SPARSE BAYESIAN CONSENSUS-BASED DISTRIBUTED FIELD ESTIMATION

Thomas Buchgraber[⋆] *Dmitriy Shutin*[†]

[⋆] Signal Processing and Speech Comm. Lab., Graz University of Technology, Austria
[†] Department of Electrical Engineering, Princeton University, USA

ABSTRACT

We present a fully decentralized algorithm that is inspired by sparse Bayesian learning (SBL) and can be used for non-parametric sparse estimation of unknown spatial functions – spatial fields – with wireless sensor networks (WSNs). It is assumed that a spatial field is represented as a linear combination of weighted fixed basis functions. By exploiting the similarity between the topology of a WSN and the proposed probabilistic graphical model for distributed SBL, a combination of variational inference and loopy belief propagation (LBP) is used to obtain the weights and the sparse subset of relevant basis functions. The algorithm requires only transmission between neighboring sensors and no multi-hop communication is needed. Furthermore, it does not rely on a fixed network structure and no information about the total number of sensors in the network is necessary. Due to consensus in the weight parameters between neighboring sensors, it is demonstrated that also the sparsity patterns of relevant basis functions generally agree. The effectiveness of the proposed algorithm is demonstrated with synthetic data.

Index Terms— Distributed, variational, sparse Bayesian, message passing, consensus

1. INTRODUCTION

In recent years, the advances in electronics and digital communications have made wireless sensor networks (WSN) a very promising tool for efficiently solving large-scale decision and information-processing tasks [1, 2]. Due to energy constraints and often limited communication capabilities, the operation of WSNs rely on distributed processing, when the aim of the whole network is achieved through the synergy of individual sensors, able to sense, compute, and communicate data.

When the models of the observed phenomenon are known or can be assumed as known, the WSN processing tasks often reduce to the problem of decentralized estimation or detection (see e.g. [3] and references therein). However, when little or no *a priori* information about the measured process is available, or when the observed data is sparse, it is often more efficient to construct the model from the data itself [4]. In this paper we are interested in the latter scenario.

Consider a WSN in the form of an undirected connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ which consists of vertices (or sensors) $\mathcal{V}$ and edges (or links) $\mathcal{E}$ defined as a set of unordered pairs $\{k, l\} \subset \mathcal{V}$ representing the communication links. Note that by defining an undirected graph, we implicitly assume that sensors can communicate in both directions along a communication link. We define the neighborhood of a sensor k as $N(k) = \{l \mid \{k, l\} \in \mathcal{E}\}$ and the total number of sensors in the network as $K = |\mathcal{V}|$.

In our work, we assume that each sensor k observes a noisy measurement t_k of the field function $f(\boldsymbol{x})$ at the sensor's position $\boldsymbol{x}_k$, where $\boldsymbol{x}_k \in \mathbb{R}^d$ are the coordinates of the sensor with respect to an arbitrary d-dimensional coordinate system (e.g., d = 2 for a planar deployment). The additive noise is assumed to be zero-mean Gaussian with known variance σ^2 and we model the unknown underlying field function as

$$f(\boldsymbol{x}) = \sum_{m=1}^{M} w_m \psi_m(\boldsymbol{x}), \tag{1}$$

i.e., a superposition of M weighted fixed basis functions ψ_m. It is typically desired that (1) is as compact as possible, thus we would like to find the sparsest representation with the smallest possible number of basis functions. The research of sparse signal representation has been very intensive in recent years (see e.g. [5, 6, 7]). One of the approaches for finding sparse models, which lays down the foundation for this work, is based on sparse Bayesian learning (SBL), exemplified by relevance vector machines (RVMs) [8, 7, 9, 10].

SBL is built on a probabilistic model that achieves posterior distributions highly peaked at zero for irrelevant weights. This is achieved by introducing a parametric prior $p(w_m|\alpha_m)$ which is chosen to be a symmetric probability density function (pdf) with the parameter α_m that is inversely proportional to the pdf width. Observe, that a large value of α_m will result in the prior for w_m being highly peaked at zero; as a result, such priors will favor solutions with a few nonzero coefficients. SBL then estimates both, the model parameters as well as the prior parameters α_m for all basis functions using

This work was supported in part by the Austrian Science Fund (FWF) under Award S10610-N13 within the national research network SISE and in part by an Erwin Schrödinger Postdoctoral Fellowship, FWF J2909-N23.

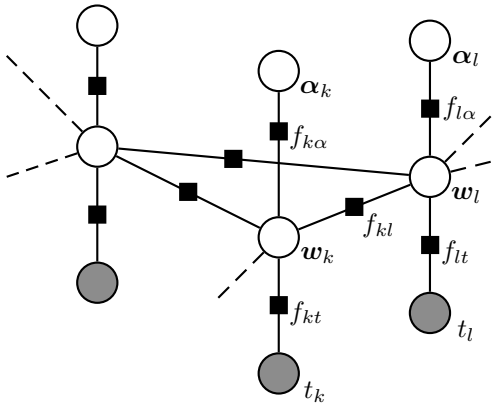


Fig. 1. A factor graph representing the sparse field estimation model. Unshaded nodes refer to hidden random variables whereas shaded nodes are observed. The factor functions are shown as black filled squares.

an approach known as the evidence procedure [9]; alternatively, approximative variational techniques can also be used [10].

To the best of our knowledge, applications of SBL for field estimation in distributed WSNs have not yet been explored in the literature. Thus, our goal in this work is to fill this gap by proposing a distributed alteration of SBL for sparse reconstruction of spatial phenomena. Note that even though the proposed algorithm is descended from standard SBL, the probabilistic model is different. This algorithm allows each sensor in the network to construct its own sparse representation of the global field. We demonstrate that by properly modifying loopy belief propagation, which has been shown to perform well for distributed inference [11, 12], a consensus between different sensors on the global sparse field representation is achieved. Such consensus-based distributed algorithms require only transmission between neighboring sensors and no multi-hop communication is needed. Furthermore in [11] it is shown that consensus algorithms are very robust against sensor and link failure as well as against poor synchronization.

The rest of the paper is organized as follows. In Section 2 we outline the probabilistic structure of the inference problem; in Section 3 the distributed learning algorithm based on variational inference and loopy belief propagation (LBP) is presented; Section 4 shows the experimental results and finally we conclude and give an outlook in Section 5.

2. BAYESIAN MODEL DEFINITION

With an independent and identically distributed Gaussian noise assumption, we can write the distribution of the observations at each sensor k as $p(t_k|\boldsymbol{w}) = \mathcal{N}(t_k|\boldsymbol{\phi}_k^T\boldsymbol{w}, \sigma^2)$, which is a likelihood function over $\boldsymbol{w}$, where we have used $\boldsymbol{w} = [w_1, \ldots, w_M]^T$ and $\boldsymbol{\phi}_k = [\psi_1(\boldsymbol{x}_k), \ldots, \psi_M(\boldsymbol{x}_k)]^T$. In SBL, the weights are modeled as random variables with hierarchical prior $p(\boldsymbol{w}|\boldsymbol{\alpha}) = \mathcal{N}(\boldsymbol{w}|\boldsymbol{0}, \mathrm{diag}(\boldsymbol{\alpha})^{-1})$, where $\mathrm{diag}(\boldsymbol{\alpha})$ is a diagonal precision matrix with hyperparameters $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_M]^T$. These hyperparameters are also defined *a priori* using a conjugate prior in the form of a Gamma distribution $p(\alpha_m) = \mathrm{Ga}(\alpha_m|a, b) \propto \alpha_m^{a-1}e^{-b\alpha_m}$. With parameterization $a = b = 0$, an improper Jeffreys non-informative prior $p(\alpha_m) \propto 1/\alpha_m$ is obtained, which results in SBL with automatic relevance determination (ARD)[1]. To solve the SBL problem, one must apply statistical inference techniques to obtain the posterior $p(\boldsymbol{w}, \boldsymbol{\alpha}|\boldsymbol{t})$, where $\boldsymbol{t} = [t_1, \ldots, t_K]^T$. Unfortunately, obtaining the posterior in closed form is intractable and approximations have to be used [9, 10]. If the posterior over the hyperparameters $\boldsymbol{\alpha}$ has most of the probability mass concentrated at large values of $\boldsymbol{\alpha}$, the corresponding posterior over the weights will be strongly peaked at zero. This fact makes the corresponding basis functions irrelevant in model (1), as they are weighted by values very close to zero. Thus, a sparse representation is achieved by pruning them from the model. In the following, we define our proposed distributed SBL model based on a factor graph representation, where each sensor k is considered to have its own weight and hyperparameter vector.

Consider a probabilistic model given by the factor graph presented in Figure 1. A factor graph is an undirected bipartite graph consisting of nodes (circles), representing random variables and factors (black boxes), representing functions of the variables they are connected to. Inference on a factor graph can be done efficiently using the sum-product algorithm which can be seen as a generalization of belief propagation [13]. The graph consists of three layers, the upper hyperlayer including all vectors $\{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_K\}$, the middle weight-layer with vectors $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K\}$ and the lower observation-layer $\{t_1, \ldots, t_K\}$, where $\boldsymbol{w}_k = [w_{k,1}, \ldots, w_{k,M}]^T$, $\boldsymbol{\alpha}_k = [\alpha_{k,1}, \ldots, \alpha_{k,M}]^T$ and t_k are the locally accessible variables at sensor k. As depicted in Figure 1, the sensors only share information on the weight-layer via the factors f_{kl}. These are defined according to the link structure of the WSN, i.e., $\{f_{kl}|\{k, l\} \in \mathcal{E}\}$.

It should be emphasized that the factor graph presented in Figure 1 is different from a centralized SBL graphical model. Specifically, there are several local SBL models with individual observation t_k. These SBL models are connected to each other according to the actual topology[2] $\mathcal{G}$ of the WSN.

We define the factors at each sensor k as[3]

$$f_{k\alpha}(\boldsymbol{w}_k, \boldsymbol{\alpha}_k) = |\mathbf{A}_k|^{a-\frac{1}{2}} \exp\left\{-\frac{1}{2}\boldsymbol{w}_k^T\mathbf{A}_k\boldsymbol{w}_k - b\cdot\mathrm{tr}(\mathbf{A}_k)\right\} \tag{2}$$

[1]Note that even though the prior is improper in this case, the posterior is a proper distribution.

[2]It is assumed that the physical and logical topologies coinside.

[3]Note that $f_{k\alpha}(\boldsymbol{w}_k, \boldsymbol{\alpha}_k) \propto p(\boldsymbol{w}_k|\boldsymbol{\alpha}_k)p(\boldsymbol{\alpha}_k)$, the SBL joint prior.

and

$$f_{kt}(\boldsymbol{w}_k, \tilde{t}_k) = \exp\left\{-\frac{1}{2\sigma^2}\left(\tilde{t}_k - \boldsymbol{\phi}_k^T \boldsymbol{w}_k\right)^2\right\}, \quad (3)$$

where $\mathbf{A}_k = \mathrm{diag}(\boldsymbol{\alpha}_k)$. To share information between neighbors, we define the coupling factors (also called compatibility functions [12]) as

$$f_{kl}(\boldsymbol{w}_k, \boldsymbol{w}_l) = \exp\left\{-\frac{\beta}{2}\|\boldsymbol{w}_k - \boldsymbol{w}_l\|_2^2\right\} \quad (4)$$

between the nodes $\boldsymbol{w}_k$ and $\boldsymbol{w}_l$ for $\{k, l\} \in \mathcal{E}$, where β is the coupling parameter. Large β values result in a strong consensus between the weight distributions of connected nodes, whereas small values make the coupling loose.

3. VARIATIONAL APPROXIMATION AND MESSAGE PASSING

Consider the global vectors $\bar{\boldsymbol{w}} = [\boldsymbol{w}_1^T, \ldots, \boldsymbol{w}_K^T]^T$ and $\bar{\boldsymbol{\alpha}} = [\boldsymbol{\alpha}_1^T, \ldots, \boldsymbol{\alpha}_K^T]^T$ which collect the corresponding local vectors from the network. Inference on the graph presented in Figure 1 corresponds to the computation of the posterior distribution $p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}|\boldsymbol{t}} = p(\bar{\boldsymbol{w}}, \bar{\boldsymbol{\alpha}}|\boldsymbol{t})$ of the hidden variables (unshaded nodes) given the observed variables (shaded nodes). Since the posterior can not be obtained in closed form, we define an approximate posterior distribution $q_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}}$ as a function of all hidden variables. By applying a structured mean field approximation, we assume the factorization

$$q_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}} = q(\bar{\boldsymbol{w}}) \prod_{k=1}^{K} q(\boldsymbol{\alpha}_k) \quad (5)$$

of the proxy distribution, where we generally – unlike in the posterior – omit to write the condition on the observations $\boldsymbol{t}$ in the $q(\cdot)$ distributions for convenience. Variational inference now iteratively optimizes each factor in (5) by minimizing the Kullback-Leibler divergence $KL(q_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}}||p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}|\boldsymbol{t}})$ which can not be performed directly because of the intractable posterior. Instead, we equivalently maximize the variational lower bound [14] on the log-evidence $\ln p(\boldsymbol{t})$:

$$\ln p(\boldsymbol{t}) \geq \mathcal{L}(q_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}}) = \int q_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}} \ln \frac{p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}\boldsymbol{t}}}{q_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}}} \, d\bar{\boldsymbol{w}} d\bar{\boldsymbol{\alpha}}, \quad (6)$$

which is a functional of the proxy posterior $q_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}}$ and the joint distribution $p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}\boldsymbol{t}} = p(\bar{\boldsymbol{w}}, \bar{\boldsymbol{\alpha}}, \boldsymbol{t})$. Since the joint distribution $p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}\boldsymbol{t}}$ is tractable instead of the posterior $p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}|\boldsymbol{t}}$, we can now solve the optimization problem. We define the distributions of the proxy factors given in (5) as

$$q(\boldsymbol{\alpha}_k) = \prod_{m=1}^{M} \mathrm{Ga}(\alpha_{k,m}|\hat{a}_{k,m}, \hat{b}_{k,m}) \quad (7)$$

and

$$q(\bar{\boldsymbol{w}}) = \mathcal{N}(\bar{\boldsymbol{w}}|\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Lambda}}^{-1}), \quad (8)$$

where $\hat{\boldsymbol{\Lambda}}$ is a precision matrix. We also should find it useful later, to define the marginal proxy weight posterior distributions at a single sensor k as

$$q(\boldsymbol{w}_k) = \int q(\bar{\boldsymbol{w}}) \, d\bar{\boldsymbol{w}}_{\sim k} = \mathcal{N}(\boldsymbol{w}_k|\hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Lambda}}_k^{-1}), \quad (9)$$

where $\bar{\boldsymbol{w}}_{\sim k}$ means a vector of all weights $\bar{\boldsymbol{w}}$ without the elements of $\boldsymbol{w}_k$.

By applying variational calculus, we obtain the logarithm of the optimal update [14] of factor $q(\boldsymbol{\alpha}_k)$ with

$$\ln q^*(\boldsymbol{\alpha}_k) = \mathbb{E}_{\sim q(\boldsymbol{\alpha}_k)}\{\ln p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}\boldsymbol{t}}\} + const., \quad (10)$$

where $\mathbb{E}_{\sim q(\boldsymbol{\alpha}_k)}\{\cdot\}$ denotes the expectation with respect to all other factors in Equation (5) instead of $q(\boldsymbol{\alpha}_k)$ and the term '$const.$' includes the terms independent of $\boldsymbol{\alpha}_k$, thus that $q^*(\boldsymbol{\alpha}_k)$ is a valid normalized distribution. Knowing that the joint distribution represented by a factor graph is the normalized product of all the factors in the graph, we can write

$$p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}\boldsymbol{t}} = \frac{1}{Z}\left(\prod_{\{i,j\}\in\mathcal{E}} f_{ij}\right) \prod_{u=1}^{K} f_{u\alpha} f_{ut}, \quad (11)$$

where Z is a normalization factor. By inserting (11) into (10), we obtain

$$\ln q^*(\boldsymbol{\alpha}_k) = \mathbb{E}_{q(\boldsymbol{w}_k)}\{\ln f_{k\alpha}(\boldsymbol{w}_k, \boldsymbol{\alpha}_k)\} + const., \quad (12)$$

where all other factors (including the normalization) were absorbed into the '$const.$' term and only the factor $f_{k\alpha}$ as a function of $\boldsymbol{\alpha}_k$ remains in the expectation. We should also notice that the expectation in (12) now is only with respect to $q(\boldsymbol{w}_k)$, since we can integrate out all other terms. This is also because $\boldsymbol{w}_k$ is the only variable in the Markov blanket (cf. [14]) of $\boldsymbol{\alpha}_k$ as can be seen in Figure 1. Inserting (2) into (12) and solving for $q^*(\boldsymbol{\alpha}_k)$ results in a product of Gamma distributions equal as defined in (7). Thus our optimal update distribution $q(\boldsymbol{\alpha}_k) = q^*(\boldsymbol{\alpha}_k)$ can be achieved with (7) and for $m = 1, \ldots, M$ we get

$$\hat{a}_{k,m} = a + 1/2 \quad (13)$$

and

$$\hat{b}_{k,m} = b + (\hat{\mu}_{k,m}^2 + \hat{\Sigma}_{k,mm})/2, \quad (14)$$

where $\hat{\mu}_{k,m}$ is the m-th element of $\hat{\boldsymbol{\mu}}_k$ and $\hat{\Sigma}_{k,mm}$ is the m-th main diagonal element of the marginal covariance matrix $\hat{\boldsymbol{\Sigma}}_k = \hat{\boldsymbol{\Lambda}}_k^{-1}$. Updating the factors $q(\boldsymbol{\alpha}_k)$ means computing (13) and (14) for all sensors k and basis functions m which can be done locally at sensor k if it can access the parameters of the proxy weight marginal $q(\boldsymbol{w}_k)$.

By again using variational calculus [14], the logarithm of the optimal global weight proxy is defined as

$$\ln q^*(\bar{\boldsymbol{w}}) = \mathbb{E}_{\sim q(\bar{\boldsymbol{w}})}\{\ln p_{\bar{\boldsymbol{w}}\bar{\boldsymbol{\alpha}}\boldsymbol{t}}\} + const., \quad (15)$$

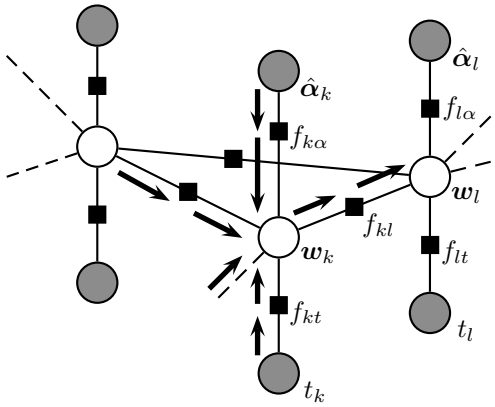


Fig. 2. Representation of the messages needed to be passed through the WSN to obtain the marginal weight proxy distributions.

and becomes

$$\ln q^*(\bar{\boldsymbol{w}}) = \sum_{k=1}^{K} \Big(\mathbb{E}_{q(\boldsymbol{\alpha}_k)} \big\{ \ln f_{k\alpha}(\boldsymbol{w}_k, \boldsymbol{\alpha}_k) \big\} + \ln f_{kt}(\boldsymbol{w}_k, \tilde{t}_k) \Big) + \sum_{\{i,j\} \in \mathcal{E}} \ln f_{ij}(\boldsymbol{w}_i, \boldsymbol{w}_j) + const. \tag{16}$$

by inserting (11) for the joint distribution. From (2), it is easy to see that

$$\mathbb{E}_{q(\boldsymbol{\alpha}_k)} \big\{ \ln f_{k\alpha}(\boldsymbol{w}_k, \boldsymbol{\alpha}_k) \big\} = \ln f_{k\alpha}(\boldsymbol{w}_k, \hat{\boldsymbol{\alpha}}_k) + const. \tag{17}$$

if we absorb all parts which are not a function of $\bar{\boldsymbol{w}}$ into the '$const.$' term and define $\hat{\boldsymbol{\alpha}}_k = \mathbb{E}_{q(\boldsymbol{\alpha}_k)}\{\boldsymbol{\alpha}_k\}$, where the elements of $\hat{\boldsymbol{\alpha}}_k$ can be obtained from (13) and (14) using the general known expectation result of a Gamma distribution, i.e., $\hat{\alpha}_{k,m} = \hat{a}_{k,m}/\hat{b}_{k,m}$. This is an interesting result, because it makes the factor $f_{k\alpha}(\boldsymbol{w}_k, \hat{\boldsymbol{\alpha}}_k)$ proportional to a Gaussian $\mathcal{N}(\boldsymbol{w}_k|0, \hat{\mathbf{A}}_k^{-1})$ with known precision matrix $\hat{\mathbf{A}}_k = \mathrm{diag}(\hat{\boldsymbol{\alpha}}_k)$ and allows us to interpret $q^*(\bar{\boldsymbol{w}})$ as being proportional to the joint distribution (11) with observed $\boldsymbol{\alpha}_k$ values set to $\boldsymbol{\alpha}_k = \hat{\boldsymbol{\alpha}}_k$. Because all factors are now Gaussians, $q^*(\bar{\boldsymbol{w}})$ must also be a Gaussian, and with definition (8) we see that $q(\bar{\boldsymbol{w}}) = q^*(\bar{\boldsymbol{w}})$, the optimal approximation can be achieved. Instead of computing the Gauss parameters $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\Lambda}}$ of (8), we are only interested in the marginal parameters $\hat{\boldsymbol{\mu}}_k$ and $\hat{\boldsymbol{\Lambda}}_k$ for two reasons. First, the local hyperparameter expectation updates $\hat{\boldsymbol{\alpha}}_k$ just depend on the marginal parameters through (14). And secondly, the marginals can be computed distributively using Gaussian believe propagation described in the following with the sum-product algorithm.

Consider the factor graph presented in Figure 2, where the graph of Figure 1 is modified in such a way, that like for the observations t_k, also all $\boldsymbol{\alpha}_k$ are observed and have values $\hat{\boldsymbol{\alpha}}_k$. We now can start to pass messages starting at the leaves of the graph. Applying the sum-product algorithm, we send the first messages $m_{\boldsymbol{\alpha}_k \to f_{k\alpha}}$ from the nodes $\boldsymbol{\alpha}_k, \forall k$ to the factors $f_{k\alpha}(\boldsymbol{w}_k, \boldsymbol{\alpha}_k)$ which are, since the nodes are observed, Dirac delta functions $\delta(\boldsymbol{\alpha}_k - \hat{\boldsymbol{\alpha}}_k)$. The following messages $m_{f_{k\alpha} \to \boldsymbol{w}_k}$ are given by the integrals over $\boldsymbol{\alpha}_k$ of the incoming messages times the factors, which gives[4] $f_{k\alpha}(\boldsymbol{w}_k, \hat{\boldsymbol{\alpha}}_k)/Z$. Likewise, we obtain $m_{\tilde{t}_k \to f_{kt}} = \delta(\tilde{t}_k - t_k)$ and $m_{f_{kt} \to \boldsymbol{w}_k} = f_{kt}(\boldsymbol{w}_k, t_k)/Z$, where $\tilde{t}_k$ denotes all possible outcomes for the observations. For the messages between the sensors, we make use of the knowledge that all involved distributions are Gaussian and thus define the incoming messages at node $\boldsymbol{w}_k$, stemming from an arbitrary neighbor u, as general Gaussians over $\boldsymbol{w}_k$ with

$$m_{f_{uk} \to \boldsymbol{w}_k} = \frac{1}{Z} \exp \left\{ -\frac{1}{2} (\boldsymbol{w}_k - \boldsymbol{\mu}_{uk})^T \boldsymbol{\Lambda}_{uk} (\boldsymbol{w}_k - \boldsymbol{\mu}_{uk}) \right\}. \tag{18}$$

The marginal distribution over $\boldsymbol{w}_k$ can be computed as the normalized product of all incoming messages

$$\tilde{q}(\boldsymbol{w}_k) = \frac{1}{Z}\, m_{f_{k\alpha} \to \boldsymbol{w}_k}\, m_{f_{kt} \to \boldsymbol{w}_k} \prod_{u \in N(k)} m_{f_{uk} \to \boldsymbol{w}_k}, \tag{19}$$

where we have used the symbol $\tilde{q}$ instead of q to highlight the difference to the real marginal weight proxy $q(\boldsymbol{w}_k)$ which can only be obtained after several message iterations if the network forms a loopy graph as will be discussed in Section 3.1. We define $\tilde{q}(\boldsymbol{w}_k)$ to be a Gaussian $\tilde{q}(\boldsymbol{w}_k) = \mathcal{N}(\boldsymbol{w}_k|\boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k^{-1})$ with parameters

$$\boldsymbol{\Lambda}_k = \hat{\mathbf{A}}_k + \sigma^{-2} \boldsymbol{\phi}_k \boldsymbol{\phi}_k^T + \sum_{u \in N(k)} \boldsymbol{\Lambda}_{uk} \tag{20}$$

and

$$\boldsymbol{\mu}_k = \boldsymbol{\Lambda}_k^{-1} \Big(\sigma^{-2} \boldsymbol{\phi}_k t_k + \sum_{u \in N(k)} \boldsymbol{\Lambda}_{uk} \boldsymbol{\mu}_{uk} \Big). \tag{21}$$

To compute the messages to a neighboring sensor, we need all local and incoming messages from the other neighbors as depicted in Figure 2. A message sent from $\boldsymbol{w}_k$ to a neighboring factor f_{kl} can be defined as

$$m_{\boldsymbol{w}_k \to f_{kl}} = \frac{1}{Z}\, m_{f_{kt} \to \boldsymbol{w}_k}\, m_{f_{k\alpha} \to \boldsymbol{w}_k} \prod_{u \in N(k) \setminus l} m_{f_{uk} \to \boldsymbol{w}_k}, \tag{22}$$

the product of the local messages and messages from all neighbors other than l. The neighbor l then obtains its incoming message from k with

$$m_{f_{kl} \to \boldsymbol{w}_l} = \frac{1}{Z} \int f_{kl}(\boldsymbol{w}_k, \boldsymbol{w}_l)\, m_{\boldsymbol{w}_k \to f_{kl}}\, d\boldsymbol{w}_k, \tag{23}$$

which is again of the form (18) and has precision matrix and

[4]Note that throughout this paper we assume all messages to be normalized, which is no restriction and allows us to directly interpret the messages as Gaussians.

mean vector given by

$$\mathbf{\Lambda}_{kl} = \left(\left(\hat{\mathbf{A}}_k + \sigma^{-2} \boldsymbol{\phi}_k \boldsymbol{\phi}_k^T + \sum_{u \in N(k) \setminus l} \mathbf{\Lambda}_{uk} \right)^{-1} + \beta^{-1} \mathbf{I} \right)^{-1} \tag{24}$$

and

$$\boldsymbol{\mu}_{kl} = (\mathbf{\Lambda}_{kl}^{-1} - \beta^{-1} \mathbf{I}) \left(\sigma^{-2} \boldsymbol{\phi}_k t_k + \sum_{u \in N(k) \setminus l} \mathbf{\Lambda}_{uk} \boldsymbol{\mu}_{uk} \right) \tag{25}$$

respectively. For more information on factor graphs and the sum-product algorithm, the reader is referred to [15].

3.1. Loopy Belief Propagation and its convergence

For nearly all realistic scenarios, the graph $\mathcal{G}$ of a WSN is a loopy one, i.e., messages can pass along paths for multiple times. Since the graph structure $\mathcal{G}$ can be found on the weight-layer of the factor graph depicted in Figure 2, we perform message passing on such a loopy graph when determining the marginal weight proxys $q(\boldsymbol{w}_k)$. Messages have to be passed through the network several times until convergence. For this work, we assume a synchronized scheduling for convenience, even though as [11] suggests, LBP on an asynchronous and even dynamically changing graph generally gives good results. From (20) and (21) we can define the marginal parameters at sensor k with update index n as

$$\mathbf{\Lambda}_k^{(n)} = \hat{\mathbf{A}}_k + \sigma^{-2} \boldsymbol{\phi}_k \boldsymbol{\phi}_k^T + \sum_{u \in N(k)} \mathbf{\Lambda}_{uk}^{(n-1)} \tag{26}$$

and

$$\boldsymbol{\mu}_k^{(n)} = \left(\mathbf{\Lambda}_k^{(n)} \right)^{-1} \left(\sigma^{-2} \boldsymbol{\phi}_k t_k + \sum_{u \in N(k)} \mathbf{\Lambda}_{uk}^{(n-1)} \boldsymbol{\mu}_{uk}^{(n-1)} \right). \tag{27}$$

The messages from node k to node l at update n can be defined as

$$\mathbf{\Lambda}_{kl}^{(n)} = \left(\left(\mathbf{\Lambda}_k^{(n)} - \mathbf{\Lambda}_{lk}^{(n-1)} \right)^{-1} + \beta^{-1} \mathbf{I} \right)^{-1} \tag{28}$$

and

$$\boldsymbol{\mu}_{kl}^{(n)} = \left(\mathbf{\Lambda}_k^{(n)} - \mathbf{\Lambda}_{lk}^{(n-1)} \right)^{-1} \left(\mathbf{\Lambda}_k^{(n)} \boldsymbol{\mu}_k^{(n)} - \mathbf{\Lambda}_{lk}^{(n-1)} \boldsymbol{\mu}_{lk}^{(n-1)} \right), \tag{29}$$

which is more efficient compared to (24) and (25) because we made use of the intermediate results (26) and (27), since we have to compute these messages for all neighbors. Note that the communication between the neighbors can be implemented even more efficiently using local broadcasts [16], but is not considered in this paper.

Convergence of LBP is not guaranteed in general. But there has been some research on the convergence of Gaussian LBP (e.g. [17]) defined for nodes with scalar Gaussian random variables. Even though, we did not encounter convergence problems during our simulations for the multivariate Gaussian LBP case used in our work, at the moment we have not proven its convergence. Also the relation between convergence-time and the coupling parameter β, like analyzed for the scalar case in [12], is part of future research.

After convergence[5] for $n \to \infty$, we can finally define the marginal weight proxy parameters from (26) and (27) as

$$\hat{\mathbf{\Sigma}}_k = \hat{\mathbf{\Lambda}}_k^{-1} = \left(\mathbf{\Lambda}_k^{(\infty)} \right)^{-1} \tag{30}$$

and

$$\hat{\boldsymbol{\mu}}_k = \boldsymbol{\mu}_k^{(\infty)}, \tag{31}$$

where $\hat{\mathbf{\Sigma}}_k$ is the covariance matrix used in (14).

3.2. Sparse communication and data processing

Since the energy consumption of each individual sensor grows with the computational and transmission complexity, we need to incorporate sparsity by pruning irrelevant model components. That is, we need to reduce the number of basis functions at each sensor's model (which is based on (1)) and hence also the size of the matrices and vectors in (26)-(31).

During the alternating updates of the factors in (5), some hyperparmeters $\hat{\alpha}_{k,m}$ related to irrelevant basis functions diverge. Practically, we look for divergence by checking if a hyperparameter exceeds a large predefined threshold γ_{th} (e.g. 10^{10}). If so, we assume it to be infinite. By inspecting Equation (24) for sensor k, it is easy to show that the inner inverse equation has only zeros in the m-th row and column if $\hat{\alpha}_{k,m} = \infty$. When we also consider the addition of the $\beta^{-1}\mathbf{I}$ term, after computing the outer inverse, we obtain a matrix $\mathbf{\Lambda}_{kl}$ with the m-th row and column equal to zero, except the m-th main diagonal element which is β. This can be easily shown by using matrix permutation and blockwise inversion rules. Since β is a design parameter in our model and is known by all sensors, there is no need for transmitting it to the neighbors. Thus, we can delete the m-th row and column of $\mathbf{\Lambda}_{kl}$ and transmit the resulting sparse message $\breve{\mathbf{\Lambda}}_{kl}$. The receiving sensor l, which is getting the message $\breve{\mathbf{\Lambda}}_{kl}$, can reinsert β in the diagonal and thus reproduce $\mathbf{\Lambda}_{kl}$ if it knows the correct positions where elements have been pruned. The only information we need to transmit additionally from a sensor k to l is a binary mask of size M indicating the used basis functions. Similarly, it can be shown that the mean message $\boldsymbol{\mu}_{kl}$ defined in (25) obtains zero elements at all positions m if $\hat{\alpha}_{k,m}$ diverges. After pruning the zero elements from the vector, we send the sparse mean message $\breve{\boldsymbol{\mu}}_{kl}$, which can also be reproduced at sensor l to give $\boldsymbol{\mu}_{kl}$ by reinserting zeros according to the binary mask.

It is important to mention that also all local computations can be done efficiently by taking sparsity patterns into account, i.e. to consider only the matrix and vector elements

[5] Practically, good results are obtained after a couple of iterations depending on the network size and connectivity.

corresponding to finite $\hat{\alpha}_{k,m}$ values similar to the previous discussion. For each $\hat{\alpha}_{k,m} = \infty$, the marginal covariance matrix (30) at sensor k gets zeros in the m-th rows and columns. Likewise the mean vector (30) gets zero entries at all positions m. Thus, the corresponding elements, including the hyperparameter, can be pruned from the local model.

3.3. Summary of the algorithm

The algorithm basically consists of alternating updates performed on the hyper- and weight-layer as summarized in Algorithm 1. We first start at the hyper-layer by locally initial-

Algorithm 1 Sparse Consensus-based Distributed Field Est.

Initialize $\hat{\boldsymbol{\alpha}}_k$ (e.g. $\hat{\alpha}_{k,m} = 10^{-3},\ \forall m), \forall k$.

% Variational update loop
while (Not converged) **do**
  Initialize: $\boldsymbol{\Lambda}_{uk}^{(0)} = \mathbf{0}$ and $\boldsymbol{\mu}_{uk}^{(0)} = \boldsymbol{0}$, $\forall (u,k) \in \mathcal{E}$.
  % Message passing update loop
  while (LBP not converged) **do**
    Send sparse messages (28) and (29) according to Section 3.2 through the network.
  end while
  Compute the sparse marginals (30) and (31), $\forall k$, as explained in Section 3.2.
  From (13) and (14) at each sensor k update the hyperparameter expectations $\hat{\alpha}_{k,m} = \hat{a}_{k,m}/\hat{b}_{k,m}, \forall m$.
end while
Compute the final sparse marginals (30) and (31), $\forall k$.

izing the hyperparameter expectations $\hat{\boldsymbol{\alpha}}_k$ at each sensor k. Afterwards, we perform message passing on the weight-layer according to (28) and (29), where the initial incoming messages are defined as $\boldsymbol{\Lambda}_{uk}^{(0)} = \mathbf{0}$ and $\boldsymbol{\mu}_{uk}^{(0)} = \boldsymbol{0}$. After convergence, we obtain $\hat{\boldsymbol{\Sigma}}_k$ and $\hat{\boldsymbol{\mu}}_k$ according to (30) and (31). The only communication necessary in the WSN is to obtain these two parameters at each sensor. They are then used locally for the hyper-layer update, i.e., updating $\hat{\boldsymbol{\alpha}}_k$ at each sensor k using (13) and (14). In the following updates we iterate between hyper- and weight-layer and perform sparse computation and communication according to Section 3.2 if hyperparameters diverge. After convergence of the variational proxy updates, we finally obtain a sparse marginal distribution $q(\boldsymbol{w}_k)$ at each sensor k. With the mean vector $\hat{\boldsymbol{\mu}}_k$ each sensor can reconstruct the field function as a superposition of weighted basis functions as in (1), but containing only the relevant components. Note that it is also possible to estimate the variance of the field function as in [9], but is not of importance here.

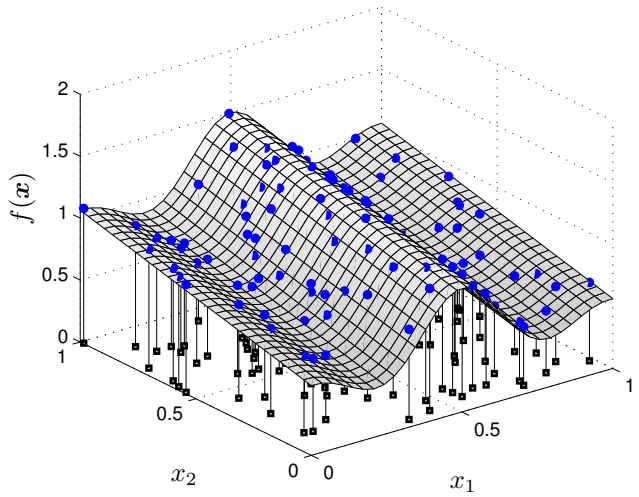


Fig. 3. True underlying field function $f(\boldsymbol{x})$ defined by (32). Squares represent sensor positions and dots the local measurements.

4. SIMULATION

For the simulations we used a constant bias $\psi_1(\boldsymbol{x}) = 1$ and Gaussian kernel basis functions $\psi_m(\boldsymbol{x}) = \kappa(\boldsymbol{x}, \boldsymbol{x}_{m-1})$ for $m = 2, \ldots, K+1$ centered on the sensor positions[6], where $\kappa(\boldsymbol{x}, \boldsymbol{x}_{m-1}) = \exp\{-\theta_\kappa ||\boldsymbol{x} - \boldsymbol{x}_{m-1}||^2\}$ with kernel parameter $\theta_\kappa = 15$. We used $K = 100$ sensors resulting in an initial model with $M = K + 1 = 101$ basis functions. We define the underlying true field function as

$$f(\boldsymbol{x}) = 0.5\,\mathrm{sinc}(x_1 - 0.5) + 0.5x_2 + 0.5, \qquad (32)$$

with dimension $d = 2$ for $\boldsymbol{x}$, as plotted in Figure 3. The sensors are deployed randomly, shown as black squares in the figure, whereas the local noisy measurements t_k, produced by adding white Gaussian noise of variance $\sigma^2 = 10^{-5}$ to $f(\boldsymbol{x}_k)$, are given as dots. The pruning threshold was set to $\gamma_{th} = 10^{10}$ and the coupling parameter in (4) is set to $\beta = 10^8$. The initial hyperparameter expectations are set as $\hat{\alpha}_{k,m} = 10^{-3}, \forall k, m$. We used the ARD case for SBL with $a = b = 0$. The network structure can be seen in Figure 4, where a connectivity of 10% relative to the total number of possible connections was used. The solid and dashed lines represent the connections between the sensors. During the variational iterations, some hyperparameters $\hat{\boldsymbol{\alpha}}_k$ reach large values as depicted in Figure 5 for two different sensors. Sensors that have diverging elements in $\hat{\boldsymbol{\alpha}}_k$, prune the corresponding basis functions and send sparse messages to their neighbors as discussed in Section 3.2. From Figure 5 we can also see, that sensors with a denser connection to the rest of the network are getting more support from their neighbors in

[6]For kernel basis functions, like used in the simulation, it is assumed that each sensor knows about the position of the others initially to be able to define the fixed basis function set. This is not necessary for general predefined fixed basis functions, where sensors need only information about their own position to compute $\boldsymbol{\phi}_k$.

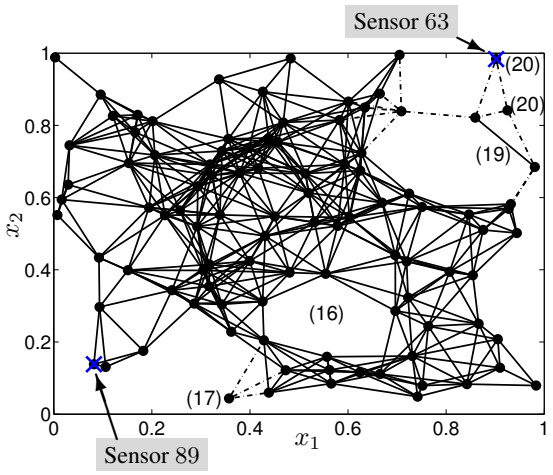


Fig. 4. WSN connections (solid and dashed lines) and number of relevant basis functions. Solid lines depict consensus in the set of basis functions between different sensors, whereas dashed lines mean that no consensus was achieved.

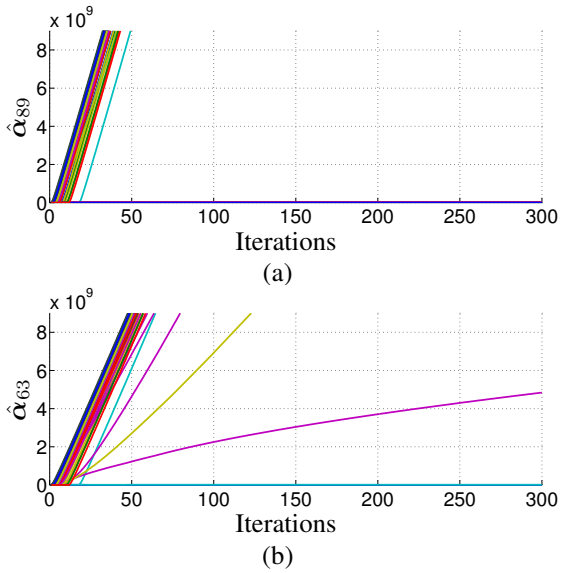


Fig. 5. Hyperparameter estimates at (a) the lower left and (b) the upper right sensor as depicted in Figure 4. Most parameters diverge whereas the others stay close to zero.

evaluating basis functions, which accelerates the pruning process. It should be stressed that compared to centralized variational SBL, the divergence-rate of the hyperparameters for distributed SBL is empirically much faster. For threshold values γ_{th} in the range used in our simulations the centralized version has significantly higher simulation times. Specifically, while the proposed distributed algorithm converges in about 2h on our used hardware, the centralized version was aborted after 1 day without converging. Both simulations where run on a single core, where also the distributed algorithm was simulated centrally. In Figure 4 we also show the number of basis functions for individual consensus clusters connected by solid lines. All the sensors in a cluster have identical binary masks, i.e., they agree on the same sparse set of basis functions. Dashed lines reveal differences between basis sets of neighboring nodes. It is clear to see that most parts of the graph reach perfect consensus in the sparse representation. Outer sensors with only a few connections are more likely to suffer from different estimates and in general have a slower convergence rate due to the lack of collaboration capabilities.

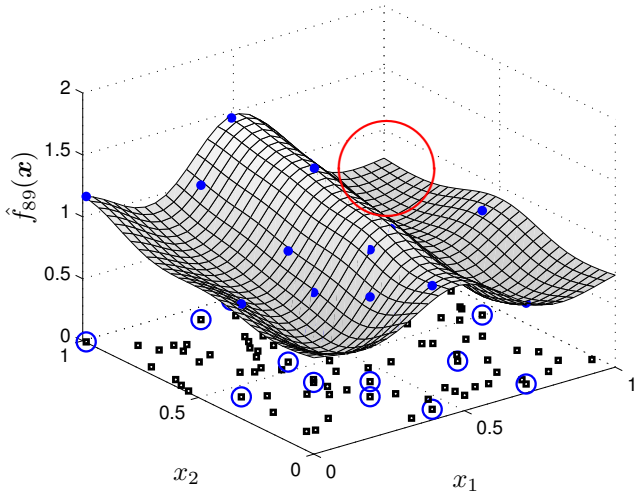
Another important observation we made when using Gaussian kernel basis functions, is that kernels with centers closer to a sensor's position are more preferred to be kept by this sensor than kernels which are further away. This can be seen in Figure 6, where the sparse estimated field of two different sensors is given. The dots show the relevance vectors [9], which correspond to kernel centers marked by circles. In Figure 6b, all kernels different to Figure 6a, are in the area of the sensor itself. This is mostly a desired feature, since for many applications sensors need more accurate information from their closer surrounding than from distant parts of the network.

Algorithm	MSE	#Bfs
proposed method	-22.08dB	16.16
DRKR	-14.07dB	16

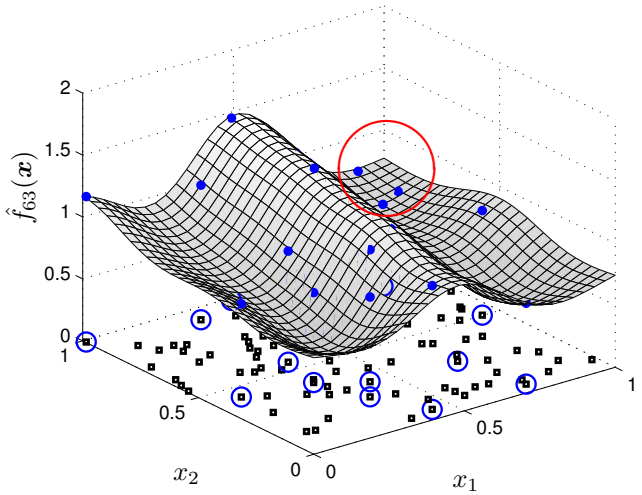
Table 1. Comparison of mean squared error (MSE) and number of basis function (#Bf) performance of the proposed method with a distributed reduced-order kernel regression algorithm (DRKR).

Finally, in Table 1 we compare the proposed method with a distributed reduced-order kernel regression algorithm (DRKR) [18]. This algorithm is not based on consensus and uses sequential message passing instead, where the last node in the chain obtains the model weights. The stepsize and threshold parameters for the DRKR method were optimized to obtain the smallest mean squared error (MSE). The performance results for our proposed method are averaged across the individual nodes. Both algorithms achieve approximately the same number of basis functions, where our proposed method achieves a 8dB better MSE[7]. This is due to the fact, that the consensus-based SBL algorithm estimates the relevant basis functions dependent on the sensor measurements $\boldsymbol{t}$ whereas DRKR uses a kernel coherence criteria which is independent of the measurements.

[7]Note that the computational complexity of our proposed consensus-algorithm is higher compared to DRKR. Instead our algorithm is robust against network changes and requires communication between neighbors only.



(a)



(b)

Fig. 6. Sparse estimated field function $\hat{f}_k(\boldsymbol{x})$ of (a) the sensor in the lower left and (b) the upper right corner of the network as depicted in Figure 4. 16 basis functions are used in (a) and 20 in (b). The dots represent the relevance vectors (RVs), whereas the small circles depict the associated relevant kernel centers. Three of the four RV differences are visible and highlighted by the big circles.

5. CONCLUSIONS AND OUTLOOK

In this work, we have presented a distributed sparse Bayesian learning (SBL) method for consensus based field estimation. The method is based on variational inference and loopy belief propagation (LBP). A field function, spatially sampled by a wireless sensor network (WSN), is represented as a sum of weighted fixed basis functions. An overcomplete basis set can be reduced to a relevant subset by using a Bayesian model that is inspired by SBL. Our robust sparsification method works fully distributed with communications only between neighboring sensors. Each sensor obtains a sparse representation of the global field function which is estimated in consensus with all others without even knowing the network size. Simulations show that densely connected regions of the network can achieve faster convergence in terms of sparsity due to stronger collaboration with neighbors. Furthermore it was shown that densely connected sensors usually achieve the same sparse subset of relevant basis functions. Compared to centralized variational SBL our method empirically converges much faster and the mean squared error performance was shown to be better than a recently proposed distributed reduced-order kernel regression method.

The convergence conditions for the used multivariate Gaussian LBP and its dependence on different message scheduling approaches should be investigated. A detailed performance analysis as a function of network size, connectivity and different coupling parameter settings is also part of future research.

Finally, it should be mentioned that the proposed algorithm has an inherent sparsification mechanism that helps sensor nodes to save energy due to the dramatic computation and transmission reduction caused by pruning many irrelevant vector and matrix elements. But especially in the initial phase, when the model is large, much resources are needed before pruning actually starts. Thus, we see adaptive models as an important research topic for sparse Bayesian distributed field estimation in the future, i.e., methods for pruning and adding basis functions as needed that start from small models.

6. REFERENCES

[1] A. Swami, Q. Zhao, Y.-W. Hong, and L. Tong, Eds., *Wireless Sensor Networks: Signal Processing and Communications*, Wiley Interscience, 2007.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," 2002.

[3] B. Chen, L. Tong, and P.K. Varshney, "Channel-aware distributed detection in wireless sensor networks," *IEEE Signal Proc. Magazine*, vol. 23, no. 4, pp. 16–26, 2006.

[4] J.B. Predd, S.B. Kulkarni, and H.V. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Proc. Magazine*, vol. 23, no. 4, pp. 56–69, 2006.

[5] R. Baraniuk, "Compressive sensing," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–121, July 2007.

[6] M.B. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[7] D. G. Tzikas, A. C. Likas, and N. P. Galatsanos, "The variational approximation for Bayesian inference," *IEEE Signal Process. Mag.*, vol. 25, no. 6, pp. 131–146, November 2008.

[8] D.P. Wipf and B.D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Trans. on Sig. Proc.*, vol. 52, no. 8, pp. 2153 – 2164, aug. 2004.

[9] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, pp. 211–244, June 2001.

[10] C. M. Bishop and M. E. Tipping, "Variational relevance vector machines," in *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, San Francisco, CA, USA, 2000, pp. 46–53.

[11] C. Crick and A. Pfeffer, "Loopy belief propagation as a basis for communication in sensor networks," in *UAI*, 2003, pp. 159–166.

[12] C.C. Moallemi and B. Van Roy, "Consensus propagation," *IEEE Transactions on Information Theory*, vol. 52, pp. 4753–4766, 2006.

[13] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[14] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer, 1st ed. 2006. corr. 2nd printing edition, October 2007.

[15] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *Information Theory, IEEE Transactions on*, vol. 47, no. 2, pp. 498 –519, Feb. 2001.

[16] V. Schwarz, C. Novak, and G. Matz, "Broadcast-based dynamic consensus propagation in wireless sensor networks," in *Proc. Forty-third Annual Asilomar Conference on Signals Systems and Computers*, Pacific Grove, California, November 2009, IEEE.

[17] D. M. Malioutov, J. K. Johnson, and A. S. Willsky, "Walk-sums and belief propagation in gaussian graphical models," *Journal of Machine Learning Research*, vol. 7, pp. 2031–2064, 2006.

[18] P. Honeine, M. Essoloh, C. Richard, and H. Snoussi, "Distributed regression in sensor networks with a reduced-order kernel model," in *Global Telecommunications Conference, IEEE GLOBECOM*, 2008.