[\ell_i(\tilde{f}(\mathbf{x}_i), y_i)\right]$, which we combine with a Tikhonov regularizer to construct the regularized loss $R(\tilde{f}) := \operatorname{argmin}_{\tilde{f} \in \mathcal{H}} L(\tilde{f}) + (\lambda/2)\|\tilde{f}\|^2_{\mathcal{H}}$ [28, 29]. Then the globally optimal regression function $\tilde{f}^*$ is defined as

$$\tilde{f}^* = \operatorname*{argmin}_{\tilde{f} \in \mathcal{H}} \sum_{i \in \mathcal{V}} \left( \mathbb{E}_{\mathbf{x}_i, y_i}\left[\ell_i(\tilde{f}(\mathbf{x}_i), y_i)\right] + \frac{\lambda}{2}\|\tilde{f}\|^2_{\mathcal{H}} \right). \quad (1)$$

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

**Algorithm 1** Greedy Projected Penalty Method

**Require:** $\{\mathbf{x}_t, \mathbf{y}_t, \eta, \epsilon\}_{t=0,1,2,\ldots}$
**initialize** $f_{i,0}(\cdot) = 0, \mathbf{D}_{i,0} = [], \mathbf{w}_0 = []$, i.e. initial dictionary, coefficients are empty for each $i \in \mathcal{V}$
**for** $t = 0, 1, 2, \ldots$ **do**
**loop in parallel** for agent $i \in \mathcal{V}$
Observe local training example realization $(\mathbf{x}_{i,t}, y_{i,t})$
Send obs. $\mathbf{x}_{i,t}$ to nodes $j \in n_i$, receive scalar $f_{j,t}(\mathbf{x}_{i,t})$
Receive obs. $\mathbf{x}_{j,t}$ from nodes $j \in n_i$, send $f_{i,t}(\mathbf{x}_{j,t})$
Compute unconstrained stochastic grad. step [cf. (8)]
$$\tilde{f}_{i,t+1}(\cdot) = (1-\eta\lambda) f_{i,t} - \eta \nabla_{f_i} \hat{\psi}_{i,c}(f_i(\mathbf{x}_{i,t}), \mathbf{y}_{i,t}) \ .$$
Update params: $\tilde{\mathbf{D}}_{i,t+1} = [\mathbf{D}_{i,t}, \ \mathbf{x}_{i,t}]$, $\tilde{\mathbf{w}}_{i,t+1}$ [cf. (10)]
Greedily compress function using matching pursuit
$$(f_{i,t+1}, \mathbf{D}_{i,t+1}, \mathbf{w}_{i,t+1}) = \textbf{KOMP}(\tilde{f}_{i,t+1}, \tilde{\mathbf{D}}_{i,t+1}, \tilde{\mathbf{w}}_{i,t+1}, \epsilon)$$
**end loop**
**end for**

Observe that this global loss is a network-wide average (scaled by $V$) of all local losses; however, in multi-agent optimization, there is no global coordination among the agents in selecting function $\tilde{f}$, but rather, each agent, via its locally observed data and message passing with its neighbors, seeks to learn $\tilde{f}^*$. Thus, we allow agent $i$ to select a distinct $f_i$, but we enforce that at optimality all function estimates $f_i$ coincide, which yields the nonparametric decentralized stochastic program:

$$f^* = \underset{\{f_i\} \in \mathcal{H}}{\operatorname{argmin}} \quad \sum_{i \in \mathcal{V}} \left( \mathbb{E}_{\mathbf{x}_i, y_i} \left[ \ell_i(f_i(\mathbf{x}), y_i) \right] + \frac{\lambda}{2} \|f_i\|^2_{\mathcal{H}} \right) \quad \text{such that} \quad f_i = f_j \ , (i,j) \in \mathcal{E} \tag{2}$$

For further reference we define the stacked Hilbert space $\mathcal{H}^V$ of functions aggregated over the network whose elements are stacked functions $f(\cdot) = [f_1(\cdot); \cdots; f_V(\cdot)]$. Moreover, define stacked random vectors $\mathbf{x} = [\mathbf{x}_1; \cdots; \mathbf{x}_V] \in \mathbb{R}^{Vp}$ and $\mathbf{y} = [y_1; \cdots y_V] \in \mathbb{R}^V$ that represents $V$ labels or physical measurements, for instance.

The goal of this paper is to develop an algorithm to solve (2) in distributed online settings where nodes do not know the distribution of the random pair $(\mathbf{x}_i, y_i)$ but sequentially observe local independent training examples $(\mathbf{x}_{i,n}, y_{i,n})$. In the next section we discuss necessary details of the function space $\mathcal{H}^V$ to make (2) tractable.

### 2.1. Function Estimation in Reproducing Kernel Hilbert Spaces

The problem in (2) is intractable in general, since it defines a variational inference problem integrated over the unknown joint distribution $\mathbb{P}(\mathbf{x}, y)$. However, when $\mathcal{H}$ is equipped with a *reproducing kernel* $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (see [3, 30]), a functional problem of the form (1) may be reduced to a parametric form via the Representer Theorem [12, 31]. Thus, we restrict the Hilbert space in (2) to be one equipped with a kernel $\kappa$ that satisfies for all functions $\tilde{f} : \mathcal{X} \to \mathbb{R}$ in $\mathcal{H}$ and for all $\mathbf{x}_i \in \mathcal{X}$:

$$(i) \ \langle \tilde{f}, \kappa(\mathbf{x}_i, \cdot) \rangle_{\mathcal{H}} = \tilde{f}(\mathbf{x}_i) \qquad (ii) \ \mathcal{H} = \overline{\text{span}\{\kappa(\mathbf{x}_i, \cdot)\}}. \tag{3}$$

Here $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the Hilbert inner product for $\mathcal{H}$. Further assume that $\kappa$ is positive semidefinite, i.e. $\kappa(\mathbf{x}_i, \mathbf{x}_i') \geq 0$ for all $\mathbf{x}_i, \mathbf{x}_i' \in \mathcal{X}$. For kernelized regularized empirical risk minimization, the Representer Theorem [32] states that optimal $\tilde{f}$ in the function class $\mathcal{H}$ may be written in terms of kernel evaluations only of the training set

$$\tilde{f}(\mathbf{x}_i) = \sum_{n=1}^{N} w_{i,n} \kappa(\mathbf{x}_{i,n}, \mathbf{x}_i) \ , \tag{4}$$

where $\mathbf{w}_i = [w_{i,1}, \cdots, w_{i,N}]^T \in \mathbb{R}^N$ denotes a set of weights. The upper index $N$ in (4) is referred to as the model order, and for ERM the model order and training sample size are equal. By exploiting the Representer Theorem, we transform an infinite dimensional optimization problem in $\mathcal{H}^V$ into a finite $NV$-dimensional parametric problem. Thus, the RKHS provides a principled framework to solve nonparametric regression problems as a search over $\mathbb{R}^{VN}$ for a set of coefficients. However, when training examples $(\mathbf{x}_{i,n}, y_{i,n})$ become sequentially available or their total number $N$ is not finite, the complexity of representing a function $\tilde{f}$ in (4) approaches infinity as well, and thus demands an infeasible amount of memory. Thus, our goal is to solve (2) in an approximate manner such that each $f_i$ admits a finite representation near $f_i^*$, while satisfying the consensus constraints $f_i = f_j$ for $(i,j) \in \mathcal{E}$.

**Algorithm 2** 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 $\epsilon > 0$
**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** $j = 1, \ldots, \tilde{M}$ **do**
Find minimal approximation error with dictionary element $\mathbf{d}_j$ 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 index minimizing approx. error: $j^* = \operatorname{argmin}_{j \in \mathcal{I}} \gamma_j$
**if** minimal approx. error exceeds threshold $\gamma_{j^*} > \epsilon$
**stop**
**else**
Prune dictionary $\mathbf{D} \leftarrow \mathbf{D}_{\mathcal{I}\setminus\{j^*\}}$
Revise set $\mathcal{I} \leftarrow \mathcal{I} \setminus \{j^*\}$ and model order $M \leftarrow M - 1$.
Update weights $\mathbf{w}$ defined by current dictionary $\mathbf{D}$
$$\mathbf{w} = \underset{\mathbf{w} \in \mathbb{R}^M}{\operatorname{argmin}} \|\tilde{f}(\cdot) - \mathbf{w}^T \boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\|_{\mathcal{H}}$$
**end**
**end while**
**return** $f, \mathbf{D}, \mathbf{w}$ of model order $M \leq \tilde{M}$ such that $\|f - \tilde{f}\|_{\mathcal{H}} \leq \epsilon$

## 3. GREEDY PROJECTED PENALTY METHOD

We now develop an online decentralized iterative solution to (2) when the functions $\{f_i\}_{i \in \mathcal{V}}$ are elements of a RKHS, as detailed in Section 2.1. To exploit the properties of this function space, we require the applicability of the Representer Theorem [cf. (4)], but this result holds for any regularized minimization problem with a convex functional. Thus, we may address the consensus constraint $f_i = f_j \ , (i,j) \in \mathcal{E}$ in (2) by enforcing approximate consensus on estimates $f_i(\mathbf{x}_i) = f_j(\mathbf{x}_i)$ in expectation. Thus, we introduce the penalty function

$$\psi_c(f) = \sum_{i \in \mathcal{V}} \Big( R_i(f_i) + \frac{c}{2} \sum_{j \in n_i} \mathbb{E}_{\mathbf{x}_i} \left\{ [f_i(\mathbf{x}_i) - f_j(\mathbf{x}_i)]^2 \right\} \Big), \tag{5}$$

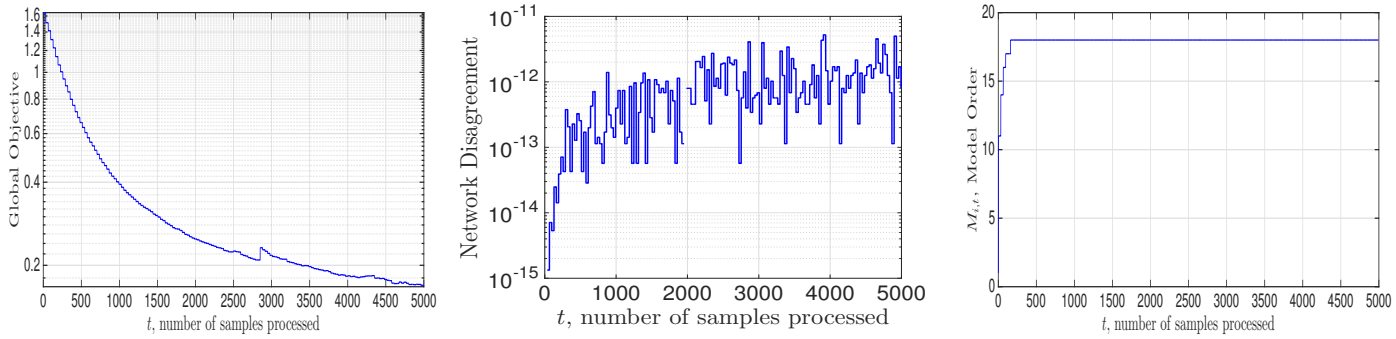
where $R_i(f_i) := \mathbb{E}_{\mathbf{x}_i, y_i} \left[ \ell_i(f_i(\mathbf{x}_i), y_i) \right] + (\lambda/2) \|f_i\|^2_{\mathcal{H}}$. Further define $f_c^* = \operatorname{argmin}_{f \in \mathcal{H}^V} \psi_c(f)$ and the local penalty as

$$\psi_{i,c}(f_i) = R_i(f_i) + \frac{c}{2} \sum_{j \in n_i} \mathbb{E}_{\mathbf{x}_i} \left\{ [f_i(\mathbf{x}_i) - f_j(\mathbf{x}_i)]^2 \right\}. \tag{6}$$

Observe from (5) - (6) that $\psi_c(f) = \sum_i \psi_{i,c}(f_i)$.

### 3.1. Functional Stochastic Gradient Method

The data distribution $\mathbb{P}(\mathbf{x}, \mathbf{y})$ is unknown, so minimizing $\psi_c(f)$ directly via variational inference is not possible. Rather than postulate a distribution for $(\mathbf{x}, \mathbf{y})$, we only require sequentially available independent and identically distributed samples $(\mathbf{x}_t, \mathbf{y}_t)$ from their joint density. Then, we address (5) using stochastic methods. Thus, compute $\nabla_f \hat{\psi}_{i,c}(f(\mathbf{x}_t), \mathbf{y}_t)$, the functional stochastic gradient (5), as in [15]




(a) Global objective vs. samples processed

(b) Disagreement vs. samples processed

(c) Model Order $M_{i,t}$ vs. samples processed

**Fig. 1**: In Fig. 1a, we plot the global objective $\sum_{i\in\mathcal{V}}(\mathbb{E}_{\mathbf{x}_i,y_i}[\ell_i(f_{i,t}(\mathbf{x}),y_i)])$ versus the number of samples processed, and observe convergence. In Fig. 1b we display the Hilbert-norm network disagreement $\sum_{(i,j)\in\mathcal{E}}\|f_{i,t}-f_{j,t}\|_{\mathcal{H}}^2$ with a penalty parameter $c$ that doubles every 200 samples. In Fig. 1c, we plot the model order of a randomly chosen agent's regression function, which stabilizes to 18 after 162 samples.

$$\nabla_f\hat{\psi}_{i,c}(f(\mathbf{x}_t),\mathbf{y}_t)=\ell_i'(f_i(\mathbf{x}_{i,t}),y_{i,t})\kappa(\mathbf{x}_{i,t},\cdot)+\lambda f_i +c\sum_{j\in n_i}(f_i(\mathbf{x}_{i,t})-f_j(\mathbf{x}_{i,t}))\kappa(\mathbf{x}_{i,t},\cdot). \tag{7}$$

Now we may define the distributed stochastic gradient method for the kernelized $\lambda$-regularized multi-agent problem in (2) as

$$f_{i,t+1}=f_{i,t}-\eta\nabla_f\hat{\psi}_{i,c}(f(\mathbf{x}_t),\mathbf{y}_t)\ , \tag{8}$$

where $\eta>0$ is an algorithm step-size. We further require that, given $\lambda>0$, the step-size satisfies $\eta<1/\lambda$ and the global sequence is initialized as $f_0=0\in\mathcal{H}^V$. With this initialization, the Representer Theorem (c.f. (4)) implies that, at time $t$, the function $f_{i,t}$ admits an expansion in terms of feature vectors $\mathbf{x}_{i,t}$ observed thus far as

$$f_{i,t}(\mathbf{x})=\sum_{n=1}^{t-1}w_{i,n}\kappa(\mathbf{x}_{i,n},\mathbf{x})=\mathbf{w}_{i,t}^T\boldsymbol{\kappa}_{\mathbf{X}_{i,t}}(\mathbf{x})\ . \tag{9}$$

On the right-hand side of (9) we have introduced the notation $\mathbf{X}_{i,t}=[\mathbf{x}_{i,1},\ldots,\mathbf{x}_{i,t-1}]\in\mathbb{R}^{p\times(t-1)}$, $\boldsymbol{\kappa}_{\mathbf{X}_{i,t}}(\cdot)=[\kappa(\mathbf{x}_{i,1},\cdot),\ldots,\kappa(\mathbf{x}_{i,t-1},\cdot)]^T$, and $\mathbf{w}_{i,t}=[w_{i,1},\ \ldots\ w_{i,t-1}]\in\mathbb{R}^{t-1}$. Moreover, observe that the kernel expansion in (9), together with the update (8), yields the fact that performing the stochastic gradient method in $\mathcal{H}^V$ amounts to $V$ parallel parametric updates on the dictionaries $\mathbf{X}_i$ and coefficients $\mathbf{w}_i$:

$$\mathbf{X}_{i,t+1}=[\mathbf{X}_{i,t},\ \ \mathbf{x}_{i,t}]\ , \tag{10}$$

$$[\mathbf{w}_{i,t+1}]_u=\begin{cases}(1-\eta\lambda)[\mathbf{w}_{i,t}]_u & \text{for } 0\le u\le t-1\\ -\eta\Big(\ell_i'(f_{i,t}(\mathbf{x}_{i,t}),y_{i,t})+c\sum_{j\in n_i}(f_{i,t}(\mathbf{x}_{i,t})-f_{j,t}(\mathbf{x}_{i,t}))\Big), & \end{cases}$$

where the second case on the last line of (10) is for $u=t$. This update causes $\mathbf{X}_{i,t+1}$ to have one more column than $\mathbf{X}_{i,t}$. We define the *model order* as number of data points $M_{i,t}$ in the dictionary of agent $i$ at time $t$ (the number of columns of $\mathbf{X}_t$). FSGD is such that $M_{i,t}=t-1$, and hence grows unbounded with iteration index $t$. Next we address this intractable memory growth such that we may execute stochastic descent through low-dimensional projections of the stochastic gradient [19].

### 3.2. Sparse Subspace Projections

To mitigate the complexity growth noted in Section 3.1, we approximate the function sequence (8) by one that is orthogonally projected onto subspaces $\mathcal{H}_{\mathbf{D}}\subseteq\mathcal{H}$ that consist only of functions that can be represented using some dictionary $\mathbf{D}=[\mathbf{d}_1,\ \ldots,\ \mathbf{d}_M]\in\mathbb{R}^{p\times M}$, i.e., $\mathcal{H}_{\mathbf{D}}=\{f\ :\ f(\cdot)=\sum_{n=1}^M w_n\kappa(\mathbf{d}_n,\cdot)=\mathbf{w}^T\boldsymbol{\kappa}_{\mathbf{D}}(\cdot)\}=\text{span}\{\kappa(\mathbf{d}_n,\cdot)\}_{n=1}^M$, and $\{\mathbf{d}_n\}\subset\{\mathbf{x}_u\}_{u\le t}$. For convenience we define $[\boldsymbol{\kappa}_{\mathbf{D}}(\cdot)=\kappa(\mathbf{d}_1,\cdot)\ldots\kappa(\mathbf{d}_M,\cdot)]$, and $\mathbf{K}_{\mathbf{D},\mathbf{D}}$ as the resulting kernel matrix from this dictionary. We enforce efficiency in function representation by selecting dictionaries $\mathbf{D}_i$ that $M_{i,t}<<\mathcal{O}(t)$ for each $i$, following [19]. To be specific, we propose replacing the local update (8) in which the dictionary grows at each iteration by its projection onto subspace $\mathcal{H}_{\mathbf{D}_{i,t+1}}=\text{span}\{\kappa(\mathbf{d}_{i,n},\cdot)\}_{n=1}^{M_{t+1}}$ as

$$f_{i,t+1}:=\mathcal{P}_{\mathcal{H}_{\mathbf{D}_{i,t+1}}}\Big[(1-\eta\lambda)f_{i,t}-\eta\Big(\nabla_{f_i}\ell_i(f_{i,t}(\mathbf{x}_{i,t}),y_{i,t}) +c\sum_{j\in n_i}(f_{i,t}(\mathbf{x}_{i,t})-f_{j,t}(\mathbf{x}_{i,t}))\kappa(\mathbf{x}_{i,t},\cdot)\Big)\Big]. \tag{11}$$

We define projection $\mathcal{P}$ onto subspace $\mathcal{H}_{\mathbf{D}_{i,t+1}}$ by the update (11). The coefficients and dictionary updates are given in Algorithm 1 - 2.

## 4. CONVERGENCE ANALYSIS

We turn to establishing –based in the analysis in Section IV of [19] – that Algorithm 1 converges with probability 1 to a neighborhood of the minimizer of the penalty function $\psi_c(f)$ [cf. (5)] and that the kernel dictionary that parameterizes the regression function $f_i$ for each agent $i$ remains finite. Let us define the local projected stochastic functional gradient associated with the update in (11) as

$$\tilde{\nabla}_{f_i}\hat{\psi}_{i,c}(f_{i,t}(\mathbf{x}_{i,t}),y_{i,t})= \Big(f_{i,t}-\mathcal{P}_{\mathcal{H}_{\mathbf{D}_{i,t+1}}}\Big[f_{i,t}-\eta\nabla_{f_i}\hat{\psi}_{i,c}(f_{i,t}(\mathbf{x}_{i,t}),y_{i,t})\Big]\Big)/\eta \tag{12}$$

such that the local update of Algorithm 1 [cf. (11)] may be expressed as a stochastic projected functional gradient descent

$$f_{i,t+1}=f_{i,t}-\eta\tilde{\nabla}_{f_i}\hat{\psi}_{i,c}(f_{i,t}(\mathbf{x}_{i,t}),y_{i,t})\ . \tag{13}$$

Subsequently, we require the following technical conditions common in the analysis of kernelized stochastic methods, see [33–35].

(A1) The sets $\mathcal{H}_{\mathbf{D}_{i,t}}$ in (11) intersect a finite norm ball $\|f\|_{\mathcal{H}}\le K$.

(A2) The feature space $\mathcal{X}\subset\mathbb{R}^p$ and target domain $\mathcal{Y}\subset\mathbb{R}$ are compact, and the reproducing kernel map satisfies

$$\sup_{\mathbf{x}\in\mathcal{X}}\sqrt{\kappa(\mathbf{x},\mathbf{x})}=X<\infty. \tag{14}$$

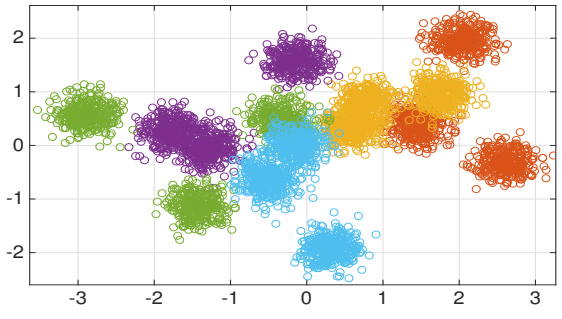
Moreover, the instantaneous losses $\ell_i:\mathcal{H}\times\mathcal{X}\times\mathcal{Y}\to\mathbb{R}$ are $C_i$-Lipschitz continuous for all $z\in\mathbb{R}$ for a fixed $y\in\mathcal{Y}$.

(A3) The local losses $\ell_i(f_i(\mathbf{x}),y)$ are convex and differentiable w.r.t. the scalar argument $f_i(\mathbf{x})$ on $\mathbb{R}$ for all $\mathbf{x}\in\mathcal{X}$, $y\in\mathcal{Y}$.

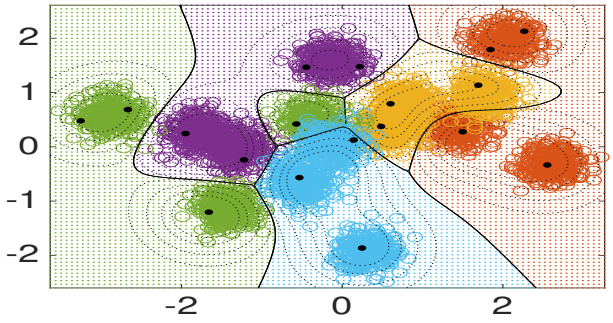
(A4) Let $\mathcal{F}_t$ be the filtration measuring the algorithm history: $\mathcal{F}_t=\{\mathbf{x}_u,y_u,f_u\}_{u=1}^t$. The projected functional stochastic gradient of the penalty which stacks (12) has finite conditional variance

$$\mathbb{E}[\|\tilde{\nabla}_f\hat{\psi}_c(f_t(\mathbf{x}_t),\mathbf{y}_t)\|_{\mathcal{H}}^2\mid\mathcal{F}_t]\le\sigma^2. \tag{15}$$

We show that under the previous assumptions the iterates of Algorithm 1 converge to a neighborhood of the minimizer of $\psi_c(f)$.



(a) Gaussian Mixtures training set.



(b) Decision surface via kernelized logistic loss.

**Fig. 2**: Visualizations of the Gaussian mixture data set (left) as in [17], where each class is denoted by a distinct color, and the learned low-memory multi-class kernel logistic regressor of a randomly chosen agent in the network (right), which attains 95.2% classification accuracy on a hold-out test set. Curved black lines denote decision boundaries between classes; dotted lines denote confidence intervals; bold black dots denote dictionary elements. Kernel dictionary elements concentrate at peaks of the Gaussian clusters and near points of class overlap.

**Theorem 1** *Consider the sequence $\{f_t\}$ generated by Algorithm 1 with $f_0 = 0$ and regularizer $\lambda > 0$. Suppose Assumptions 1-4 hold, and we select $\eta < 1/\lambda$ and compression budget $\epsilon = K\eta^{3/2}$ for any $K > 0$. Then we have convergence to a neighborhood with probability 1 as*

$$\liminf_{t\to\infty} \|f_t - f_c^*\|_{\mathcal{H}} \leq \frac{\sqrt{\eta}}{\lambda}\Big[2KV + \sqrt{\lambda\sigma^2}\Big] = \mathcal{O}(\sqrt{\eta}) \quad \text{a.s.} \tag{16}$$

Empirically, the use of constant step-sizes has the effect of maintaining consistent algorithm adaptivity in the face of new data. Moreover, we may apply Theorem 3 of [19], which guarantees the model order of the function sequence remains finite.

**Corollary 1** *Denote $f_t \in \mathcal{H}^V$ as the sequence defined by Algorithm 1 with $\eta$ and $\epsilon$ as in Theorem 1. Let $M_t$ be the model order of function $f_t$. Then there exists a finite upper bound $M^\infty$ such that, for all $t \geq 0$, the model order is always bounded as $M_t \leq M^\infty$, and the model order of the limiting function $f_c^\infty = \lim_t f_t$ is finite.*

Thus, use of constant step-sizes yields approximate convergence to $f_c^*$ while ensuring the memory requirements are always finite, as stated in Corollary 1. We are left to analyze the goodness of the solution $f_c^*$ as an approximation of the solution of the original problem (2). In particular, we establish consensus in the mean square sense. First we establish an upper bound on the penalty term in (5).

**Proposition 1** *Let Assumptions 2 and 3 hold. Let $f_c^*$ be the minimizer of $\psi_c(f)$ in (5) and $p^*$ be the optimal value of (2). Then*

$$\frac{1}{2}\sum_{i\in\mathcal{V}}\sum_{j\in n_i} \mathbb{E}_{\mathbf{x}_i}\left\{[f_{c,i}^*(\mathbf{x}_i) - f_{c,j}^*(\mathbf{x}_i)]^2\right\} \leq \frac{p^*}{c}. \tag{17}$$

As a consequence of the previous proposition, considering the limit with $c$ going to infinity in (17), allows us to write for all $i, j$ that

$$\lim_{c\to\infty} \frac{1}{2}\mathbb{E}_{\mathbf{x}_i}\left\{[f_{c,i}^*(\mathbf{x}_i) - f_{c,j}^*(\mathbf{x}_i)]^2\right\} = 0. \tag{18}$$

The latter is equivalent to consensus in the mean square sense.

## 5. NUMERICAL EXPERIMENTS

Consider the task of kernel logistic regression from multi-class training data that is scattered across a multi-agent network. In this case, the merit of a particular regressor for agent $i$ is quantified by its contribution to the class-conditional probability. We define a set of class-specific activation functions $f_{i,k} : \mathcal{X} \to \mathbb{R}$, and denote them jointly as $\mathbf{f}_i \in \mathcal{H}^K$, where $\{1, \ldots, K\}$ denotes the set of classes. Then, define the probabilistic model of the odds ratio of a sample being in class $k$ versus all others

$$P(y_i = k \,|\, \mathbf{x}_i) := \frac{\exp(f_{i,k}(\mathbf{x}_i))}{\sum_{k'} \exp(f_{i,k'}(\mathbf{x}_i))}. \tag{19}$$

The negative log likelihood defined by (19) is the instantaneous loss (see, e.g., [32]) at sample $(\mathbf{x}_{i,n}, y_{i,n})$:

$$\ell(\mathbf{f}_i, \mathbf{x}_{i,n}, y_{i,n}) = -\log P(y_i = y_{i,n}|\mathbf{x}_{i,n}) + \frac{\lambda}{2}\sum_k \|f_{i,k}\|_{\mathcal{H}}^2 \tag{20}$$

Following [17, 19], we generate a data set from Gaussian mixture models, which consists $N = 5000$ feature-label pairs for training and 2500 for testing. Each label $y_n$ was drawn uniformly at random from the label set. The corresponding feature vector $\mathbf{x}_n \in \mathbb{R}^p$ was then drawn from a planar Gaussian mixture model, i.e., $\mathbf{x} \,|\, y \sim (1/3)\sum_{j=1}^3 \mathcal{N}(\boldsymbol{\mu}_{y,j}, \sigma_{y,j}^2\mathbf{I})$ where $\sigma_{y,j}^2 = 0.2$ for all values of $y$ and $j$. The means $\boldsymbol{\mu}_{y,j}$ are themselves realizations of their own Gaussian distribution with class-dependent parameters, i.e., $\boldsymbol{\mu}_{y,j} \sim \mathcal{N}(\boldsymbol{\theta}_y, \sigma_y^2\mathbf{I})$, where $\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K\}$ are equitably spaced around the unit circle, one for each class label, and $\sigma_y^2 = 1.0$. We fix the number of classes $K = 5$, meaning that the feature distribution has 15 distinct modes. The data points are plotted in Figure 2a.

Here the $V = 20$ node digraph is random and with edges generated randomly with probability $1/5$ repeatedly until we obtain one that is connected, and then symmetrize it. We run Algorithm 1 when each agent observes a unique random stream of samples from this common training set, a Gaussian kernel is used with bandwidth $d = 0.6$, with constant learning rate $\eta = 3$, compression budget $\epsilon = \eta^{3/2}$, parsimony constant $K = 0.04$, mini-batch size 32, and regularizer $\lambda = 10^{-6}$. The penalty coefficient is initialized $c = 0.01$ and doubled every 200 samples.

The results of this implementation[1] in Figures 2b and 1. In Figure 1a, we plot the global objective $\sum_{i\in\mathcal{V}}(\mathbb{E}_{\mathbf{x}_i,y_i}[\ell_i(f_{i,t}(\mathbf{x}), y_i)])$ relative to the number of training examples processed, and observe stable convergence to a global minimum. In Figure 1b we display Hilbert-norm network disagreement $\sum_{(i,j)\in\mathcal{E}} \|f_{i,t} - f_{j,t}\|_{\mathcal{H}}^2$ versus observed sample points. Since each regression function is initialized as null, initially the disagreement is trivially null, but it remains small over the function sample path as model training occurs. Moreover, the model order of an arbitrarily chosen agent $i = 15$ versus samples processed is given in Figure 1c: observe that the model order stabilizes after only a couple hundred training examples to 18, which is only a couple more than 15, the number of modes of the joint data density function. The resulting decision surface of node 15 is given in Figure 2b, which achieves 95.2% classification accuracy on the test set which is comparable to existing centralized batch approaches (see Table 2 of [19]) to kernel logistic regression.

[1] We would like to thank Garrett Warnell of the U.S. Army Research Laboratory-CISD for invaluable implementation assistance.

## 6. REFERENCES


[1] M. Anthony and P. L. Bartlett, *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

[2] Z. Marinho, B. Boots, A. Dragan, A. Byravan, G. J. Gordon, and S. Srinivasa, "Functional gradient motion planning in reproducing kernel hilbert spaces," in *Proceedings of Robotics: Science and Systems*, Ann Arbor, MI, July 2016.

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

[4] P. A. Forero, A. Cano, and G. B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, no. May, pp. 1663–1707, 2010.

[5] J. Liu, Q. Chen, and H. D. Sherali, "Algorithm design for femtocell base station placement in commercial building environments," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 2951–2955.

[6] A. Ghosh and S. Sarkar, "Pricing for profit in internet of things," in *Information Theory (ISIT), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 2211–2215.

[7] A. Koppel, J. Fink, G. Warnell, E. Stump, and A. Ribeiro, "Online learning for characterizing unknown environments in ground robotic vehicle models," in *Proc. Int. Conf. Intelligent Robots and Systems*.

[8] M. Schwager, P. Dames, D. Rus, and V. Kumar, "A multi-robot control policy for information gathering in the presence of unknown hazards," in *Robotics Research*. Springer, 2017, pp. 455–472.

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

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] O. Dekel, S. Shalev-Shwartz, and Y. Singer, "The forgetron: A kernel-based perceptron on a fixed budget," in *Advances in Neural Information Processing Systems 18*. MIT Press, 2006, p. 259266. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=78226

[17] 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.

[18] 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.

[19] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "Parsimonious online learning with kernels via sparse projections in function space," *arXiv preprint arXiv:1612.04111*, 2016.

[20] B. Johansson, T. Keviczky, M. Johansson, and K. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *Proc. of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico, 2008, pp. 4185–4190.

[21] S. Ram, A. Nedic, and V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J Optimiz. Theory App.*, vol. 147, no. 3, pp. 516–545, Sep. 2010.

[22] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 4, pp. 791–804, 2012.

[23] P. Chainais and C. Richard, "Learning a common dictionary over a sensor network," in *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2013 IEEE 5th International Workshop on*. IEEE, 2013, pp. 133–136.

[24] A. Koppel, G. Warnell, E. Stump, and A. Ribeiro, "D4l: Decentralized dynamic discriminative dictionary learning," *IEEE Trans. Signal and Info. Process. over Networks*, vol. (submitted), June 2017, available at http://www.seas.upenn.edu/ aribeiro/wiki.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[26] A. Ribeiro, "Ergodic stochastic optimization algorithms for wireless communication and networking," *IEEE Transactions on Signal Processing*, vol. 58, no. 12, pp. 6369–6386, 2010.

[27] R. J. Kozick and B. M. Sadler, "Source localization with distributed sensor arrays and partial spatial coherence," *IEEE Transactions on Signal Processing*, vol. 52, no. 3, pp. 601–616, 2004.

[28] 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.

[29] 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.

[30] 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

[31] R. Wheeden, 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

[32] K. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT press, 2012.

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

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

[35] Y. Nesterov, "Introductory lectures on convex programming volume i: Basic course," 1998.