DISTRIBUTED DECORRELATION IN SENSOR NETWORKS WITH APPLICATION TO DISTRIBUTED PARTICLE FILTERING

Michael Moldaschl[1], Wilfried N. Gansterer[1], Ondrej Hlinka[2], Florian Meyer[2], and Franz Hlawatsch[2]

[1]University of Vienna, Research Group Theory and Applications of Algorithms, Vienna, Austria ({michael.moldaschl, wilfried.gansterer}@univie.ac.at)

[2]Institute of Telecommunications, Vienna University of Technology, Vienna, Austria ({ohlinka, fmeyer, fhlawats}@nt.tuwien.ac.at)

ABSTRACT

Most distributed statistical signal processing methods assume conditionally uncorrelated sensor measurements although this assumption is often not satisfied. Here, we propose a distributed algorithm for decorrelating the sensor measurements in a wireless sensor network. The algorithm employs a matrix-valued Chebyshev approximation to achieve an approximate decorrelation using only local computations and communication between neighboring sensors. We apply the algorithm to consensus-based distributed particle filtering in a target tracking problem with correlated measurement noises. Simulations show that the decorrelation yields a substantial accuracy improvement while causing only a small communication overhead.

Index Terms— Distributed decorrelation, wireless sensor network, Chebyshev approximation, distributed particle filtering, target tracking.

1. INTRODUCTION

Distributed statistical signal processing in decentralized sensor networks arises in many applications [1–3]. Many methods such as consensus-based estimators [4–6] or message passing algorithms [7] rely on the assumption that the sensor measurements are conditionally uncorrelated. However, this assumption is often not satisfied. An example is given by acoustic measurements corrupted by some ambient sound such as wind or machine noise.

Here, we propose a distributed algorithm for (approximately) decorrelating the sensor measurements. The algorithm is based on a matrix-valued Chebyshev approximation of the inverse square root of the global measurement covariance matrix. This approach enables an approximate decorrelation using only local computations at the individual sensor nodes (without a fusion center) and communication between neighboring sensor nodes. Once the sensor measurements have been decorrelated, distributed signal processing algorithms assuming uncorrelated measurements can be applied. As an example, we consider distributed sequential state estimation using a consensus-based distributed particle filter [5].

This paper is organized as follows. In Section 2, we state the system model and the decorrelation problem to be solved. A survey of existing decorrelation schemes, both centralized and distributed, is given in Section 3. The proposed distributed decorrelation algorithm is described in Section 4 and applied to distributed particle filtering in Section 5. Simulation results are presented in Section 6.

2. SYSTEM MODEL AND PROBLEM STATEMENT

We consider a wireless sensor network composed of K sensors. Sensor $k \in \{1, \ldots, K\}$ acquires a random scalar measurement x_k. The covariance matrix of the all-sensors measurement vector $\mathbf{x} \triangleq (x_1 \cdots x_K)^\top$ is $\mathbf{C} \triangleq \mathsf{E}\{(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top\} \in \mathbb{R}^{K\times K}$, where $\boldsymbol{\mu} \triangleq \mathsf{E}\{\mathbf{x}\}$. The expectation is usually conditioned on some unknown parameter or state vector $\boldsymbol{\theta}$; however, this will not be indicated by our notation at this point.

In many applications, entries $C_{k,k'} = \mathsf{E}\{(x_k-\mu_k)(x_{k'}-\mu_{k'})\}$ of $\mathbf{C}$ for sensors k and k' that are spatially distant from one another tend to be (approximately) zero [8]. Let us define the *neighbor set* $\mathcal{N}_k$ of sensor k as the set of all sensors $k' \neq k$ such that $C_{k,k'} \neq 0$. We assume that sensor k is able to communicate with all sensors $k' \in \mathcal{N}_k$; this is consistent with the fact that, typically, these sensors are spatially close to sensor k. Thus, a nonzero off-diagonal element of $\mathbf{C}$ implies that the corresponding element of the adjacency matrix of the communication graph [9] is also nonzero. In other words, the support of the adjacency matrix contains the support of $\mathbf{C}$. Furthermore, we assume that sensor k knows the kth row of $\mathbf{C}$ (or, equivalently, the kth column of $\mathbf{C}$; note that $\mathbf{C} = \mathbf{C}^\top$). This means that sensor k knows all its covariances with other sensor measurements. However, it does not have to know covariances $C_{l,l'}$ with $l, l' \neq k$.

Let $\mathbf{C}^{1/2}$ denote a square root of $\mathbf{C}$, i.e., any $K \times K$ matrix that satisfies $\mathbf{C}^{1/2}(\mathbf{C}^{1/2})^\top = \mathbf{C}$. Note that $\mathbf{C}^{1/2}$ is not unique, since the matrix $\mathbf{C}^{1/2}\mathbf{Q}$, where $\mathbf{Q}$ is any orthonormal $K \times K$ matrix, is also a square root of $\mathbf{C}$. In what follows, we assume that $\mathbf{C}$ (and, hence, $\mathbf{C}^{1/2}$) is nonsingular, and we denote the inverse of $\mathbf{C}^{1/2}$ by $\mathbf{C}^{-1/2}$. Then, it is well known [10] that the vector $\mathbf{y} \in \mathbb{R}^K$ defined as

$$\mathbf{y} \triangleq \mathbf{C}^{-1/2}\mathbf{x} \tag{1}$$

is uncorrelated and normalized in the sense that its covariance matrix equals the identity matrix $\mathbf{I}$. Furthermore, we assume that scalars a, b are known such that $0 < a \leq \lambda_{\min} \leq \lambda_{\max} \leq b$, where $\lambda_{\min}$ ($\lambda_{\max}$) is the smallest (largest) eigenvalue of $\mathbf{C}$.

We can now state the distributed decorrelation problem to be solved: *Using only communication with neighboring sensors $k' \in \mathcal{N}_k$, and based on knowledge of only x_k and the kth row of $\mathbf{C}$, sensor k calculates the kth component y_k of the decorrelated measurement vector $\mathbf{y} = \mathbf{C}^{-1/2}\mathbf{x}$.* As a result of this calculation, each sensor k has a new measurement y_k, where all y_k, $k \in \{1, \ldots, K\}$ are uncorrelated with unit variances.

3. SURVEY OF DECORRELATION ALGORITHMS

Before presenting the proposed distributed decorrelation algorithm, we review existing centralized and distributed algorithms. To the best of our knowledge, none of them is suitable for distributed decorrelation in the setting described in Section 2.

A first class of centralized decorrelation algorithms explicitly computes $\mathbf{C}^{-1/2}$ and then calculates $\mathbf{y}$ according to (1). We note that the matrix-vector multiplication $\mathbf{C}^{-1/2}\mathbf{x}$ in (1) cannot be straightforwardly done in a distributed manner because, in contrast to $\mathbf{C}$, the support of $\mathbf{C}^{-1/2}$ is generally not contained in the support of the

This work was partially supported by the Austrian Science Fund (FWF) under grants S10608 and S10603.

adjacency matrix of the communication graph. An iterative computation of $\mathbf{C}^{-1/2}$ can be based on a coupled inverse Newton iteration [11], the Schur-Newton method [12], or the Newton-Raphson method [13]. Decorrelation algorithms that approximately compute $\mathbf{C}^{-1/2}$ from a training set of samples of $\mathbf{x}$ have been proposed in [14] and [15]. The Cholesky decomposition [16] can be used to compute the lower triangular square root $\mathbf{C}^{1/2}$, which can in turn be used to solve the system of equations $\mathbf{C}^{1/2}\mathbf{y} = \mathbf{x}$ (cf. (1)) for $\mathbf{y}$. Finally, any algorithm for calculating $\operatorname{sign}(\mathbf{A}) \triangleq \mathbf{A}(\mathbf{A}^2)^{-1/2}$ [17] (here, a symmetric square root is used) for a nonsingular matrix $\mathbf{A}$ can be used to calculate the inverse square root of a matrix. In fact, if $\mathbf{A}$ is chosen as a 2×2 block matrix with first block row $(\mathbf{0}\ \ \mathbf{C})$ and second block row $(\mathbf{I}\ \ \mathbf{0})$, then $\operatorname{sign}(\mathbf{A})$ has first block row $(\mathbf{0}\ \ \mathbf{C}^{1/2})$ and second block row $(\mathbf{C}^{-1/2}\ \ \mathbf{0})$. One type of iterative algorithms for computing $\operatorname{sign}(\mathbf{A})$ is based on the Padé approximants [18].

Another class of centralized decorrelation algorithms avoids explicit calculation of $\mathbf{C}^{-1/2}$ or $\mathbf{C}^{1/2}$ and computes $\mathbf{y}$ in (1) by applying operations directly to $\mathbf{x}$. Various algorithms based on Krylov subspace methods or the Chebyshev approximation have been proposed to transform uncorrelated vectors into correlated vectors [19]. These algorithms can also be used for decorrelation by replacing the approximation of $\mathbf{C}^{1/2}$ with an approximation of $\mathbf{C}^{-1/2}$. The Krylov-type algorithms require global aggregation functions (either for a scalar product or for a QR factorization), which can be calculated in a distributed way using, e.g., consensus algorithms [20]. However, the consensus algorithms must be executed in every iteration of the decorrelation algorithm, which can lead to excessive communication requirements.

A distributed Karhunen-Loève transform for decorrelation in sensor networks is proposed in [21]; however, the algorithm requires a fusion center. A truly decentralized decorrelation algorithm is the sparse matrix transform described in [22], which uses Givens rotations to iteratively eliminate the largest off-diagonal element of $\mathbf{C}$. However, this method requires a search over the entire network in each iteration. Moreover, each sensor needs to be able to communicate with all other sensors.

4. THE PROPOSED ALGORITHM

The proposed distributed decorrelation algorithm uses a Chebyshev approximation of matrix functions, which will be reviewed first. The Chebyshev approximation provides a nearly optimal polynomial approximation for any continuous function in an interval $[a, b]$ [23–25]. More precisely, the error of the best polynomial approximation of degree N (which is in general very difficult to find) is smaller by a factor of at most $4 + \ln(N)$ than the error of the Chebyshev approximation with the same degree [25].

4.1. Chebyshev Approximation

A function $f(z)\colon \mathbb{R} \to \mathbb{R}$ can be approximated in the interval $[a, b]$ by the polynomial

$$f^{(N)}(z) \triangleq \sum_{i=1}^{N} \gamma_i T_{i-1}(z') - \frac{\gamma_1}{2},$$

where

$$\gamma_i = \frac{2}{N}\sum_{j=1}^{N} \cos\left((i-1)\,\frac{\pi(j-1/2)}{N}\right) f\left(\frac{1}{\alpha}\cos\left(\frac{\pi(j-1/2)}{N}\right) + \frac{\alpha}{\beta}\right), \tag{2}$$

$T_i(z)$ denotes the Chebyshev polynomial of degree i [26], and

$$z' \triangleq \alpha z - \beta \quad \text{with } \alpha \triangleq \frac{2}{b-a},\ \beta \triangleq \frac{a+b}{b-a} \tag{3}$$

(note that $z' \in [-1, 1]$ for $z \in [a, b]$). The coefficients γ_i in (2) can be computed efficiently via a discrete cosine transform. Furthermore, since polynomials involve only powers, multiplications by scalar factors, and additions, they can be extended to (square) matrix arguments in a straightforward manner. In [19], the above Chebyshev approximation has been generalized to a matrix function $g(\mathbf{Z})\colon \mathbb{R}^{K\times K} \to \mathbb{R}^{K\times K}$ as follows:

$$g(\mathbf{Z}) \approx g^{(N)}(\mathbf{Z}) \triangleq \sum_{i=1}^{N} \gamma_i T_{i-1}(\mathbf{Z}') - \frac{\gamma_1}{2}\mathbf{I}, \tag{4}$$

where $\mathbf{Z}' \triangleq \alpha\mathbf{Z} - \beta\mathbf{I}$ with α and β as defined in (3). For $N \to \infty$, $g^{(N)}(\mathbf{Z})$ converges to $g(\mathbf{Z})$ for all matrices $\mathbf{Z}$ satisfying $a \le \lambda_{\min} \le \lambda_{\max} \le b$, which implies that α and β are related to the smallest eigenvalue $\lambda_{\min}$ and largest eigenvalue $\lambda_{\max}$ of $\mathbf{Z}$ [19].

Using the matrix function $g(\mathbf{Z}) \triangleq \mathbf{Z}^{-1/2}$, we can rewrite (1) as $\mathbf{y} = g(\mathbf{C})\mathbf{x}$. Inserting the Chebyshev approximation (4) then yields

$$\mathbf{y} \approx \mathbf{y}^{(N)} \triangleq \left[\sum_{i=1}^{N} \gamma_i T_{i-1}(\mathbf{C}') - \frac{\gamma_1}{2}\mathbf{I}\right]\mathbf{x} = \sum_{i=1}^{N} \gamma_i \mathbf{t}_i - \frac{\gamma_1}{2}\mathbf{x}, \tag{5}$$

with $\mathbf{C}' \triangleq \alpha\mathbf{C} - \beta\mathbf{I}$, the vectors $\mathbf{t}_i \triangleq T_{i-1}(\alpha\mathbf{C} - \beta\mathbf{I})\mathbf{x} \in \mathbb{R}^K$, and the coefficients γ_i given by (2) with $f(z) = z^{-1/2}$. Based on the well-known recursive calculation of the Chebyshev polynomials $T_i(\cdot)$ [26], the $\mathbf{t}_i$ can be calculated recursively as

$$\mathbf{t}_i = 2(\alpha\mathbf{C} - \beta\mathbf{I})\,\mathbf{t}_{i-1} - \mathbf{t}_{i-2}, \quad i \in \{3, 4, \ldots, N\}, \tag{6}$$

with initialization

$$\mathbf{t}_1 = \mathbf{x}, \quad \mathbf{t}_2 = (\alpha\mathbf{C} - \beta\mathbf{I})\,\mathbf{x}. \tag{7}$$

We finally note that the Chebyshev approximation in (5) can be written as $\mathbf{y}^{(N)} = \mathbf{A}\mathbf{x}$, where $\mathbf{A} \triangleq \sum_{i=1}^{N} \gamma_i T_{i-1}(\alpha\mathbf{C} - \beta\mathbf{I}) - \frac{\gamma_1}{2}\mathbf{I}$ is a symmetric matrix. Thus, in the approximation of $\mathbf{y} = \mathbf{C}^{-1/2}\mathbf{x} = (\mathbf{C}^{1/2})^{-1}\mathbf{x}$, $\mathbf{C}^{1/2}$ is a *symmetric* matrix square root.

4.2. Distributed Decorrelation Algorithm

The approximate computation of $\mathbf{y}$ according to (5)–(7) requires repeated multiplication by the matrix $\alpha\mathbf{C} - \beta\mathbf{I}$. Since the support of this matrix is contained in the support of the adjacency matrix of the communication graph (cf. Section 2), and since each sensor k needs to obtain only the kth component of the resulting vector $\mathbf{y}^{(N)}$, all matrix-vector multiplications can be performed in a decentralized way using only local computations and communication with neighbors. Furthermore, the vector summations in (5) and (6) are performed locally, i.e., the kth component summation is performed at the kth sensor without any intersensor communication. The resulting distributed decorrelation algorithm can be stated as follows.

DISTRIBUTED DECORRELATION ALGORITHM

Sensor k performs the following steps:

Step 1 (Initialization):

a) $t_{1,k} = x_k$ (cf. (7)) is broadcast to all neighbor sensors $k' \in \mathcal{N}_k$.

b) $t_{2,k} = \left[(\alpha\mathbf{C} - \beta\mathbf{I})\mathbf{x}\right]_k$ (cf. (7)) is calculated and broadcast to all neighbor sensors $k' \in \mathcal{N}_k$.

c) $y_k^{(2)} = \gamma_2\, t_{2,k} + \gamma_1\,(t_{1,k} - x_k/2)$ (cf. (5)) is calculated locally.

Step 2 (Iteration): For $i = 3, 4, \ldots, N$:

a) $t_{i,k} = \left[2(\alpha\mathbf{C} - \beta\mathbf{I})\,\mathbf{t}_{i-1} - \mathbf{t}_{i-2}\right]_k$ (cf. (6)) is calculated and, if $i \le N-1$, broadcast to all neighbor sensors $k' \in \mathcal{N}_k$.

b) $y_k^{(i)} = y_k^{(i-1)} + \gamma_i\, t_{i,k}$ (cf. (5)) is calculated locally.

The result of this algorithm at sensor k is $y_k^{(N)}$, which is the kth element of the Chebyshev approximation of $\mathbf{y}$ in (5), i.e., $y_k^{(N)} = \sum_{i=1}^{N} \gamma_i t_{i,k} - \frac{\gamma_1}{2} x_k \approx y_k$. The algorithm requires knowledge (at sensor k) of x_k and of the kth row of $\mathbf{C}$ (see Steps 1b and 2a). Other quantities used in Steps 1b and 2a are $x_{k'}$ and $t_{i-1,k'}$ for $k' \in \mathcal{N}_k$; these have been received from the neighbor sensors. The coefficients γ_i do not depend on $\mathbf{x}$ and can thus be precomputed at each sensor.

At iteration $i \in \{1, 2, \ldots, N-1\}$ (here, $i = 1$ and $i = 2$ refer to the initialization in Step 1), sensor k broadcasts $t_{i,k}$ to its neighbors $k' \in \mathcal{N}_k$. Thus, during the entire iterative decorrelation process, sensor k broadcasts $N-1$ real numbers to its neighbors.

The approximation error $\varepsilon \triangleq \|\mathbf{y}^{(N)} - \mathbf{y}\|_2 = \big\| \sum_{i=N+1}^{\infty} \gamma_i \mathbf{t}_i \big\|_2$ is dominated by $|\gamma_{N+1}| \|\mathbf{t}_{N+1}\|_2$ [27]. Using the approximation $\varepsilon \approx |\gamma_{N+1}| \|\mathbf{t}_{N+1}\|_2$, the relative error $\overline{\varepsilon} \triangleq \varepsilon / \|\mathbf{x}\|_2$ can be approximated locally at each sensor as $\overline{\varepsilon} \approx |\gamma_{N+1}| \|\mathbf{t}_{N+1}\|_2 / \|\mathbf{x}\|_2 \leq |\gamma_{N+1}| < |\gamma_N|$ for sufficiently large N. Therefore, the N required to achieve a desired value $\bar{\varepsilon}_0$ of $\overline{\varepsilon}$ can be determined locally at each sensor as the smallest N for which $|\gamma_N| < \bar{\varepsilon}_0$. The obtained value of N is identical for all k. In our numerical experiments, we always observed that $\overline{\varepsilon}$ is well approximated by $|\gamma_N|$ as long as the accuracy does not approach machine precision. In Section 6.2, we will illustrate this approximation for a target tracking scenario.

All sensors need to use the same approximation interval $[a, b]$ in order to operate with the same values α and β in (3). As mentioned in Section 2, the approximation interval $[a, b]$ has to include the spectrum of $\mathbf{C}$, i.e., $0 < a \leq \lambda_{\min} \leq \lambda_{\max} \leq b$. The closer a and b are to $\lambda_{\min}$ and $\lambda_{\max}$, respectively, the faster converges the Chebyshev approximation. The problem of distributed estimation of $\lambda_{\min}$ and $\lambda_{\max}$ is beyond the scope of this paper; we just note that distributed eigensolvers have been proposed recently [28, 29]. Estimation of $\lambda_{\min}$ and $\lambda_{\max}$ requires additional intersensor communication. However, this preparatory step is needed only once, provided that $\mathbf{C}$ does not change.

5. APPLICATION: DISTRIBUTED PARTICLE FILTERING

Distributed particle filters (DPFs) are powerful methods for distributed sequential Bayesian state estimation in wireless sensor networks [6]. DPFs that employ consensus algorithms are especially advantageous because they are robust to sensor and communication link failures and can obtain a global estimate at each sensor. All consensus-based DPFs (with one exception [30]) rely on the assumption that the measurement noises at the various sensors are mutually uncorrelated.

In this section, we address the case of correlated measurement noises and use our distributed decorrelation algorithm to enable the application of standard consensus-based DPFs. We consider the DPF of [5] as a concrete example. The resulting DPF is different from the DPF for correlated measurement noises proposed in [30], which assumes that the *inverse* covariance matrix (known as the precision matrix) has the same support as the adjacency matrix.

5.1. System Model

We consider sequential Bayesian estimation of a random, time-varying state vector $\boldsymbol{\theta}_n = (\theta_{n,1} \cdots \theta_{n,M})^\top$. The state evolves according to $\boldsymbol{\theta}_n = \mathbf{a}_n(\boldsymbol{\theta}_{n-1}, \mathbf{u}_n)$, where $\mathbf{a}_n(\cdot, \cdot)$ is a generally nonlinear function and $\mathbf{u}_n$ is driving noise. At time n, $\boldsymbol{\theta}_n$ is sensed by K sensors according to the sensor measurement models

$$x_{n,k} = h_k(\boldsymbol{\theta}_n) + v_{n,k}\,, \quad k \in \{1, \ldots, K\}. \tag{8}$$

Here, $x_{n,k} \in \mathbb{R}$ is the local measurement of sensor k, $h_k(\cdot)$ is a generally nonlinear local measurement function, and $v_{n,k}$ is local measurement noise. We assume that at any time n, the random variables $v_{n,k}$, $k \in \{1, \ldots, K\}$ are jointly Gaussian, zero-mean, and possibly correlated; however, $v_{n,k}$ and $v_{n',k'}$ are assumed independent at different times $n \neq n'$. Combining all local measurement models (8) yields the global (all-sensors) measurement model

$$\mathbf{x}_n = \mathbf{h}(\boldsymbol{\theta}_n) + \mathbf{v}_n\,, \tag{9}$$

where $\mathbf{x}_n \triangleq (x_{n,1} \cdots x_{n,K})^\top$, $\mathbf{h}(\cdot) \triangleq (h_1(\cdot) \cdots h_K(\cdot))^\top$, and $\mathbf{v}_n \triangleq (v_{n,1} \cdots v_{n,K})^\top \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_v)$. The noise covariance matrix $\mathbf{C}_v$ is also the conditional covariance matrix of the global (all-sensors) measurement vector $\mathbf{x}_n$ given $\boldsymbol{\theta}_n$, i.e., $\mathsf{E}\big\{(\mathbf{x}_n - \boldsymbol{\mu})(\mathbf{x}_n - \boldsymbol{\mu})^\top \big| \boldsymbol{\theta}_n\big\} = \mathbf{C}_v$, with $\boldsymbol{\mu} \triangleq \mathsf{E}\{\mathbf{x}_n | \boldsymbol{\theta}_n\} = \mathbf{h}(\boldsymbol{\theta}_n)$. As in Section 2, we define the neighbor set of sensor k as $\mathcal{N}_k \triangleq \big\{k' \neq k \big| [\mathbf{C}_v]_{k,k'} \neq 0\big\}$, and we assume that sensor k knows the kth row of $\mathbf{C}_v$ and is able to communicate with all sensors $k' \in \mathcal{N}_k$.

5.2. Distributed Particle Filtering

After application of our distributed decorrelation algorithm with $\mathbf{C} = \mathbf{C}_v$ to the sensor measurements $x_{n,k}$, $k \in \{1, \ldots, K\}$, each sensor k has a new local measurement $y_{n,k}$. Because $\mathbf{C} = \mathbf{C}_v$ is the conditional covariance matrix of $\mathbf{x}_n$ given $\boldsymbol{\theta}_n$, the $y_{n,k}$ are (approximately) conditionally uncorrelated given $\boldsymbol{\theta}_n$. Ignoring the approximations introduced by our algorithm, the decorrelated measurement vector $\mathbf{y}_n \triangleq (y_{n,1} \cdots y_{n,K})^\top$ can be written as

$$\mathbf{y}_n = \mathbf{C}_v^{-1/2} \mathbf{x}_n \overset{(9)}{=} \mathbf{C}_v^{-1/2} \big[\mathbf{h}(\boldsymbol{\theta}_n) + \mathbf{v}_n\big] = \mathbf{h}'(\boldsymbol{\theta}_n) + \mathbf{v}'_n\,,$$

with the new measurement function $\mathbf{h}'(\boldsymbol{\theta}_n) \triangleq \mathbf{C}_v^{-1/2} \mathbf{h}(\boldsymbol{\theta}_n)$ and the new—decorrelated and normalized—measurement noise vector $\mathbf{v}'_n \triangleq \mathbf{C}_v^{-1/2} \mathbf{v}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Note furthermore that $\mathbf{y}_n | \boldsymbol{\theta}_n \sim \mathcal{N}\big(\mathbf{C}_v^{-1/2} \mathbf{h}(\boldsymbol{\theta}_n), \mathbf{I}\big)$.

Because the measurement noises $v'_{n,k}$ contained in the new measurements $y_{n,k}$ are uncorrelated, any standard DPF assuming uncorrelated measurement noises can now be used. In the consensus-based DPF presented in [5], in particular, each sensor k runs a local particle filter that computes a global state estimate $\hat{\boldsymbol{\theta}}_n$ based on the all-sensors measurement vector $\mathbf{y}_n$. For this, sensor k needs to know, besides the new local measurement $y_{n,k}$, the kth component $h'_k(\cdot) \triangleq \big[\mathbf{h}'(\cdot)\big]_k$ of the new measurement function $\mathbf{h}'(\cdot) = \mathbf{C}_v^{-1/2} \mathbf{h}(\cdot)$ [5]. We have

$$h'_k(\cdot) = \mathbf{d}_k^\top \mathbf{h}(\cdot)\,, \tag{10}$$

where $\mathbf{d}_k^\top$ denotes the kth row of $\mathbf{C}_v^{-1/2}$. Note that $\mathbf{d}_k$ also equals the kth column of $\mathbf{C}_v^{-1/2}$, because our distributed decorrelation algorithm implicitly constructs a symmetric inverse square root $\mathbf{C}_v^{-1/2}$.

Evaluation of (10) presupposes that sensor k knows $\mathbf{d}_k$ and the original measurement functions $h_l(\cdot)$ of all other sensors. A distributed calculation of $\mathbf{d}_k$ can be based on the following consideration. With $\boldsymbol{\delta}_l$, $l \in \{1, \ldots, K\}$ denoting the lth unit vector of length K (i.e., $[\boldsymbol{\delta}_l]_i = \delta_{il}$), we have $\mathbf{C}_v^{-1/2} \boldsymbol{\delta}_l = \mathbf{d}_l$. If our distributed decorrelation algorithm is applied to input vector $\boldsymbol{\delta}_l$ (instead of $\mathbf{x}$), then sensor k obtains the kth component of the lth column $\mathbf{d}_l$, $[\mathbf{d}_l]_k$, which is simultaneously the lth component of the kth row, $[\mathbf{d}_k^\top]_l$. Thus, if our distributed decorrelation algorithm is executed K times, using $\boldsymbol{\delta}_l$ as input at the lth execution, with $l = 1, \ldots, K$, sensor k obtains all K components of $\mathbf{d}_k$.

Finally, if the original measurement functions $h_k(\cdot)$ of all sensors are not known to each sensor *a priori*, they have to be distributed, e.g., through flooding or routing. In most applications, a closed-form, parametric expression of the functions $h_k(\cdot)$ is available, so that only certain parameters of the $h_k(\cdot)$ (e.g., the sensor locations) have to be transmitted. Note that the calculation of the kth row of

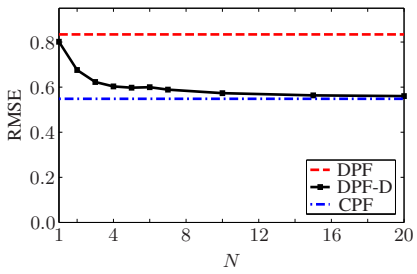

Fig. 1. RMSE of DPF-D (proposed), DPF, and CPF versus number of Chebyshev iterations N.

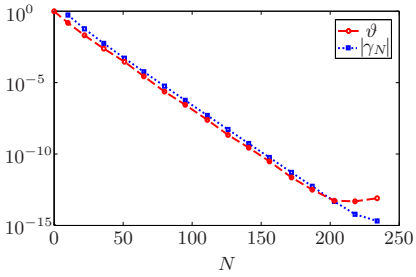

Fig. 2. Decorrelation error ϑ and magnitude of Chebyshev approximation coefficient $|\gamma_N|$ versus number of Chebyshev iterations N.

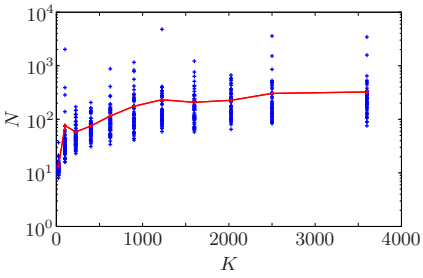

Fig. 3. Estimated number of Chebyshev iterations N for decorrelation error $\vartheta = 10^{-4}$ versus network size K. The red curve shows the average.

$\mathbf{C}_v^{-1/2}$ and the dissemination of the original measurement functions $h_k(\cdot)$ are preparatory steps that have to be executed only once (provided that $\mathbf{C}_v$ and the $h_k(\cdot)$ do not change with time n).

6. NUMERICAL EXPERIMENTS

We present simulation results assessing the performance of the DPF described in Section 5.2 and the accuracy of the proposed distributed decorrelation algorithm.

6.1. DPF Accuracy

We consider a target tracking problem based on a network of $K = 25$ sensors, which are deployed on a jittered grid within a square region of size $d_a \times d_a$ with $d_a = 40$. The system model (target dynamics and measurement model) is identical to that used in [6], except that the measurement noises at different sensors are mutually correlated according to the spatial correlation model of [8]. This model defines $\mathbf{C}_v \in \mathbb{R}^{25\times 25}$ as

$$[\mathbf{C}_v]_{k,k'} = \begin{cases} \sigma_k^2\,, & k = k' \\ \sigma_k \sigma_{k'} \exp(-\eta d_{k,k'}^2)\,, & k \neq k',\ d_{k,k'} \leq d_c \\ 0\,, & k \neq k',\ d_{k,k'} > d_c\,, \end{cases} \quad (11)$$

where $d_{k,k'}$ is the spatial distance between sensors k and k', and d_c is the maximum correlation radius (i.e., the measurement noises of two sensors whose distance exceeds d_c are uncorrelated). Simultaneously, d_c is also the communication range. We set $\sigma_k^2 = 0.01$ for all k, $\eta = 0.007$, and $d_c = 20$.

We study the performance of the DPF with decorrelation as described in Section 5.2 (abbreviated DPF-D), of the original DPF of [5] without decorrelation (abbreviated DPF), and of a centralized PF (CPF) that processes all sensor measurements at a fusion center (here, no decorrelation is needed). Both DPF and DPF-D use 10 consensus iterations. For the Chebyshev approximation used in our decorrelation algorithm, we set $a = \lambda_{\min}$ and $b = \lambda_{\max}$, where $\lambda_{\min}$ ($\lambda_{\max}$) is the smallest (largest) eigenvalue of $\mathbf{C}_v$. Fig. 1 shows the root-mean-square error (RMSE) of the estimated position of the target obtained with DPF-D versus the number of iterations N used for decorrelation. The RMSE was computed by averaging over the 25 sensors, 200 time instants, and 5000 simulation runs. As a reference, the RMSEs of DPF and CPF (which are independent of N) are also shown. We see that the RMSE of DPF-D is significantly lower than that of DPF, and it approaches that of CPF within approximately $N = 20$ iterations. For distributed decorrelation with $N = 20$, each sensor has to broadcast 19 real numbers to its neighbors. For comparison, we note that for execution of the DPF (without decorrelation) using 10 consensus iterations, each sensor has to broadcast 140 real numbers to its neighbors. Thus, the additional amount of communication required for decorrelation is moderate.

6.2. Decorrelation Accuracy

To investigate the accuracy of our decorrelation algorithm, we applied it to 4000 vectors $\mathbf{x}_l \in \mathbb{R}^{25}$, $l \in \{1, \ldots, 4000\}$ randomly drawn from $\mathcal{N}(\mathbf{0}, \mathbf{C}_v)$, with $\mathbf{C}_v$ as in Section 6.1. Based on the resulting vectors $\mathbf{y}_l$, the decorrelation error was then measured by $\vartheta \triangleq \max_{i,j} \left|[\hat{\mathbf{C}}_y - \mathbf{I}]_{i,j}\right|$, where $\hat{\mathbf{C}}_y$ is the sample covariance matrix of the $\mathbf{y}_l$, i.e., $\hat{\mathbf{C}}_y \triangleq \frac{1}{4000}\sum_{l=1}^{4000}(\mathbf{y}_l - \hat{\boldsymbol{\mu}}_y)(\mathbf{y}_l - \hat{\boldsymbol{\mu}}_y)^\top$ with $\hat{\boldsymbol{\mu}}_y \triangleq \frac{1}{4000}\sum_{l=1}^{4000}\mathbf{y}_l$. In Fig. 2, we see that ϑ decreases exponentially fast with growing N up to about $N = 200$, where ϑ has decayed to a residual of about 10^{-13}. Interestingly, this behavior is quite different from that of the RMSE of DPF-D in Fig. 1, which flattens and becomes close to that of CPF for N larger than about 20, even though the decorrelation error is still quite large. Fig. 2 also shows that the magnitude of the last Chebyshev coefficient $|\gamma_N|$ is quite close to ϑ until it gets close to machine precision. We recall from Section 4.2 that this approximate equality of $|\gamma_N|$ and ϑ, combined with the fact that γ_N can be calculated locally by each sensor, allows each sensor to estimate *locally* the number of iterations N that is required for a desired accuracy.

We also simulated the decorrelation algorithm for larger networks with K up to 3600. Here, we used the parameter $\eta = 0.02$ in (11) in order to obtain positive definite matrices $\mathbf{C}_v$. For each network size K, we randomly generated 60 networks. In Fig. 3, the estimated number of iterations N (estimated as described in Section 4.2) that is required to achieve decorrelation error $\vartheta = 10^{-4}$ is displayed for each network versus the network size K. In addition, the figure shows the average of the estimated N over all 60 networks of a given size. The actual N required to achieve $\vartheta = 10^{-4}$ was verified to be close to the estimated N in all cases. Our results suggest that the average number of Chebyshev iterations N required for $\vartheta = 10^{-4}$ grows roughly according to the square root of the network size K (up to a factor and a constant term).

7. CONCLUSION

We presented a distributed decorrelation algorithm for sensor networks with mutually conditionally correlated sensor measurements. Based on the approximation of a matrix function using Chebyshev polynomials, the proposed algorithm requires only local computations and communication with neighboring sensors. We used the algorithm to obtain a consensus-based distributed particle filter capable of processing correlated sensor measurements. Simulation results demonstrated the good performance of the distributed particle filter with only a small amount of additional communication required by the distributed decorrelation (in addition to the communication required by the distributed particle filter). The extension of the proposed distributed decorrelation algorithm to vector-valued sensor measurements is an interesting topic for future research.

8. REFERENCES

[1] F. Zhao and L. J. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. Amsterdam, The Netherlands: Morgan Kaufmann, 2004.

[2] S. Haykin and K. J. R. Liu, *Handbook on Array Processing and Sensor Networks*. Hoboken, NJ: Wiley, 2009.

[3] G. Ferrari, *Sensor Networks: Where Theory Meets Practice*. Heidelberg, Germany: Springer, 2010.

[4] B. N. Oreshkin and M. J. Coates, "Asynchronous distributed particle filter via decentralized evaluation of Gaussian products," in *Proc. FUSION-10*, Edinburgh, UK, Jul. 2010.

[5] O. Hlinka, O. Slučiak, F. Hlawatsch, P. M. Djurić, and M. Rupp, "Likelihood consensus and its application to distributed particle filtering," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4334–4349, 2012.

[6] O. Hlinka, F. Hlawatsch, and P. M. Djurić, "Distributed particle filtering in agent networks: A survey, classification, and comparison," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 61–81, 2013.

[7] A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Comm.*, vol. 23, pp. 809–819, Apr. 2005.

[8] Z. Quan, W. J. Kaiser, and A. H. Sayed, "Innovations diffusion: A spatial sampling scheme for distributed estimation and detection," *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 738–751, 2009.

[9] A. E. Brouwer and W. H. Haemers, *Spectra of Graphs*. New York, NY: Springer, 2012.

[10] C. W. Therrien, *Discrete Random Signals and Statistical Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1992.

[11] B. Iannazzo, "On the Newton method for the matrix *p*th root," *SIAM J. Matrix Anal. Appl.*, vol. 28, no. 2, pp. 503–523, 2006.

[12] C. Guo and N. J. Higham, "A Schur-Newton method for the matrix *p*th root and its inverse," *SIAM J. Matrix Anal. Appl.*, vol. 28, no. 3, pp. 788–804, 2006.

[13] J.-A. Pineiro and J. D. Bruguera, "High-speed double-precision computation of reciprocal, division, square root, and inverse square root," *IEEE Trans. Computers*, vol. 51, no. 12, pp. 1377–1388, 2002.

[14] S. C. Douglas and A. Cichocki, "Neural networks for blind decorrelation of signals," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2829–2842, 1997.

[15] S. Gazor and T. Liu, "Adaptive filtering with decorrelation for coloured AR environments," *IEE Proc. Vision, Image, Sig. Process.*, vol. 152, no. 6, pp. 806–818, 2005.

[16] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins Univ. Press, 1996.

[17] N. J. Higham, *Functions of Matrices*. Philadelphia, PA: SIAM, 2008.

[18] C. Kenney and A. J. Laub, "Rational iterative methods for the matrix sign function," *SIAM J. Matrix Anal. Appl.*, vol. 12, no. 2, pp. 273–291, 1991.

[19] T. Ando, E. Chow, Y. Saad, and J. Skolnick, "Krylov subspace methods for computing hydrodynamic interactions in Brownian dynamics simulations," *J. Chem. Phys.*, vol. 137, no. 6, pp. 064106–1–064106–14, 2012.

[20] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[21] M. Gastpar, P. L. Dragotti, and M. Vetterli, "The distributed Karhunen-Loève transform," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5177–5196, 2006.

[22] L. R. Bachega, S. Hariharan, C. A. Bouman, and N. Shroff, "Distributed signal decorrelation in wireless sensor networks using the sparse matrix transform," *Proc. SPIE 8058*, pp. 80580V–1–80580V–15, 2011.

[23] K. Geddes, "Near-minimax polynomial approximation in an elliptical region," *SIAM J. Numer. Anal.*, vol. 15, no. 6, pp. 1225–1233, 1978.

[24] M. J. D. Powell, "On the maximum errors of polynomial approximations defined by interpolation and by least squares criteria," *The Computer Journal*, vol. 9, no. 4, pp. 404–407, 1967.

[25] E. W. Cheney, *Introduction to Approximation Theory*. Providence, RI: AMS Chelsea Pub., 1982.

[26] J. C. Mason and D. C. Handscomb, *Chebyshev Polynomials*. Boca Raton, FL: CRC Press, 2002.

[27] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in Fortran 77*, vol. 1. New York, NY: Cambridge University Press, 1997.

[28] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," *J. Comp. Syst. Sci.*, vol. 74, pp. 70– 83, Feb. 2008.

[29] H. Strakova and W. N. Gansterer, "A distributed eigensolver for loosely coupled networks," in *Proc. 21st Euromicro Conf. Parallel, Distrib., Network-Based Process.*, Belfast, UK, pp. 51–57, Feb. 2013.

[30] O. Hlinka and F. Hlawatsch, "Distributed particle filtering in the presence of mutually correlated sensor noises," in *Proc. IEEE ICASSP-13*, Vancouver, BC, Canada, pp. 6269–6273, 2013.