A HYBRID DICTIONARY APPROACH FOR DISTRIBUTED KERNEL ADAPTIVE FILTERING IN DIFFUSION NETWORKS

Ban-Sok Shin[⋆]*, Masahiro Yukawa*[†§]*, Renato L. G. Cavalcante*[‡]*, and Armin Dekorsy*[⋆]

[⋆]Dept. of Communications Engineering, University of Bremen, Germany
[†]Dept. of Electronics and Electrical Engineering, Keio University, Japan
[§]Center for Advanced Intelligence Project, RIKEN, Japan
[‡] Fraunhofer Heinrich Hertz Institute and Technical University of Berlin, Germany
shin@ant.uni-bremen.de yukawa@elec.keio.ac.jp renato.cavalcante@hhi.fraunhofer.de dekorsy@ant.uni-bremen.de

ABSTRACT

We propose a hybrid dictionary approach for distributed kernel-based adaptive learning of a nonlinear function by a network of nodes. The hybrid dictionary incorporates a local part to improve learning of high frequency components in the function within the local domain of each node and a global part to provide a consensus estimate of the function over the whole region of interest. We apply our scheme to the reconstruction of a spatial distribution by a network of mobile nodes. Performance evaluations show that high frequency components are reconstructed accurately by our hybrid dictionary approach while common schemes are not able to recover them completely.

***Index Terms*—** Kernel adaptive filter, hybrid dictionary, distributed adaptive learning, spatial reconstruction

1. INTRODUCTION

Distributed kernel adaptive filters and kernel least-squares approaches for sensor networks are a topic of ongoing research in the signal processing community [1–8]. A common scenario is the reconstruction of an unknown, nonlinear function by a network of nodes. Applications lie e.g. in the field of environmental monitoring of physical quantities such as gas, temperature and their spatial reconstruction by sensor networks [9, 10]. However, one of the main challenges for distributed kernel-based estimation algorithms is the kernel-specific dictionary. Approaches [7, 8] utilize online dictionary learning schemes since the decentralization is implemented via a sequential update of the filter weight vector from node to node. However, these algorithms do not work in a parallel fashion, i.e. nodes do not execute their update synchronously, and they do not enforce a consensus estimate in the network. On the other hand, approaches as in [1, 2, 4] enable a synchronous update of the filter weight vector among the nodes where [1, 2] enforce consensus on the weight vectors in the network. Nevertheless, in these approaches the dictionary is assumed to be known a-priori and to be equal among all nodes in the network. Furthermore, it stays fixed during the estimation process and thus, online dictionary strategies cannot be applied to these schemes.

To incorporate online dictionary learning while enabling parallel processing in the network we propose a novel architecture for distributed consensus-based kernel adaptive filtering by incorporating a *hybrid dictionary*. This dictionary consists of a fixed part common to all nodes and a dynamic part which is individual to each node. The dynamic part is the *local dictionary* which is responsible for an accurate reconstruction in the input domain relevant to the specific node. The fixed part or *global dictionary* is used to deliver each node with a rough estimate of the function over the whole region of interest where we demand consensus on the global estimate. Figure 1 illustrates the main concept of our proposed approach: The depicted nonlinear function includes some high frequency components within the input domain of node j. This node samples the nonlinear function in its input domain and based on these input samples it builds a local dictionary. With the help of the local dictionary the node is able to reconstruct the high frequency components resulting in the green curve while achieving a global consensus estimate via the global dictionary. In contrast, using a global dictionary only will result in the estimate illustrated by the black curve where the high frequency components cannot be recovered. The motivation for this approach is that nodes acquire a detailed view of their direct vicinity to perform tasks relevant to their specific input domain while being aware of the global situation.

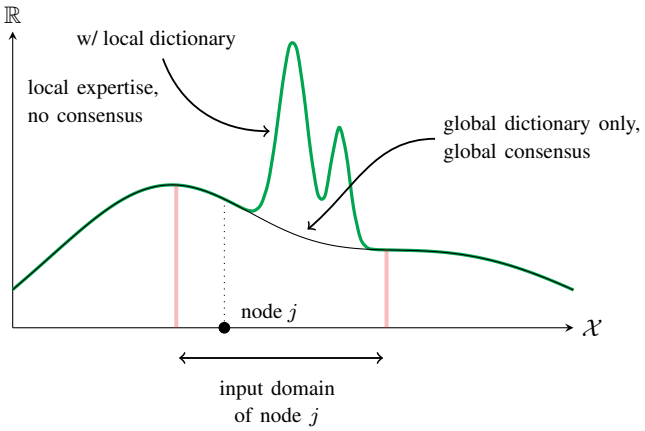

Fig. 1. The main concept of the proposed hybrid dictionary approach.

We combine the hybrid dictionary approach with the CHYPASS algorithm from [11] and with consensus averaging to develop a distributed consensus-based scheme. In numerical evaluations, we

M. Yukawa is thankful to JSPS Grants-in-Aid (15K06081, 15K13986, 15H02757).

show that employing an additional local dictionary significantly improves the reconstruction performance in the input domain of the nodes while providing a good estimate of the whole function.

2. SYSTEM MODEL

We consider a network of J nodes observing a common nonlinear, continuous function $\psi : \mathcal{X} \to \mathbb{R}$ which can model e.g. the spatial distribution of temperature or gas. We denote the input space by $\mathcal{X} \subseteq \mathbb{R}^L$ and the output space by $\mathbb{R}$. The function ψ is composed as a superposition of a global part $\psi_{\mathrm{G}} : \mathcal{X} \to \mathbb{R}$ and a local part $\psi_{\mathrm{L}} : \mathcal{X} \to \mathbb{R}$ where ψ_{G} incorporates low and ψ_{L} high frequency components. Each node j measures ψ per time index $k \in \mathbb{N}$ by feeding it with its input sample $\boldsymbol{x}_{j,k} \in \mathbb{R}^L$ following

$$d_{j,k} = \psi(\boldsymbol{x}_{j,k}) + n_{j,k} = \psi_{\mathrm{G}}(\boldsymbol{x}_{j,k}) + \psi_{\mathrm{L}}(\boldsymbol{x}_{j,k}) + n_{j,k}. \quad (1)$$

Hence, each node j acquires a new observation $d_{j,k}$ of ψ per time k based on its input sample $\boldsymbol{x}_{j,k}$. The noise $n_{j,k}$ is assumed to be zero-mean white Gaussian with variance σ_n^2. To describe the network we use a graph $\mathcal{G} = \{\mathcal{J}, \mathcal{E}\}$ with a set of nodes $\mathcal{J}$ and a set of edges $\mathcal{E}$. The set $\mathcal{E}$ contains all connections among the nodes in the network where we assume that each node is connected to itself. Each node j has a neighborhood given by the set $\mathcal{N}_j$ containing all nodes connected to it (including itself). We consider undirected and connected graphs, i.e. edges are symmetric and each node can be reached by any other node over multiple hops, respectively. The objective of the network is to distributedly reconstruct the global ψ_{G} and local ψ_{L} by an exchange of information among neighboring nodes. Additionally, we demand that each node j refines the reconstruction of ψ_{L} in its local domain based on its measurements acquired over time.

3. DIFFUSION-BASED KERNEL ADAPTIVE FILTER WITH HYBRID DICTIONARY

3.1. Derivation

To derive our hybrid dictionary approach, we first introduce a common kernel adaptive filter $\varphi_j : \mathcal{X} \to \mathbb{R}$ per node j. The filter φ_j utilizes a positive-definite kernel $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with a dictionary set $\mathcal{D} = \{\kappa(\,\cdot\,, \bar{\boldsymbol{x}}_n)\}_{n=1}^N$ containing functions $\kappa(\,\cdot\,, \bar{\boldsymbol{x}}_n)$ centered around $\bar{\boldsymbol{x}}_n \in \mathcal{X}$ [12]. The output for an arbitrary input sample $\boldsymbol{x} \in \mathcal{X}$ of the kernel adaptive filter $\varphi_j(\boldsymbol{x})$ as an estimate of $\psi(\boldsymbol{x})$ is given by

$$\varphi_j(\boldsymbol{x}) := \sum_{n=1}^{N} w_{j,n} \kappa(\boldsymbol{x}, \bar{\boldsymbol{x}}_n) = \langle \boldsymbol{w}_j, \boldsymbol{\kappa}(\boldsymbol{x}) \rangle \quad (2)$$

with the vectors

$$\boldsymbol{w}_j := [w_{j,1}, \ldots, w_{j,N}]^{\mathsf{T}},$$
$$\boldsymbol{\kappa}(\boldsymbol{x}) := [\kappa(\boldsymbol{x}, \bar{\boldsymbol{x}}_1), \ldots, \kappa(\boldsymbol{x}, \bar{\boldsymbol{x}}_N)]^{\mathsf{T}}.$$

We denote the standard inner product as $\langle \cdot, \cdot \rangle$ and the Euclidean norm as $||\cdot||$. We observe that φ_j can be parameterized by the weight vector $\boldsymbol{w}_j$ and many algorithms have been developed to compute $\boldsymbol{w}_j$ in an adaptive manner, e.g. [13–16]. A commonly used kernel function is the Gaussian kernel $\kappa(\boldsymbol{x}_1, \boldsymbol{x}_2) = \exp\left(-\frac{||\boldsymbol{x}_1 - \boldsymbol{x}_2||^2}{2\zeta^2}\right)$ with kernel bandwidth ζ.

For the hybrid dictionary approach we employ two kernel functions per φ_j, one global kernel κ_{G} and one local kernel κ_{L}. We define the global dictionary as $\mathcal{D}^{\mathrm{G}} := \{\kappa_{\mathrm{G}}(\,\cdot\,, \bar{\boldsymbol{x}}_n^{\mathrm{G}})\}_{n=1}^N$ of cardinality N with global dictionary entries $\bar{\boldsymbol{x}}_n^{\mathrm{G}}$. The global dictionary $\mathcal{D}^{\mathrm{G}}$ is equal for all nodes and stays fixed, i.e. its sample set does not change over time. For κ_{L} we employ the local dictionary $\mathcal{D}_{j,k}^{\mathrm{L}} := \{\kappa_{\mathrm{L}}(\,\cdot\,, \boldsymbol{x}_{j,n})\}_{n \in \mathcal{R}_{j,k}}$ with the index set $\mathcal{R}_{j,k} := \{n_{j,1}, \ldots, n_{j,R_{j,k}}\} \subset \{0, 1, \ldots, k\}$ of cardinality $R_{j,k}$. The local dictionary depends on the node j and is allowed to grow over time k. The index set $\mathcal{R}_{j,k}$ selects a subset of the input samples $\boldsymbol{x}_{j,k}$ used over time by node j. Thus, the corresponding kernel vector $\boldsymbol{\kappa}_j(\boldsymbol{x})$ for the local dictionary depends on node j and changes over time. To incorporate the global and local kernels with their dictionaries we separate the node specific weight vector $\boldsymbol{w}_{j,k}$ and the corresponding kernel vector $\boldsymbol{\kappa}_j(\boldsymbol{x})$ into a global and a local part:

$$\boldsymbol{w}_{j,k} := \begin{bmatrix} \boldsymbol{w}_{j,k}^{\mathrm{G}} \\ \boldsymbol{w}_{j,k}^{\mathrm{L}} \end{bmatrix}, \quad \boldsymbol{\kappa}_j(\boldsymbol{x}) := \begin{bmatrix} \boldsymbol{\kappa}_{\mathrm{G}}(\boldsymbol{x}) \\ \boldsymbol{\kappa}_{\mathrm{L},j}(\boldsymbol{x}) \end{bmatrix}$$

where $\boldsymbol{w}_{j,k}^{\mathrm{G}} \in \mathbb{R}^{N \times 1}$ is the global and $\boldsymbol{w}_{j,k}^{\mathrm{L}} \in \mathbb{R}^{R_{j,k} \times 1}$ the local weight vector and

$$\boldsymbol{\kappa}_{\mathrm{G}}(\boldsymbol{x}) := [\kappa_{\mathrm{G}}(\boldsymbol{x}, \bar{\boldsymbol{x}}_1^{\mathrm{G}}), \ldots, \kappa_{\mathrm{G}}(\boldsymbol{x}, \bar{\boldsymbol{x}}_N^{\mathrm{G}})]^{\mathsf{T}},$$
$$\boldsymbol{\kappa}_{\mathrm{L},j}(\boldsymbol{x}) := [\kappa_{\mathrm{L}}(\boldsymbol{x}, \boldsymbol{x}_{n_{j,1}}), \ldots, \kappa_{\mathrm{L}}(\boldsymbol{x}, \boldsymbol{x}_{n_{j,R_{j,k}}})]^{\mathsf{T}}.$$

We assume that both the global and local dictionaries are linearly independent such that their corresponding kernel Gram matrices $\bar{\boldsymbol{K}}$ and $\tilde{\boldsymbol{K}}_{j,k}$ are positive definite:

$$\bar{\boldsymbol{K}} := \begin{bmatrix} \kappa_{\mathrm{G}}(\bar{\boldsymbol{x}}_1^{\mathrm{G}}, \bar{\boldsymbol{x}}_1^{\mathrm{G}}) & \ldots & \kappa_{\mathrm{G}}(\bar{\boldsymbol{x}}_1^{\mathrm{G}}, \bar{\boldsymbol{x}}_N^{\mathrm{G}}) \\ \vdots & \ddots & \vdots \\ \kappa_{\mathrm{G}}(\bar{\boldsymbol{x}}_N^{\mathrm{G}}, \bar{\boldsymbol{x}}_1^{\mathrm{G}}) & \ldots & \kappa_{\mathrm{G}}(\bar{\boldsymbol{x}}_N^{\mathrm{G}}, \bar{\boldsymbol{x}}_N^{\mathrm{G}}) \end{bmatrix} \in \mathbb{R}^{N \times N}$$

$$\tilde{\boldsymbol{K}}_{j,k} := \begin{bmatrix} \kappa_{\mathrm{L}}(\boldsymbol{x}_{n_{j,1}}, \boldsymbol{x}_{n_{j,1}}) & \ldots & \kappa_{\mathrm{L}}(\boldsymbol{x}_{n_{j,R_{j,k}}}, \boldsymbol{x}_{n_{j,1}}) \\ \vdots & \ddots & \vdots \\ \kappa_{\mathrm{L}}(\boldsymbol{x}_{n_{j,1}}, \boldsymbol{x}_{n_{j,R_{j,k}}}) & \ldots & \kappa_{\mathrm{L}}(\boldsymbol{x}_{n_{j,R_{j,k}}}, \boldsymbol{x}_{n_{j,R_{j,k}}}) \end{bmatrix} \in \mathbb{R}^{R_{j,k} \times R_{j,k}}.$$

In contrast to the global Gram matrix $\bar{\boldsymbol{K}}$ the local Gram matrices $\tilde{\boldsymbol{K}}_{j,k}$ will be specific to each node j and might vary in their dimensions depending on time k. We include both matrices into a full kernel Gram matrix $\boldsymbol{K}_{j,k}$ as

$$\boldsymbol{K}_{j,k} := \begin{bmatrix} \bar{\boldsymbol{K}} & \boldsymbol{0} \\ \boldsymbol{0} & \tilde{\boldsymbol{K}}_{j,k} \end{bmatrix} \in \mathbb{R}^{NR_{j,k} \times NR_{j,k}}.$$

Our proposed algorithm consists of two steps as in the D-CHYPASS from [17]: 1) a local update on the weight vector $\boldsymbol{w}_{j,k}$ per node j and 2) a diffusion step exchanging the global weight vectors $\boldsymbol{w}_{j,k}^{\mathrm{G}}$ among neighboring nodes to achieve consensus on the global reconstruction. For step 1) we define a hyperplane $H_{j,k}$ per node j and time k containing those weight vectors $\boldsymbol{w}$ which set the instantaneous error between desired output $d_{j,k}$ and estimated output $\varphi_{j,k}(\boldsymbol{x}_{j,k})$ to zero:

$$H_{j,k} := \left\{ \boldsymbol{w} \in \mathbb{R}^{NR_{j,k}} | \langle \boldsymbol{w}, \boldsymbol{K}_{j,k}^{-1} \boldsymbol{\kappa}(\boldsymbol{x}_{j,k}) \rangle_{\boldsymbol{K}_{j,k}} = d_{j,k} \right\} \quad (3)$$

with the $\boldsymbol{K}_{j,k}$ inner product defined as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle_{\boldsymbol{K}_{j,k}} := \boldsymbol{a}^{\mathsf{T}} \boldsymbol{K}_{j,k} \boldsymbol{b}$ for any vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^{NR_{j,k}}$. This metric showed improved convergence speed and steady-state performance in previous works [18,19]. Based on the CHYPASS algorithm [11] we apply the following update rule on $\boldsymbol{w}_{j,k}$ per node j:

$$\boldsymbol{v}_{j,k+1} = \boldsymbol{w}_{j,k} - \mu \left(\boldsymbol{w}_{j,k} - P_{H_{j,k}}^{\boldsymbol{K}_{j,k}}(\boldsymbol{w}_{j,k}) \right), \quad (4)$$

where $P_{H_{j,k}}^{\boldsymbol{K}_{j,k}}(\boldsymbol{w}_{j,k})$ is the $\boldsymbol{K}_{j,k}$-orthogonal projection[1] of $\boldsymbol{w}_{j,k}$ onto the hyperplane $H_{j,k}$ and $\boldsymbol{v}_{j,k+1} = [(\boldsymbol{v}_{j,k}^{\mathrm{G}})^{\mathsf{T}}, (\boldsymbol{v}_{j,k}^{\mathrm{L}})^{\mathsf{T}}]^{\mathsf{T}}$ is an intermediate variable at node j. The projection can be evaluated via [20]

$$P_{H_{j,k}}^{\boldsymbol{K}_{j,k}}(\boldsymbol{w}) = \boldsymbol{w} - \frac{\boldsymbol{w}^{\mathsf{T}}\boldsymbol{\kappa}(\boldsymbol{x}_{j,k}) - d_{j,k}}{||\boldsymbol{K}_{j,k}^{-1}\boldsymbol{\kappa}(\boldsymbol{x}_{j,k})||_{\boldsymbol{K}_{j,k}}^2}\boldsymbol{K}_{j,k}^{-1}\boldsymbol{\kappa}(\boldsymbol{x}_{j,k}). \quad (5)$$

To construct a distributed scheme and to achieve consensus on the global reconstruction of ψ, we apply consensus averaging on the global weight vectors. Then each node j fuses its intermediate variable $\boldsymbol{v}_{j,k+1}^{\mathrm{G}}$ with those from neighboring nodes $i \in \mathcal{N}_j$ via the matrix $\boldsymbol{G} \in \mathbb{R}^{J\times J}$. This matrix assigns weights to all edges in the network and thus, can be used to achieve a weighted average on the global weight vectors in the network. Each node j performs the fusion step $\boldsymbol{w}_{j,k+1}^{\mathrm{G}} = \sum_{i\in\mathcal{N}_j} \boldsymbol{G}_{ji}\boldsymbol{v}_{i,k+1}^{\mathrm{G}}$. Then, if matrix $\boldsymbol{G}$ fulfills the conditions[2] $||\boldsymbol{G} - (1/J)\mathbf{1}_J\mathbf{1}_J^{\mathsf{T}}||_2 < 1$ and $\boldsymbol{G}\mathbf{1}_J = \mathbf{1}_J$ vectors $\boldsymbol{w}_{j,k}^{\mathrm{G}}$ will converge to the average of all global weight vectors $\boldsymbol{w}_{j,k}^{\mathrm{G}}$ as a consensus state [21]. The local weight vectors are kept at each node as private information via $\boldsymbol{w}_{j,k+1}^{\mathrm{L}} = \boldsymbol{v}_{j,k+1}^{\mathrm{L}}$. Thus, each node j conserves its local expertise for its own refinement based on the local dictionary. Then, each node j is able to reconstruct the function ψ for arbitrary inputs $\boldsymbol{x}$ via $\varphi_{j,k}(\boldsymbol{x}) = \langle \boldsymbol{w}_{j,k}, \boldsymbol{\kappa}(\boldsymbol{x})\rangle$. We call the proposed algorithm the *Hybrid Diffusion-Based CHYPASS* (HD-CHYPASS) due to its hybrid dictionary approach and its local CHYPASS-based kernel adaptive filter.

3.2. Local Dictionary Learning

To build the local dictionary we use the input samples $\boldsymbol{x}_{j,k}$ of each node j. Per time k the node j uses a new input sample $\boldsymbol{x}_{j,k}$ which is examined for inclusion into the local dictionary $\mathcal{D}_{j,k}^{\mathrm{L}}$. To this end, each node j uses the coherence criterion [16] and an estimation error criterion. Both criteria have to be met by the current sample $\boldsymbol{x}_{j,k}$ to be included into the local dictionary. The coherence criterion evaluates the similarity of the current input sample $\boldsymbol{x}_{j,k}$ with the already included dictionary samples in $\mathcal{D}_{j,k}^{\mathrm{L}}$:

$$\max_{n\in\mathcal{R}_{j,k}} |\kappa_{\mathrm{L}}(\boldsymbol{x}_{j,k}, \boldsymbol{x}_{j,n})| \leq \tau \quad (6)$$

where $0 < \tau \leq 1$ is the coherence threshold. If an input sample $\boldsymbol{x}_{j,k}$ fulfills above condition, the corresponding a priori estimation error $e_{j,k}$ is investigated defined as

$$e_{j,k} := d_{j,k} - \varphi_{j,k}(\boldsymbol{x}_{j,k}) = d_{j,k} - \langle \boldsymbol{w}_{j,k}, \boldsymbol{\kappa}(\boldsymbol{x}_{j,k})\rangle \quad (7)$$

Then the local dictionary is updated as follows assuming that the coherence criterion has been passed successfully [22]:

$$\mathcal{D}_{j,k+1}^{\mathrm{L}} := \begin{cases} \mathcal{D}_{j,k}^{\mathrm{L}} \cup \{\kappa_{\mathrm{L}}(\cdot\,, \boldsymbol{x}_{j,k})\} & \text{if } |e_{j,k}/d_{j,k}| > \varepsilon \\ \mathcal{D}_{j,k}^{\mathrm{L}} & \text{else} \end{cases} \quad (8)$$

where $\varepsilon \geq 0$ is the error threshold. The error criterion includes the measurement $d_{j,k}$ into the dictionary learning process and thus makes sure that input samples where the estimation by the adaptive filter $\varphi_{j,k}$ is of low accuracy are added for further adaptation. In case the dictionary $\mathcal{D}_{j,k}^{\mathrm{L}}$ grows by a new sample, the current local vector $\boldsymbol{w}_{j,k}^{\mathrm{L}}$ is expanded by a single zero entry which corresponds to the weight for the newly added dictionary sample. Then update (4) for the intermediate weight vector becomes

$$\boldsymbol{v}_{j,k+1} = \begin{bmatrix} \boldsymbol{w}_{j,k} \\ 0 \end{bmatrix} - \mu\frac{[\boldsymbol{w}_{j,k}^{\mathsf{T}}, 0]\,\boldsymbol{\kappa}(\boldsymbol{x}_{j,k}) - d_{j,k}}{||\boldsymbol{K}_{j,k}^{-1}\boldsymbol{\kappa}(\boldsymbol{x}_{j,k})||_{\boldsymbol{K}_{j,k}}^2}\boldsymbol{K}_{j,k}^{-1}\boldsymbol{\kappa}(\boldsymbol{x}_{j,k})$$

In case the dictionary $\mathcal{D}_{j,k}^{\mathrm{L}}$ is not changed, the weight vector $\boldsymbol{w}_{j,k}$ stays the same.

Remark: Per time k an inversion of $\boldsymbol{K}_{j,k}$ has to be executed by each node causing high computational complexity, see (5). However, the global Gram matrix $\bar{\boldsymbol{K}}$ stays fixed over time and thus, only needs to be inverted once before the algorithm iterates. The complexity of the inversion of the local Gram matrix $\tilde{\boldsymbol{K}}_{j,k}$ can be significantly reduced by applying the *selective update* [19]. This strategy allows the update of only a few coefficients in the local vector $\boldsymbol{w}_{j,k}^{\mathrm{L}}$ per time k with minor performance loss. By this, the size of the local Gram matrix $\tilde{\boldsymbol{K}}_{j,k}$ and thus the complexity can be drastically reduced.

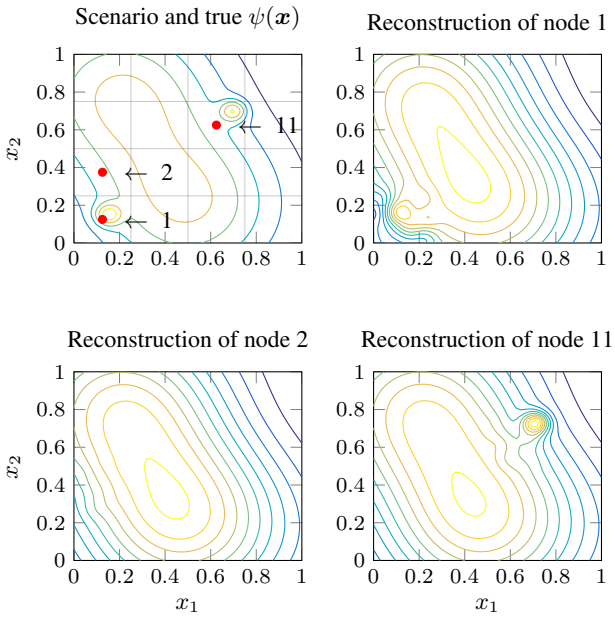

Fig. 2. Upper left: basic scenario with node regions and true $\psi(\boldsymbol{x})$. Remaining plots: node-specific reconstructions using the *proposed method*.

4. NUMERICAL EVALUATION

We apply HD-CHYPASS to the spatial reconstruction of multiple Gaussian functions over the unit-square area $\mathcal{X} = [0,1]^2$. Following model (1) the global $\psi_{\mathrm{G}}(\boldsymbol{x})$ incorporates low and the local $\psi_{\mathrm{L}}(\boldsymbol{x}_{j,k})$ high frequency components. Both parts are given as

$$\psi_{\mathrm{G}}(\boldsymbol{x}) = \exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{p}_1||^2}{2\cdot 0.3^2}\right) + \exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{p}_2||^2}{2\cdot 0.3^2}\right)$$

$$\psi_{\mathrm{L}}(\boldsymbol{x}) = 0.8\exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{p}_3||^2}{2\cdot 0.04^2}\right) + 0.8\exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{p}_4||^2}{2\cdot 0.04^2}\right)$$

with positions $\boldsymbol{p}_1 = [0.5, 0.2]^{\mathsf{T}}$, $\boldsymbol{p}_2 = [0.2, 0.8]^{\mathsf{T}}$, $\boldsymbol{p}_3 = [0.7, 0.7]^{\mathsf{T}}$, $\boldsymbol{p}_4 = [0.15, 0.15]^{\mathsf{T}}$. We consider a mobile sensor network of $J = 16$

[1] The $\boldsymbol{K}$-projection of a vector $\boldsymbol{w}$ onto a closed convex set C is defined as $P_C^{\boldsymbol{K}}(\boldsymbol{w}) := \mathrm{argmin}_{\boldsymbol{y}\in C}||\boldsymbol{y}-\boldsymbol{w}||_{\boldsymbol{K}}$ with the positive definite matrix $\boldsymbol{K}$ [13]

[2] The vector $\mathbf{1}_J$ is the $J \times 1$ vector of only ones and $||\boldsymbol{X}||_2$ denotes the spectral norm of a matrix $\boldsymbol{X}$

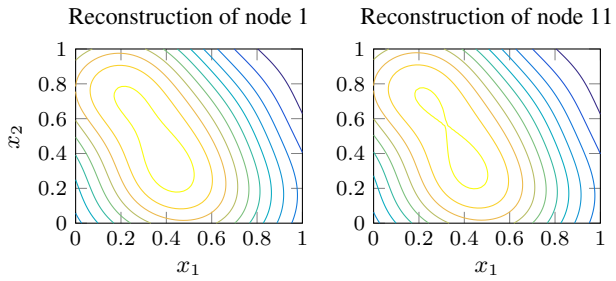

Fig. 3. Reconstruction by D-CHYPASS (I) with one global kernel of bandwidth $\zeta_{\mathrm{G},1} = 0.04$.

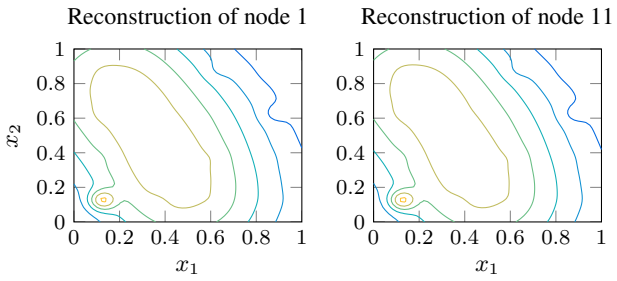

Fig. 4. Reconstruction by D-CHYPASS (II) with two global kernels of bandwidths $\zeta_{\mathrm{G},1} = 0.3, \zeta_{\mathrm{G},2} = 0.04$.

nodes where the current position of each node serves as input sample $\boldsymbol{x}_{j,k}$. We separate the unit-square into 16 equal square regions with center points $\{\boldsymbol{c}_j\}_{j=1}^J$. Each node is assigned to one region where it is placed randomly per time k following a uniform distribution. Thus, per time k the nodes change their position within their region and take a new measurement $d_{j,k}$. Nodes with a distance less than $D < 0.3$ to each other are considered as neighbors. For the HD-CHYPASS we use a Gaussian kernel for both global and local kernel with bandwidths $\zeta_{\mathrm{G}} = 0.3$ and $\zeta_{\mathrm{L}} = 0.04$, respectively. The global dictionary contains the center points of all regions, i.e. $\mathcal{D}^{\mathrm{G}} = \{\kappa_{\mathrm{G}}(\cdot, \boldsymbol{c}_j)\}_{j=1}^J$. For the local dictionary learning we set $\tau = 0.1$ and $\varepsilon = 0.3$. For comparison, we use D-CHYPASS [17] which employs a global dictionary only but else is equivalent to the HD-CHYPASS. D-CHYPASS (I) uses one global kernel with bandwidth $\zeta_{\mathrm{G},1} = 0.3$ while D-CHYPASS (II) uses two global kernels with $\zeta_{\mathrm{G},1} = 0.3, \zeta_{\mathrm{G},2} = 0.04$ where each dictionary contains the center points, i.e. $\mathcal{D}^{\mathrm{G},1} = \{\kappa_{\mathrm{G},1}(\cdot, \boldsymbol{c}_j)\}_{j=1}^J$ and $\mathcal{D}^{\mathrm{G},2} = \{\kappa_{\mathrm{G},2}(\cdot, \boldsymbol{c}_j)\}_{j=1}^J$. The step size for all algorithms is $\mu = 0.1$ and we use the Metropolis-Hastings weights for $\boldsymbol{G}$ [23]. We assume a noise variance of $\sigma_n^2 = 0.1$ and average the performance over 100 trials with a new randomly generated sensor placement in each trial.

Figure 2 depicts the scenario with the true nonlinear function $\psi(\boldsymbol{x})$ and the reconstructions by three chosen nodes in the network at steady-state. Nodes 1 and 11 are located close to high frequency components of ψ. The contour plots show that the high frequency components can be recovered successfully by node 1 and node 11 while the reconstruction by node 2 is not tampered by those of the other nodes. In contrast, we observe in Figure 3 that the D-CHYPASS (I) with one kernel is not able to reconstruct the two high frequency components but gives an estimate of the global field only. On the other hand, Figure 4 shows that the D-CHYPASS (II) with two global kernels is able to recover the component at node 1, but not at node 11. Figure 5 shows the corresponding error curves over the iteration k. We compute the normalized mean square error (NMSE) per node j and time k via $\mathrm{NMSE}_{j,k} = \mathrm{E}\left\{\int_A |\psi(\boldsymbol{x}) - \boldsymbol{w}_{j,k}^{\mathsf{T}}\boldsymbol{\kappa}(\boldsymbol{x})|^2 d\boldsymbol{x}\right\} / \int_A |\psi(\boldsymbol{x})|^2 d\boldsymbol{x}$ over the specified area A. The upper left graph illustrates the *global network performance* where the NMSE is evaluated over the whole unit-square area and averaged over all nodes. The remaining plots show the *local performance* of the three nodes. Here, the NMSE is evaluated over the node-specific region only, i.e. one square area. We can see that with respect to the overall network performance both the D-CHYPASS (I) and (II) outperform the HD-CHYPASS. This is due to averaging the performance over all nodes by which the local improvement by nodes 1 and 11 are not visible anymore. However, regarding the local error performance we can observe for

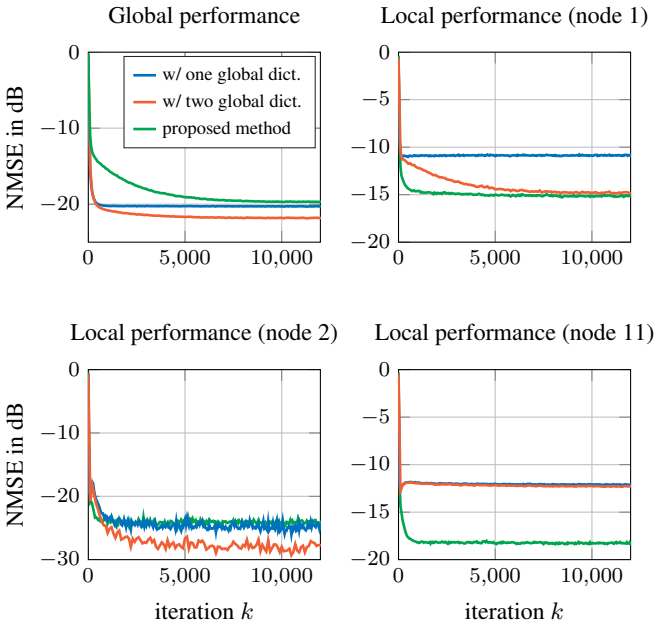

Fig. 5. Global performance of the network over the whole area and local performances of node 1, 2 and 11 in their specific region. The green and red curve use the same set of kernels, where the green curve utilizes the hybrid approach.

node 1 and node 11 that the HD-CHYPASS clearly outperforms the other algorithms. Especially for node 11 the local reconstruction performance is significantly improved by the local dictionary. At node 2 all algorithms perform similarly well since no high frequency component is present in this area.

5. CONCLUSION

We proposed a novel scheme using a hybrid dictionary to reconstruct a nonlinear function by a network of nodes. We extended the CHYPASS algorithm by consensus averaging to develop a distributed scheme and utilized two kernel functions for global and local reconstruction. We could observe that high frequency components in the local regions of nodes can be reconstructed accurately by our approach outperforming common schemes. In future work, the local dictionary stage should be extended by multiple kernels with a selection mode choosing the kernel function with the best fit.

6. REFERENCES

[1] Ban-Sok Shin, Henning Paul, Masahiro Yukawa, and Armin Dekorsy, "Distributed nonlinear regression with multiple gaussian kernels," in *SPAWC*, 2017.

[2] Ban-Sok Shin, Henning Paul, and Armin Dekorsy, "Distributed kernel least squares for nonlinear regression applied to sensor networks," in *EUSIPCO*, 2016.

[3] Pantelis Bouboulis, Symeon Chouvardas, and Sergios Theodoridis, "Online distributed learning over networks in RKH spaces using random fourier features," *arXiv:1703.08131 [cs.LG]*, 2017.

[4] Wei Gao, Jie Chen, Cédric Richard, and Jianguo Huang, "Diffusion adaptation over networks with kernel least-mean-square," in *CAMSAP*, 2015.

[5] Symeon Chouvardas and Moez Draief, "A diffusion kernel LMS algorithm for nonlinear adaptive networks," in *ICASSP*, 2016.

[6] Pedro A Forero, Alfonso Cano, and Georgios B. Giannakis, "Consensus-based distributed support vector machines," *Journal of Machine Learning Research*, vol. 11, 2010.

[7] Paul Honeine, Cédric Richard, José Carlos M. Bermudez, Hichem Snoussi, Mehdi Essoloh, and Francois Vincent, "Functional estimation in Hilbert space for distributed learning in wireless sensor networks," in *ICASSP*, 2009.

[8] Paul Honeine, Cédric Richard, José Carlos M. Bermudez, Jie Chen, and Hichem Snoussi, "A decentralized approach for nonlinear prediction of time series data in sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2010, no. 1, 2010.

[9] John Murray-Bruce and Pier Luigi Dragotti, "Reconstructing non-point sources of diffusion fields using sensor measurements," in *ICASSP*, 2016.

[10] Slobodan N. Simic and Shankar Sastry, "Distributed environmental monitoring using random sensor networks," in *Int. Workshop on Information Processing in Sensor Networks*, 2003.

[11] Masahiro Yukawa, "Adaptive learning in cartesian product of reproducing kernel hilbert spaces," *IEEE Transactions on Signal Processing*, vol. 63, no. 22, 2015.

[12] Bernhard Schölkopf and Alexander J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press, 2002.

[13] Masa-aki Takizawa and Masahiro Yukawa, "Efficient Dictionary-Refining Kernel Adaptive Filter With Fundamental Insights," *IEEE Transactions on Signal Processing*, vol. 64, no. 16, 2016.

[14] Wei Gao, Jie Chen, Cédric Richard, Jianguo Huang, and Remi Flamary, "Kernel LMS algorithm with forward-backward splitting for dictionary learning," in *ICASSP*, 2013.

[15] Weifeng Liu, José C. Príncipe, and Simon Haykin, *Kernel adaptive filtering*, John Wiley & Sons, 2010.

[16] Cédric Richard, José Carlos M. Bermudez, and Paul Honeine, "Online prediction of time series data with kernels," *IEEE Transactions on Signal Processing*, vol. 57, no. 3, 2009.

[17] Ban-Sok Shin, Masahiro Yukawa, Renato L.G. Cavalcante, and Armin Dekorsy, "Distributed adaptive learning with multiple kernels in diffusion networks," *arXiv:1801.07087 [eess.SP]*, 2018.

[18] Masahiro Yukawa, Konstantinos Slavakis, and Isao Yamada, "Adaptive parallel quadratic-metric projection algorithms," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 5, 2007.

[19] Masahiro Yukawa and Ryu-ichiro Ishii, "An efficient kernel adaptive filtering algorithm using hyperplane projection along affine subspace," in *EUSIPCO*, 2012.

[20] Henry Stark and Yongyi Yang, *Vector space projections: a numerical approach to signal and image processing, neural nets, and optics*, Wiley, 1998.

[21] Renato L. G. Cavalcante, Isao Yamada, and Bernard Mulgrew, "An adaptive projected subgradient approach to learning in diffusion networks," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, 2009.

[22] John Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, no. 2, 1991.

[23] Lin Xiao, Stephen Boyd, and Seung-Jean Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, 2007.