Distributed Splitting-Over-Features Sparse Bayesian Learning with Alternating Direction Method of Multipliers

Manss, Christoph; Shutin, Dmitriy; Leus, Geert

DOI
10.1109/ICASSP.2018.8462229

Publication date
2018

Document Version
Final published version

Published in
2018 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2018 - Proceedings

DISTRIBUTED SPLITTING-OVER-FEATURES SPARSE BAYESIAN LEARNING WITH ALTERNATING DIRECTION METHOD OF MULTIPLIERS

Christoph Manss, Dmitriy Shutin

German Aerospace Center (DLR)
Institute of Communication and Navigation
Münchener Straße 22
82234 Weßling, Germany

Geert Leus

Delft University of Technology
Fac. EEMCS
Mekelweg 4
2628CD Delft, Netherlands

ABSTRACT

In processing spatially distributed data, multi-agent robotic platforms equipped with sensors and computing capabilities are gaining interest for applications in inhospitable environments. In this work an algorithm for a distributed realization of sparse bayesian learning (SBL) is discussed for learning a static spatial process with the splitting-over-features approach over a network of interconnected agents. The observed process is modeled as a superposition of weighted kernel functions, or features as we call it, centered at the agent's measurement locations. SBL is then used to determine which feature is relevant for representing the spatial process. Using upper bounding convex functions, the SBL parameter estimation is formulated as ℓ_1-norm constrained optimization, which is solved distributively using alternating direction method of multipliers (ADMM) and averaged consensus. The performance of the method is demonstrated by processing real magnetic field data collected in a laboratory.

Index Terms— Sparse Bayesian learning, ADMM, multi-agent systems, learning over networks.

1. INTRODUCTION

Mobile multi-agent systems, or swarms, are very promising platforms for exploration or monitoring tasks in hazardous or inhospitable environments, where a human operator might be at risk. This pertains to emergency scenarios caused by technogenic accidents, as well as to exploration scenarios in extraterrestrial environments, to name only a few. Through cooperation, swarms can significantly accelerate reconnaissance missions, speed up mapping tasks, increase robustness, and combine resources.

In processing spatially distributed data with a swarm of intelligent agents two strategies can be distinguished. In the first strategy, sometimes termed *homogeneous* or *splitting-over-examples* learning [1, 2], agents collect their own measurements that are used to learn a *common* model. In contrast, the second approach, sometimes referred to as *splitting-over-features* or *heterogeneous* learning [2, 3], assumes that the observed sensor data is known to the whole system. However, each agent only learns his own representation and has only access to his own set of features. Individual models are then coherently combined to form a global estimator. In this work we will consider the second approach, because this approach scales well with large sets of potential features and, thus, reduces the computational complexity [2, Sec. 8.3].

There are two key challenges associated with a splitting-over-features approach toward distributed learning. The first challenge is in combining individual agent responses in a coherent fashion. In particular, the information from individual agents should be appropriately combined to form a collaborative estimator. Several techniques were proposed to address this problem. For instance, a residual refitting algorithm [4, 3] or expectation-maximization based approaches [5] were developed for splitting-over-features approaches with nonlinear agent models. For generalized linear models, the ADMM [2] has recently gained popularity in the community due to its ability to handle non-smooth constraints on model parameters.

The second challenge associated with splitting-over-features approaches is the selection of features relevant for representing the measured data. In recent years, there has been a surge in research related to sparsity (see e.g., [6, 7, 8, 9]) that can be seen as a form of feature selection. Sparsity can be enforced through appropriate penalization of an optimization problem, by either using probabilistic approaches, known as SBL [7, 8], or non-smooth regularization involving, e.g., ℓ_1-norm constraint [6], which are well exemplified by the least absolut shrinkage operator (LASSO) [10, 11]. In this work we will consider the SBL approach to address the feature selection. The reason for this choice is mainly the ability of SBL to cope well with correlated (coherent) features.

There are two main types of SBL methods: Type I and Type II approaches (see e.g., [12]). Both are realized through a hierarchical prior over the unknown model parameters that leads to a sparse maximum a posteriori (MAP) parameter estimate. In Type I SBL the model weights are estimated directly by marginalizing over the prior parameters; through an appropriate choice of the hierarchical prior many traditional "non-Bayesian" methods for learning sparse representations can be realized [12], e.g., basis pursuit de-noising or re-weighted ℓ_p-norm regressions [6, 13].

In contrast, Type II SBL can be used to estimate model parameters indirectly via estimation of prior parameters – hyperparameters – latent variables that regulate the sparsity of the estimated model. The resulting Type II objective function typically exhibits significantly fewer local minima of the corresponding Type I estimator and promotes greater sparsity [14]. Unfortunately, the distributed optimization of the Type II approach is difficult to realize and, to the best of our knowledge, has not been done in the literature.

Thus, our goal is to extend the SBL techniques to splitting-over-features problems applied to learning spatial functions with the Type II approach. To this end we model a spatial field as a superposition of radial basis functions (or kernels) centered at spatial sampling locations; the latter plays the role of features. SBL is then used to impose sparsity constraints on the weights of the basis functions in the superposition. This not only regularizes the estimation problem, but also keeps only the relevant features at each agent. With the help

of an approximation we realize a distributed Type II approach for SBL.

2. SIGNAL MODEL

Let us consider a swarm with K mobile agents. Each agent $k \in \mathcal{K} \triangleq \{1, \ldots, K\}$, is making a scalar sensor measurement $y_k[n] \in \mathcal{Y} \subset \mathbb{R}$, e.g., gas concentration, magnitude of a magnetic field, terrain height, etc., at a position $\boldsymbol{x}_k[n] \in \mathcal{X} \subset \mathbb{R}^2$. The measurements of agent k are identified by an index n, where $n \in \mathcal{N}_k \triangleq \{1, \ldots, N_k\}$, with N_k as the total number of measurements made by the k-th agent. Thus, there are $N = \sum_{k=1}^{K} N_k$ measurements. The positions $\boldsymbol{x}_k[n]$ are assumed to be estimated with a positioning system, such as GNSS, SLAM [15], or motion capture system (e.g., VICON). Here we will assume that all agents are connected in a communication network such that there is a (possibly multi-hop) connection between any two agents in the network.

The goal of the swarm is to reconstruct a spatial scalar function $f : \mathcal{X} \mapsto \mathcal{Y}$ using all collected measurements. We will assume the function f to be sufficiently smooth and model it as a superposition of known and weighted kernel functions $\phi_k(\boldsymbol{x}, \boldsymbol{x}') : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, centered at $\boldsymbol{x}'$ as follows

$$f(\boldsymbol{x}) = \sum_{k=1}^{K} \sum_{n=1}^{N_k} w_{k,n} \phi_k(\boldsymbol{x}, \boldsymbol{x}_k[n]). \tag{1}$$

The role of a kernel function is to model spatial correlations of the observed scalar field. Radial basis functions [16] exemplify well possible kernel choices. Their distinctive feature is that they change monotonically with distance from their central point $\boldsymbol{x}'$, which in our case represents a measurement location $\boldsymbol{x}_k[n]$.

We now aggregate the measurements of the k-th agent into a vector $\boldsymbol{y}_k = [y_k[1], \ldots, y_k[N_k]]^T$, $k \in \mathcal{K}$; total measurements can then be combined in a single vector $\boldsymbol{y} = [\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_K^T]^T$. Consequently, each agent's kernel is evaluated at all available measurement positions such that

$$\begin{aligned} \boldsymbol{\phi}_{k,n} = [&\phi_k(\boldsymbol{x}_1[1], \boldsymbol{x}_k[n]), \ldots, \phi_k(\boldsymbol{x}_1[N_1], \boldsymbol{x}_k[n]), \ldots, \\ &\phi_k(\boldsymbol{x}_K[1], \boldsymbol{x}_k[n]), \ldots, \phi_k(\boldsymbol{x}_K[N_K], \boldsymbol{x}_k[n])]^T. \end{aligned} \tag{2}$$

Thus, $\boldsymbol{\Phi}_k = [\boldsymbol{\phi}_{k,1}, \ldots, \boldsymbol{\phi}_{k,N_k}]$ is the k-th agent's kernel matrix and $\boldsymbol{w}_k = [w_{k,1}, \ldots, w_{k,N_k}]^T$ the corresponding kernel weight vector. In practice, however, the measurements will be perturbed by additive measurement noise. To model this, we assume that measurements $\boldsymbol{y}$ are noisy samples of the function f we intend to learn. It follows then that $\boldsymbol{y}$ can be represented as

$$\boldsymbol{y} = \boldsymbol{\Phi}_1 \boldsymbol{w}_1 + \ldots + \boldsymbol{\Phi}_K \boldsymbol{w}_K + \boldsymbol{\xi} = \boldsymbol{\Phi} \boldsymbol{w} + \boldsymbol{\xi}, \tag{3}$$

where $\boldsymbol{\Phi} = [\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K]$ and $\boldsymbol{w} = [\boldsymbol{w}_1^T, \ldots, \boldsymbol{w}_K^T]^T$ are the aggregated design matrix and weight vector, respectively, for the whole swarm, and $\boldsymbol{\xi}$ represents an additive zero-mean Gaussian measurement noise with covariance matrix $\boldsymbol{\Lambda}^{-1}$.

3. PROBLEM FORMULATION

From (3) the likelihood of the unknown weights $\boldsymbol{w}$ can be written as

$$p(\boldsymbol{y}|\boldsymbol{w}) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{w})^T \boldsymbol{\Lambda} (\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{w})} = \mathrm{e}^{-\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}\boldsymbol{w}\|_{\boldsymbol{\Lambda}}^2}.$$

Maximizing this function directly often leads to overfitting. Also, the matrix $\boldsymbol{\Phi}$ can loose rank if agents make measurements at close spatial locations; this makes numerical estimation of the weights $\boldsymbol{w}$ challenging. This problem can be circumvented by using a regularization of the optimization problem, which removes irrelevant or superfluous features (i.e., columns of $\boldsymbol{\Phi}$ with associated zero weight) from the model.

This can be achieved with SBL [8, 7], where the weights $\boldsymbol{w}$ are constrained with a parametric (hierarchical) prior $p(\boldsymbol{w}|\boldsymbol{\gamma}) = \prod_{l=1}^{N} p(w_l|\gamma_l)$, where $p(w_l|\gamma_l) = \mathcal{N}(0, \gamma_l)$ is a Gaussian probability density function (pdf) with zero mean and variance γ_l (also called sparsity parameter). A small value of γ_l will drive the posterior estimate of the weight w_l toward zero, thus encouraging a sparse solution [8, 14]. The hyperprior $p(\boldsymbol{\gamma})$ is in general a design parameter, which can be chosen according to the problem (see also [12] for possible choices for $p(\boldsymbol{\gamma})$). Yet often the hyperprior $p(\boldsymbol{\gamma})$ is selected to be non-informative, which was shown both empirically and theoretically to perform well [17, 14, 12].

In the Type II estimation framework, the SBL algorithm infers the hyperparameters $\boldsymbol{\gamma}$ by maximizing $p(\boldsymbol{\gamma}|\boldsymbol{y})$ [8, 7, 12]:

$$\begin{aligned} p(\boldsymbol{\gamma}|\boldsymbol{y}) &\propto p(\boldsymbol{\gamma}) \int p(\boldsymbol{y}|\boldsymbol{w}) p(\boldsymbol{w}|\boldsymbol{\gamma}) \mathrm{d}\boldsymbol{w} \\ &\propto p(\boldsymbol{\gamma}) |\boldsymbol{\Sigma}_\gamma|^{-\frac{1}{2}} \mathrm{e}^{-\frac{1}{2}\boldsymbol{y}^T \boldsymbol{\Sigma}_\gamma^{-1} \boldsymbol{y}}, \end{aligned} \tag{4}$$

where $\boldsymbol{\Sigma}_\gamma = \boldsymbol{\Lambda}^{-1} + \boldsymbol{\Phi}\boldsymbol{\Gamma}\boldsymbol{\Phi}^T$. The estimate of the weight vector $\boldsymbol{w}$ is then found by approximating the posterior pdf $p(\boldsymbol{w}|\boldsymbol{y})$ as $p(\boldsymbol{w}|\boldsymbol{y}) \approx p(\boldsymbol{w}|\boldsymbol{y}, \widehat{\boldsymbol{\gamma}}) \propto p(\boldsymbol{y}|\boldsymbol{w}) p(\boldsymbol{w}|\widehat{\boldsymbol{\gamma}})$, where $\widehat{\boldsymbol{\gamma}}$ is a hyperparameter estimate that maximizes (4). Note that $p(\boldsymbol{w}|\boldsymbol{y}, \widehat{\boldsymbol{\gamma}})$ is a Gaussian pdf, since both, the likelihood $p(\boldsymbol{y}|\boldsymbol{w})$ as well as the prior $p(\boldsymbol{w}|\widehat{\boldsymbol{\gamma}})$, are Gaussian. The mean $\widehat{\boldsymbol{w}}$ and covariance matrix $\boldsymbol{\Sigma}_w$ of $p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\gamma})$ can be easily computed as

$$\boldsymbol{\Sigma}_w = \left(\boldsymbol{\Phi}^T \boldsymbol{\Lambda} \boldsymbol{\Phi} + \boldsymbol{\Gamma}^{-1}\right)^{-1}, \quad \widehat{\boldsymbol{w}} = \boldsymbol{\Sigma}_w \boldsymbol{\Phi}^T \boldsymbol{\Lambda} \boldsymbol{y}, \tag{5}$$

where $\boldsymbol{\Gamma} = \mathrm{diag}(\boldsymbol{\gamma})$ is a diagonal matrix with sparsity parameters $\boldsymbol{\gamma}$ on the main diagonal. Note that (5) is essentially a linear minimum mean squared error estimator of the weights $\boldsymbol{w}$ conditioned on the sparsity parameters $\widehat{\boldsymbol{\gamma}}$. Also, we see that the parameters $\widehat{\boldsymbol{\gamma}}$ effectively act as regularization coefficients.

4. DISTRIBUTED SBL

As we mentioned, SBL reduces to finding the hyperparameter vector $\widehat{\boldsymbol{\gamma}}$ that maximizes $p(\boldsymbol{\gamma}|\boldsymbol{y})$ in (4). Here we propose a distributed algorithm that solves this problem, while also estimating the weights $\boldsymbol{w}$ in a distributed fashion for the case when $p(\boldsymbol{\gamma})$ is selected to be non-informative.

Define $\mathcal{L}(\boldsymbol{\gamma}) = -\log p(\boldsymbol{\gamma}|\boldsymbol{y})$. Naturally, the posterior $p(\boldsymbol{\gamma}|\boldsymbol{y})$ in (4) is maximized at [17]

$$\widehat{\boldsymbol{\gamma}} = \underset{\boldsymbol{\gamma}}{\mathrm{argmin}}\, \mathcal{L}(\boldsymbol{\gamma}) = \underset{\boldsymbol{\gamma}}{\mathrm{argmin}} \log(|\boldsymbol{\Sigma}_\gamma|) + \boldsymbol{y}^T \boldsymbol{\Sigma}_\gamma^{-1} \boldsymbol{y}. \tag{6}$$

In [17] the authors have shown that (6) can be upper bounded by

$$\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{z}) = \boldsymbol{z}^T \boldsymbol{\gamma} - g^*(\boldsymbol{z}) + \boldsymbol{y}^T \boldsymbol{\Sigma}_\gamma^{-1} \boldsymbol{y} \geq \mathcal{L}(\boldsymbol{\gamma}), \tag{7}$$

where $g^*(\boldsymbol{z}) = \min_\gamma \boldsymbol{z}^T \boldsymbol{\gamma} - \log(|\boldsymbol{\Sigma}_\gamma|)$ is the concave conjugate of $\log(|\boldsymbol{\Sigma}_\gamma|)$. Problem (6) can then be optimized iteratively via a successive minimization with respect to $\boldsymbol{\gamma}$ and $\boldsymbol{z}$ of an upper bounding functional $\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{z})$. Specifically, we minimize $\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{z})$ with respect to $\boldsymbol{\gamma}$ for a fixed $\boldsymbol{z} = \widehat{\boldsymbol{z}}$; then with $\boldsymbol{\gamma}$ fixed at a new estimate, we perform the minimization with respect to $\widehat{\boldsymbol{z}}$. However, (7) is difficult

to optimize when columns of $\mathbf{\Phi}$ are distributed over different agents. To obtain $\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{z})$ in a more suited form for distributed optimization, we consider the contribution of each agent to the left-hand side of (7).

First, we note that for a fixed $\widehat{\boldsymbol{\gamma}}$, the minimum of $\mathcal{L}(\widehat{\boldsymbol{\gamma}}, \boldsymbol{z})$ along $\boldsymbol{z}$ can be found in closed form at $\widehat{\boldsymbol{z}} = \operatorname{argmin}_{\boldsymbol{z}} \mathcal{L}(\widehat{\boldsymbol{\gamma}}, \boldsymbol{z}) = \operatorname{diag}\left(\mathbf{\Phi}^T \mathbf{\Sigma}_\gamma^{-1} \mathbf{\Phi}\right)$ [17]. Applying matrix inversion lemma [18, 19] to $\mathbf{\Sigma}_\gamma^{-1}$ we can rewrite $\widehat{\boldsymbol{z}}$ as

$$\widehat{\boldsymbol{z}} = \operatorname{diag}\left(\mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi}\right) - \operatorname{diag}\left(\mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi} \mathbf{\Sigma}_w \mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi}\right). \quad (8)$$

Now, we consider an agent $k \in \mathcal{K}$. Both diagonal terms in (8) can then be split according to local variables and variables of other agents such that $\widehat{\boldsymbol{z}} = [\widehat{\boldsymbol{z}}_1^T, \ldots, \widehat{\boldsymbol{z}}_K^T]^T$. It is easy to see that $\operatorname{diag}\left(\mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi}\right)\big|_k = \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k$. Using an appropriate permutation matrix, $\mathbf{\Sigma}_w$ in (8) can always be brought in the following form

$$\mathbf{\Sigma}_w^{-1} = \begin{pmatrix} \mathbf{\Sigma}_{w,k}^{-1} & \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_{\overline{k}} \\ \mathbf{\Phi}_{\overline{k}}^T \mathbf{\Lambda} \mathbf{\Phi}_k & \mathbf{\Sigma}_{w,\overline{k}}^{-1} \end{pmatrix}, \quad (9)$$

where $\mathbf{\Sigma}_{w,k}^{-1} = \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k + \mathbf{\Gamma}_k^{-1}$, $\mathbf{\Sigma}_{w,\overline{k}}^{-1} = \mathbf{\Phi}_{\overline{k}}^T \mathbf{\Lambda} \mathbf{\Phi}_{\overline{k}} + \mathbf{\Gamma}_{\overline{k}}^{-1}$ and subscript index $\overline{k}$ denotes a set of features belonging to all other agents but agent k. The contribution of the k-th agent to the second diagonal term in (8) can be computed as

$$\begin{aligned} \operatorname{diag}\left(\mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi} \mathbf{\Sigma}_w \mathbf{\Phi}^T \mathbf{\Lambda} \mathbf{\Phi}\right)\Big|_k &= \operatorname{diag}\Big(\mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k \mathbf{\Sigma}_{w,k} \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k \\ &+ (\mathbf{\Sigma}_{w,k} \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k - \boldsymbol{I})^T \mathbf{\Omega}_k (\mathbf{\Sigma}_{w,k} \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k - \boldsymbol{I})\Big), \end{aligned} \quad (10)$$

with $\mathbf{\Omega}_k = \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_{\overline{k}} (\mathbf{\Sigma}_{w,\overline{k}}^{-1} - \mathbf{\Phi}_{\overline{k}}^T \mathbf{\Lambda} \mathbf{\Phi}_k \mathbf{\Sigma}_{w,k} \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_{\overline{k}})^{-1} \mathbf{\Phi}_{\overline{k}}^T \mathbf{\Lambda} \mathbf{\Phi}_k$. Observe that the second summand in (10) aggregates the cross-agent correlations; to compute it, we need the features which are distributed among the agents. In contrast, the first term depends only on the information of agent k. Now, using (10) we can re-write (8) as follows

$$\widehat{\boldsymbol{z}}_k = \tilde{\boldsymbol{z}}_k - \operatorname{diag}\Big((\mathbf{\Sigma}_{w,k} \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k - \boldsymbol{I})^T \mathbf{\Omega}_k (\mathbf{\Sigma}_{w,k} \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k - \boldsymbol{I})\Big), \quad (11)$$

where

$$\tilde{\boldsymbol{z}}_k = \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k - \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k \mathbf{\Sigma}_{w,k} \mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_k \quad (12)$$

is the term that collects the correlations between features of the agent k only. Moreover, from (11) it is easy to see that $\widehat{\boldsymbol{z}}_k \leq \tilde{\boldsymbol{z}}_k$, and thus $\tilde{\boldsymbol{z}} = [\tilde{\boldsymbol{z}}_1, \ldots, \tilde{\boldsymbol{z}}_K]$ upper bounds $\widehat{\boldsymbol{z}}$. This permits us to claim that $\mathcal{L}(\boldsymbol{\gamma}, \tilde{\boldsymbol{z}}) \geq \mathcal{L}(\boldsymbol{\gamma}, \widehat{\boldsymbol{z}}) \geq \mathcal{L}(\boldsymbol{\gamma})$, with the first inequality becoming tight when $\mathbf{\Phi}_k^T \mathbf{\Lambda} \mathbf{\Phi}_{k'} = \mathbf{0}$ for any $k \neq k'$ and $k, k' \in \mathcal{K}$. Let us stress that, when agents make measurements at spatially distinct locations, the features will be uncorrelated, which will ensure $\mathcal{L}(\boldsymbol{\gamma}, \tilde{\boldsymbol{z}}) = \mathcal{L}(\boldsymbol{\gamma}, \widehat{\boldsymbol{z}})$. This motivates us to approximate $\widehat{\boldsymbol{z}}$ with $\tilde{\boldsymbol{z}}$, where the latter can be computed from (12) using only local information available at the agent k.

Now we consider a distributed estimation of $\widehat{\boldsymbol{\gamma}}$ using $\mathcal{L}(\boldsymbol{\gamma}, \tilde{\boldsymbol{z}})$. Following [17, Lemma 2], we can upper bound $\mathcal{L}(\boldsymbol{\gamma}, \tilde{\boldsymbol{z}})$ as follows

$$\mathcal{L}(\boldsymbol{\gamma}, \tilde{\boldsymbol{z}}) \leq \boldsymbol{\gamma}^T \tilde{\boldsymbol{z}} + \|\boldsymbol{w}\|_{\mathbf{\Gamma}^{-1}}^2 + \|\boldsymbol{y} - \mathbf{\Phi}\boldsymbol{w}\|_{\mathbf{\Lambda}}^2 = \mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \tilde{\boldsymbol{z}}), \quad (13)$$

which is jointly convex in both $\boldsymbol{\gamma}$ and $\boldsymbol{w}$. Due to this, $\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \tilde{\boldsymbol{z}})$ can be interchangeably minimized with respect to $\boldsymbol{\gamma}$ and $\boldsymbol{w}$. Note that for a fixed $\boldsymbol{\gamma}$ the minimum of $\mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \tilde{\boldsymbol{z}})$ with respect to $\boldsymbol{w}$ is obtained at the maximum of the weight posterior $p(\boldsymbol{w}|\boldsymbol{y}, \boldsymbol{\gamma})$. Now, for any $\boldsymbol{w}$, the bound $\mathcal{L}(\boldsymbol{\gamma}; \boldsymbol{w}; \tilde{\boldsymbol{z}})$ is minimized at value

$$\widehat{\boldsymbol{\gamma}} = \operatorname*{argmin}_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{\gamma}, \boldsymbol{w}; \tilde{\boldsymbol{z}}) = \begin{bmatrix} \frac{|w_{k,n}|}{\sqrt{\tilde{z}_{k,n}}}; & \forall k \in \mathcal{K} \\ & n \in \mathcal{N}_k \end{bmatrix}^T. \quad (14)$$

By inserting (14) in (13), we can see that the bound $\mathcal{L}(\widehat{\boldsymbol{\gamma}}, \boldsymbol{w}; \tilde{\boldsymbol{z}})$ can be made tight by finding $\boldsymbol{w}$ as a solution to the following optimization problem

$$\widehat{\boldsymbol{w}} = \operatorname*{argmin}_{\boldsymbol{w}} \left\| \boldsymbol{y} - \sum_{k=1}^{K} \mathbf{\Phi}_k \boldsymbol{w}_k \right\|_{\mathbf{\Lambda}}^2 + 2 \sum_{k=1}^{K} \sum_{l=1}^{N_k} \sqrt{\tilde{z}_{k,l}} |w_{k,l}| \quad (15)$$

The optimization problem (15) can be solved distributively using ADMM [2, Sec. 8.3]. The method introduces additional variables $\boldsymbol{\zeta}$ and $\boldsymbol{u}$ that can be used to split the optimization into individual problems that can be solved by each agent and stabilize the convergence. The scaled form of the ADMM for our problem and a single agent k can be formulated as [2]

$$\begin{aligned} \boldsymbol{w}_k^{[i+1]} &= \operatorname*{argmin}_{\boldsymbol{w}_k} 2 \sum_{n=1}^{N_k} \sqrt{\tilde{z}_{k,n}} |w_{k,n}| \\ &\quad + \frac{\rho}{2} \|\mathbf{\Phi}_k \boldsymbol{w}_k - \mathbf{\Phi}_k \boldsymbol{w}_k^{[i]} - \boldsymbol{\zeta}_k^{[i]} + \overline{\mathbf{\Phi}\boldsymbol{w}}^{[i]} + \boldsymbol{u}_k^{[i]}\|_2^2 \quad (16) \\ \boldsymbol{\zeta}_k^{[i+1]} &= \operatorname*{argmin}_{\boldsymbol{\zeta}_k} \|\boldsymbol{y} - \boldsymbol{\zeta}_k\|_{\mathbf{\Lambda}}^2 + \frac{\rho}{2} \|\overline{\mathbf{\Phi}\boldsymbol{w}}^{[i]} - \boldsymbol{\zeta}_k + \boldsymbol{u}_k^{[i]}\|_2^2 \\ &= \frac{1}{K|\mathbf{\Lambda}| + \rho} \left(\mathbf{\Lambda}\boldsymbol{y} + \rho \left(\overline{\mathbf{\Phi}\boldsymbol{w}}^{[i]} + \boldsymbol{u}_k^{[i]}\right)\right) \quad (17) \\ \boldsymbol{u}_k^{[i+1]} &= \boldsymbol{u}_k^{[i]} + \overline{\mathbf{\Phi}\boldsymbol{w}}^{[i]} - \boldsymbol{\zeta}_k^{[i+1]}. \quad (18) \end{aligned}$$

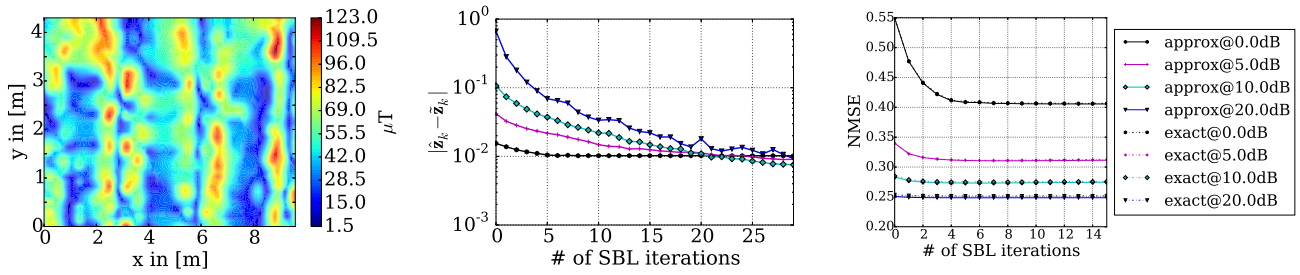
Here $\overline{\mathbf{\Phi}\boldsymbol{w}}^{[i]} = \frac{1}{K} \sum_k \mathbf{\Phi}_k \boldsymbol{w}_k^{[i]}$ is an averaged response of the agents that can be efficiently computed in a distributed fashion using averaged consensus algorithm [20, 21]. Thus, each agent k has to communicate $\mathbf{\Phi}_k \boldsymbol{w}_k^{[i]}$ over the network. Additionally, all agents must have the same measurements $\boldsymbol{y}$, hence this needs to be coordinated as well. Regarding the optimization of (16) a threshold operator is introduced, which sets "irrelevant" weights in $\boldsymbol{w}_k$ to 0. The corresponding features can then be removed from the model, i.e., the number of columns in $\mathbf{\Phi}_k$ is reduced, which keeps the algorithm's computational complexity low.

One downside of ADMM is the introduced parameter ρ of the augmented Lagrangian; when its value is too high, the resulting estimate can become significantly biased; when it is too small, the algorithm might experience convergence problems. Therefore, we propose to use an adaptive approach to estimate ρ as mentioned in [2, Sec. 3.4].

Now, let us summarize the whole algorithm. Its pseudocode is given in Alg. 1. The parameters $c_1 = c_2 = 1e-6$ in Alg. 1 are used as thresholds to check for convergence and $(\cdot)^\dagger$ is the Moore-Penrose Pseudo inverse [19].

5. SIMULATIONS

Since we neglect the cross correlation terms in (11) and therefore only use $\tilde{\boldsymbol{z}}_k$ for obtaining a local solution on agent k, it is necessary, to show the convergence compared to the exact solution of (8). Thus, we show in Fig. 2(a) the qualitative convergence of the new distributed SBL - *approx* - compared to the *exact* solution. We artificially generated a scenario with two γ values such that we can actually compute (6) and the trace of the iteratively estimated $\boldsymbol{\gamma}$ for



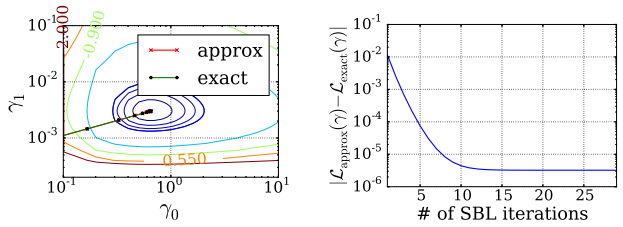
(a) (b) (c)

Fig. 1. (a) The magnetic field of our laboratory used for the simulations in this paper. (b) Difference between exact $\widehat{\boldsymbol{z}}_k$ and approximated $\tilde{\boldsymbol{z}}_k$. (c) NMSE in dependence of SBL iterations. Iteration 0 is the iteration after initialization.

Algorithm 1 Distributed SBL for agent k

Require: $\boldsymbol{y} \in \mathcal{Y}, \boldsymbol{\Phi}_k, \rho > 0$

1: Init $\tilde{\boldsymbol{z}}_k^{[0]} \leftarrow \operatorname{diag}\left(\boldsymbol{\Phi}_k^T \boldsymbol{\Lambda} \boldsymbol{\Phi}_k\right)$; $\boldsymbol{w}_k^{[0]} \leftarrow \left(\boldsymbol{\Phi}_k^T \boldsymbol{\Phi}_k\right)^\dagger \boldsymbol{\Phi}_k^T \boldsymbol{\Lambda} \boldsymbol{y}$
2: **while** $j <$ max_Iteration$_1$ **do**
3: **while** $i <$ max_Iteration$_2$ **do**
4: $\overline{\boldsymbol{\Phi}\boldsymbol{w}}^{[i]} \leftarrow \frac{1}{K} \sum_k \boldsymbol{\Phi}_k \boldsymbol{w}_k^{[i]}$ using averaged consensus
5: $\boldsymbol{w}_k^{[i+1]}, \boldsymbol{\zeta}_k^{[i+1]}, \boldsymbol{u}_k^{[i+1]} \leftarrow$ (16), (17), (18)
6: Update ρ (optional)
7: **if** $\|\boldsymbol{u}_k^{[i+1]} - \boldsymbol{u}_k^{[i]}\| < c_2$ **then**
8: stop
9: **end if**
10: $i \leftarrow i + 1$
11: **end while**
12: $\boldsymbol{w}_k^{[j+1]} \leftarrow \boldsymbol{w}_k^{[i+1]}$
13: Remove zero entries in $\boldsymbol{w}_k^{[j]}$; update $\boldsymbol{\Phi}_k, \boldsymbol{z}_k^{[j]}, \boldsymbol{\gamma}_k^{[j]}$
14: $\boldsymbol{\gamma}^{[j+1]} \leftarrow$ (14)
15: **if** $\|\boldsymbol{\gamma}_k^{[j+1]} - \boldsymbol{\gamma}_k^{[j]}\| < c_1$ **then**
16: stop
17: **end if**
18: $\tilde{\boldsymbol{z}}_k^{[j+1]} \leftarrow$ (12); $j \leftarrow j + 1$
19: **end while**



(a) (b)

Fig. 2. (a) This figure shows the convergence of $\boldsymbol{\gamma}$. For the distributed case the convergence is the same as for the exact solution. (b) Difference between the exact and approximated objective function.

$\widehat{\boldsymbol{z}}$ and $\tilde{\boldsymbol{z}}$.

The difference between both trajectories is marginal, as shown in Fig. 2(b), and the algorithm with the approximated $\tilde{\boldsymbol{z}}_k$ gives the same results for this scenario. Thus, we use this approach in the following.

In the next scenario we use real obtained data of the magnetic field in our laboratory. For this we take spatially uniform distributed samples from the data $N = 500$ and distribute the estimation among $K = 5$ agents. The magnetic field is shown in Fig. 1(a). Each estimation is repeated 50 times with different measurement locations. Each measurement is perturbed with additive white Gaussian noise. We further assume that the noise power is constant, i.e. $\boldsymbol{\Lambda} = \lambda \boldsymbol{I}$. For the estimation of $\widehat{\boldsymbol{w}}$ we use fast iterative shrinkage-thresholding algorithm (FISTA) [22] as a solution to (16). The difference $\|\tilde{\boldsymbol{z}}_k - \widehat{\boldsymbol{z}}_k\|$ is shown in Fig.1(b) for different signal-to-noise ratios (SNRs). It is observed that the difference decreases with increasing number of SBL iterations. This is explained by the consensus in Alg. 1 line 4, which resolves the correlation between other agents. In the case of very low SNR the model contains only of few kernel with values close to zero such that the difference results in low values.

At the end we also investigated the NMSE of the algorithm for different SNR. For evaluation we compared the proposed approximated algorithm with the exact algorithm. As already shown with the qualitative convergence, the performance of the proposed algorithm is almost the same as is shown in Fig. 1(c).

6. CONCLUSION

This paper shows a novel algorithm for distributed SBL with a splitting-over-features or heterogeneous learning approach for multi-agent systems. The algorithm yields the same results as its centralized counterpart by only exchanging the agent's estimates. Due to splitting-over-features, we are able to reduce the computational complexity of the algorithm with respect to the number of features. Additionally, the model size is further reduced by SBL techniques.

We have also shown that the algorithm converges to the same results, although we neglect the inter-agent correlations in (12). This opens the way for other faster versions of distributed optimization for SBL.

In the future we would like to test the algorithm with other data sets and in experiments. Also this algorithm might by interesting for data processing on graphic cards, where the hardware structure, i.e. the cache memory, facilitates the processing.

7. REFERENCES

[1] J. B. Predd, *Topics in Distributed Inference*, Ph.D. thesis, Department of Electrical Engineering, Princeton University, Princeton, NJ, 2006.

[2] Stephen Boyd, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Found. Trends® Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2010.

[3] Haipeng Zheng, S.R. Kulkarni, and H. V. Poor, "Attribute-distributed learning: Models, limits, and algorithms," *IEEE Trans. on Sig. Process.*, vol. 59, no. 1, pp. 386–398, Jan. 2011.

[4] Haipeng Zheng, S.R. Kulkarni, and H. V. Poor, "Dimensionally distributed learning models and algorithm," in *Proc. 11th International Conference on Information Fusion*, Cologne, Germany, 2008, pp. 1–8.

[5] Dmitriy Shutin, Haipeng Zheng, Bernard H. Fleury, Sanjeev R. Kulkarni, and H. Vincent Poor, "Space-alternating attribute-distributed sparse learning," in *Proc. of Int. Workshop on Cognitive Inf. Processing*, Elba Island, Italy, 2010.

[6] E.J. Candes and Michael B Wakin, "An Introduction To Compressive Sampling," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, 2008.

[7] D.P. Wipf and B.D. Rao, "Sparse Bayesian Learning for Basis Selection," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2153–2164, aug 2004.

[8] Michael Tipping, "Sparse Bayesian Learning and The Relevance Vector Machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, jun 2001.

[9] Trevor Park and George Casella, "The Bayesian Lasso," *Journal of the American Statistical Association*, vol. 103, no. 482, pp. 681–686, 2008.

[10] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *J. R. Statist. Soc.*, vol. 58, pp. 267–288, 1994.

[11] Christopher M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer, Singapore, August 2006.

[12] Ritwik Giri and Bhaskar D Rao, "Type i and type ii bayesian methods for sparse signal recovery using scale mixtures.," *IEEE Trans. Signal Processing*, vol. 64, no. 13, pp. 3418–3428, 2016.

[13] Rick Chartrand and Wotao Yin, "Iteratively reweighted algorithms for compressive sensing," in *Acoustics, speech and signal processing, 2008. ICASSP 2008. IEEE international conference on*. IEEE, 2008, pp. 3869–3872.

[14] D.P. Wipf, B.D. Rao, and S. Nagarajan, "Latent variable Bayesian models for promoting sparsity," *IEEE Trans. Inf. Theory*, vol. 57, no. 9, pp. 6236–6255, Sept. 2011.

[15] MWM Gamini Dissanayake, Paul Newman, Steve Clark, Hugh F Durrant-Whyte, and Michael Csorba, "A solution to the simultaneous localization and map building (slam) problem," *IEEE Transactions on robotics and automation*, vol. 17, no. 3, pp. 229–241, 2001.

[16] Mark J. L. Orr, "Introduction to radial basis function networks," Tech. Rep., Centre For Cognitive Science, 1996.

[17] David Wipf and Srikantan Nagarajan, "A new view of automatic relevance determination," in *Proc. 21 Annual Conf. Neural Information Processing Systems*, Vancouver, British Columbia, Canada, Dec. 2007, MIT Press.

[18] Daniel J Tylavsky and Guy RL Sohie, "Generalization of the matrix inversion lemma," *Proceedings of the IEEE*, vol. 74, no. 7, pp. 1050–1052, 1986.

[19] Gene H. Golub and Charles F. Van Loan, *Matrix Computations (3rd Ed.)*, Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[20] Reza Olfati-Saber, J Alex Fax, and Richard M Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[21] Reza Olfati-Saber and Richard M Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on automatic control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[22] Amir Beck and Marc Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.