AVERAGE CONSENSUS IN WIRELESS SENSOR NETWORKS: WILL IT BLEND?

Valentin Schwarz and Gerald Matz

Institute of Communications and Radio-Frequency Engineering, Vienna University of Technology
Gusshausstrasse 25/389, 1040 Vienna, Austria; email: {vschwarz,gmatz}@nt.tuwien.ac.at

ABSTRACT

Average consensus (AC) is a popular method for distributed in-network averaging that is closely related to diffusion processes. Recently, it has been suggested that AC designs inspired by physical advection processes feature improved performance. In this paper, we show how such advection-diffusion inspired AC designs can be obtained for realistic network topologies. The resulting AC algorithms act like a blender that mixes the measured data. Numerical simulations corroborate the performance improvements of our blending approach to the design of AC algorithms.

Index Terms— Average consensus, wireless sensor networks, distributed inference, advection-diffusion equation

1. INTRODUCTION

The distributed computation of the average (arithmetic mean) of measured data is the basis for more sophisticated distributed inference algorithms in wireless sensor networks (WSN), e.g., likelihood consensus [1] or distributed field reconstruction [2]. *Average consensus* (AC), introduced in [3,4], is a particularly popular method for distributed averaging. An overview of AC methods is given in [5] and an AC weight design that optimizes the asymptotic convergence rate has been proposed in [6]. Further performance improvements can be obtained with time-varying AC weights, , e.g. [7]. A per-step MSE-optimal weight design exploiting the measurement statistics has been described in [8]; in this paper, we found that the correlation of the data to be averaged has a crucial impact on the performance of AC, i.e., strong correlation tends to diminish the convergence speed.

In a recent paper [9], the authors use an analogy with fluid dynamics to improve the convergence speed of AC based on a discretized advection-diffusion equation with a simple velocity field (i.e., flow). However, their design is limited to networks with nodes uniformly distributed on a torus (i.e., without boundaries). Unfortunately, the practical relevance of such a setup is rather limited.

Inspired by [9], this paper proposes improved AC weight designs for realistic WSN scenarios. More specifically, our contributions are:

- a new discretization for advection-diffusion processes that leads to AC designs with improved performance;
- for the case where sensors are arranged on a uniform grid, we propose new velocity fields and provide a stability analysis for the resulting AC algorithm (following [10]);
- for arbitrary network topologies, we present an optimization approach for the advection component of the AC weights;
- we illustrate the performance gains of our AC weight designs via numerical simulations.

Funded by WWTF Grant ICT08-044 and FWF Grant S10606. The authors thank Yiğit Onat for valuable input.

2. PRELIMINARIES

2.1. Wireless Sensor Networks

We consider WSN consisting of I sensor nodes that are located at positions $\boldsymbol{r}_i$, $i = 1, \ldots, I$. The WSN topology is modeled via a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$; here, $\mathcal{V}$ denotes the set of nodes ($I = |\mathcal{V}|$) and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the set of undirected edges, which represent the bidirectional communication links between the nodes.

Two important WSN topologies used in what follows are i) uniform rectangular WSN in which nodes are arranged on a square lattice and communicate with their nearest neighbors only; ii) random geometric WSN, in which the node positions are i.i.d. uniformly distributed and communication occurs between nodes within a prescribed range.

2.2. Average Consensus

The goal of AC is to compute the average $\bar{s} = \frac{1}{I}\sum_i s_i$ in a distributed fashion; here, s_i denotes the data (measurement) at node i. Node's i estimate of the mean at iteration k is given by the state $x_i[k]$. The states are updated in each iteration according to

$$x_i[k+1] = w_{ii}\, x_i[k] + \sum_{j:\,(i,j)\in\mathcal{E}} w_{ij}\, x_j[k]\,,$$

with initialization $x_i[0] = s_i$. The choice of the weights w_{ij} will be discussed in more detail later. The state updates can be rewritten as,

$$\boldsymbol{x}[k+1] = \boldsymbol{W}\boldsymbol{x}[k]\,, \tag{1}$$

where the weight matrix is defined by $[\boldsymbol{W}]_{ij} = w_{ij}$ for $(i,j) \in \mathcal{E}$ or $i = j$ and $w_{ij} = 0$ otherwise. Convergence of the AC iterations to the true mean is guaranteed whenever the following conditions are satisfied: $\boldsymbol{W}\mathbf{1} = \mathbf{1}$, $\mathbf{1}^T\boldsymbol{W} = \mathbf{1}^T$, and $\rho(\boldsymbol{W} - \frac{1}{I}\mathbf{1}\mathbf{1}^T) < 1$, where $\rho(\cdot)$ is the spectral radius [11]. For node i, the first condition reads $w_{ii} + \sum_{j:\,(i,j)\in\mathcal{E}} w_{ij} = 1$.

Hence, we rewrite the weight matrix as $\boldsymbol{W} = \boldsymbol{I} - \boldsymbol{D}$, where $[\boldsymbol{D}]_{ij} = -w_{ij}$ for $(i,j) \in \mathcal{E}$, $[\boldsymbol{D}]_{ii} = \sum_{j:\,(i,j)\in\mathcal{E}} w_{ij}$, and all other elements are zero. Implicitly we have $\boldsymbol{D}\mathbf{1} = \mathbf{0}$. It is seen that the design freedom is reduced to ensure the first condition for convergence. For a symmetric weight matrix the second condition holds automatically if the first one is fulfilled. The third condition implies that $\boldsymbol{D}$ has to be positive semi-definite with only one eigenvalue at zero and the spectral radius of $\boldsymbol{D}$ has to be less than two.

There are different methods for the design of the weights w_{ij}. In [6] they present methods that concentrate on the optimization of the convergence rate under different constraints, e.g., all weights should be constant (in that case we have $\boldsymbol{D} = \epsilon\boldsymbol{L}$ where $\boldsymbol{L}$ is the graph Laplacian and ϵ denotes the global weight). In our numerical simulations we will use their result for the asymptotically best

weights, denoted $\boldsymbol{W}^{\text{CVX}}$ since they are a result of a convex optimization problem. In [8], we derived the mean-square-error (MSE) optimum per-step weights assuming that the second-order statistic of the initial data is available. Our results revealed that for uncorrelated states (i.e., during the first few iterations) the Metropolis-Hastings (MH) weights [12] are close to these optimum weights. The MH weights $\boldsymbol{W}^{\text{MH}}$ are simple to compute in a distributed fashion.

3. ADVECTION-DIFFUSION

3.1. Basic Theory

We start with a brief explanation of the advection-diffusion process [13], which is a combination of a diffusion process and an advection process, i.e., a bulk flow induced by an external velocity field. We consider a quantity $x(\boldsymbol{r},t)$, with $\boldsymbol{r}$ and t denoting space and time, respectively. Assuming that the diffusion coefficient is constant, that the velocity vector $\boldsymbol{v}(\boldsymbol{r})$ is constant over time, and that there are neither sources nor sinks, $x(\boldsymbol{r},t)$ is governed by the advection-diffusion equation

$$\frac{\partial x}{\partial t} = D\nabla^2 x - \boldsymbol{v}\cdot\nabla x\,. \tag{2}$$

Additionally the equation presumes that the flow created by the velocity field is incompressible, which means that there are no points where the velocity field starts or ends (i.e., $\nabla\cdot\boldsymbol{v}=0$). The first term on the right-hand side of (2) is the diffusion part and the second term characterizes the advection.

So far we have ignored boundary conditions, which amounts to a toroidal geometry (cf. [9]). A more realistic scenario is to define the region of interest as the unit square $[0,1]\times[0,1]$. This requires the specification of boundary conditions for the advection-diffusion process. For the diffusion we impose a reflection at the boundary (nothing of the quantity in the unit square gets lost), which is equal to the condition that the first derivative of the normal component has to be zero. For the velocity field, the normal component is assumed to be zero at the entire boundary (nothing is transported outside the unit square).

3.2. Discretization of the advection-diffusion equation

We next describe the discretization of the advection-diffusion equation (2), using a different approach than [9]. We discretize the advection-diffusion equation by using the finite difference method [14]. In particular, our scheme is similar to the second-order centered spatial differencing and first-order forward temporal differencing (Euler) scheme, which is also called FTCS (forward time, centered space); we discretize the first-order derivative slightly differently, however. In particular, we assume that the velocity vector field is sampled between the field sampling points. Our method yields the same result as in [13] where they use control-volume formulation for the discretization which is well tailored for advection-diffusion processes. In [9], in contrast, they are using FTBS (forward time, backward space). Due to lack of space, we only describe the main discretization steps.

The advection-diffusion field is sampled on a two dimensional regular grid, i.e., $r_1 = p\Delta r$ and $r_2 = q\Delta r$ with $p,q\in\mathbb{N}$. Moreover, time is discretized as $t=k\Delta t$ with $k\in\mathbb{N}$. Hence, we obtain the sample points $x[p,q,k]=x(p\Delta r, q\Delta r, k\Delta t)$ from the continuous field $x(r_1,r_2,t)$. The velocity vector is sampled between the grid points, which is motivated by the fact that the velocity field determines the interaction of two points. The velocity field $\boldsymbol{v}(\boldsymbol{r}) = [v_1(\boldsymbol{r})\ v_2(\boldsymbol{r})]^T$ is thus sampled in the two main directions of the grid as $v_1[p,q]=v_1((p+\frac{1}{2})\Delta r, q\Delta r)$, and $v_2[p,q]=v_2(p\Delta r,(q+\frac{1}{2})\Delta r)$. For the moment, we assume a periodic spatial field, which means that there are no boundaries. The discretization of the advection part $\boldsymbol{v}\cdot\nabla x = v_1(r_1,r_2)\frac{\partial x(r_1,r_2,t)}{\partial r_1} + v_2(r_1,r_2)\frac{\partial x(r_1,r_2,t)}{\partial r_2}$ is now obtained by replacing derivatives with first-order centered differences,

$$\begin{aligned} v_1(r_1,r_2)\frac{\partial x(r_1,r_2,t)}{\partial r_1} &\approx \frac{v_1(r_1+1/2\Delta r_1,r_2)x(r_1+\Delta r,r_2,t)}{2\Delta r} \\ &- \frac{v_1(r_1,r_2-1/2\Delta r)x(r_1-\Delta r_1,r_2)}{2\Delta r} \\ &= \frac{v_1[p,q]x[p+1,q,k]-v_1[p-1,q]x[p-1,q,k]}{2\Delta r}. \end{aligned}$$

This is the average of the centered difference at $(r_1+1/2\Delta r, r_2)$ and at $(r_1-1/2\Delta r, r_2)$, using the property that $\nabla\cdot\boldsymbol{v}=0$. For the diffusion $D\nabla^2 x = D\frac{\partial^2 x(r_1,r_2,t)}{\partial r_1^2} + D\frac{\partial^2 x(r_1,r_2,t)}{\partial r_2^2}$ we have to approximate the second-order derivatives of the field, e.g., for the r_1 direction

$$\frac{\partial^2 x(r_1,r_2,t)}{\partial r_1^2} \approx \frac{x[p+1,q,k]-2x[p,q,k]+x[p-1,q,k]}{(\Delta r)^2}\,,$$

and similarly for the r_2 direction. This step is the same as in [9]. Finally, the temporal forward difference reads

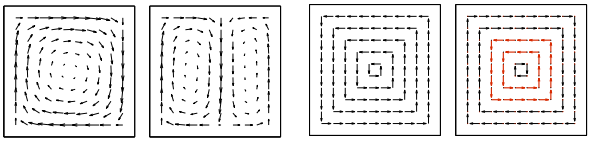
$$\frac{\partial x(r_1,r_2,t)}{\partial t} \approx \frac{x[p,q,k+1]-x[p,q,k]}{\Delta t}\,.$$

With these definitions, and arranging the spatial field samples into a vector $\boldsymbol{x}[k]$, the finite-difference approximation of the advection-diffusion equation (2) amounts to the linear system of equations

$$\boldsymbol{x}[k+1] = (\boldsymbol{I}-\alpha\boldsymbol{D}-\zeta\boldsymbol{C})\boldsymbol{x}[k]\,. \tag{3}$$

Here, the matrix $\boldsymbol{D}$ represents the diffusion and the matrix $\boldsymbol{C}$ (which contains the sampled velocity field) summarizes the advection. Moreover we have $\alpha=\frac{D\Delta t}{\Delta r^2}$ and $\zeta=\frac{v_{\max}\Delta t}{2\Delta r}$, with $v_{\max}$ the maximum velocity at the sampling positions and hence we have $\max|[\boldsymbol{C}]_{ij}|=1$. The ratio ζ/α is called the Peclet number [13]. The matrices $\boldsymbol{D}$ and $\boldsymbol{C}$ are directly obtained by using the discretization results. The matrix $\boldsymbol{D}$ equals 4 on the main diagonal and in each row (or column) there are four elements equal to -1 indicating a link to an immediate neighbor on the sampling grid. The matrix $\boldsymbol{C}$ has the same zero pattern as $\boldsymbol{D}$, with the difference that even the diagonal elements are zero. The values of the non-zero entries depend on the local velocity vectors, i.e., the values $v_1[p,q]/v_{\max}$ and $v_2[p,q]/v_{\max}$ are assigned to $[\boldsymbol{C}]_{ij}$ accordingly. Since the velocity which points from one node to another is the negative of the velocity pointing into the other direction, $\boldsymbol{C}$ is skew-symmetric, i.e., $\boldsymbol{C}=-\boldsymbol{C}^T$; this property does not hold for the discretization in [9].

If the field is constant and there is only diffusion, there is no change in the field over time. In the discretization this amounts to $\boldsymbol{x}[k+1]=\boldsymbol{x}[k]$ and requires $\boldsymbol{D}\mathbf{1}=\mathbf{0}$, which is fulfilled by the discretization. If there is advection, the same condition has to hold for $\boldsymbol{C}$, but this follows also from the incompressible flow condition $\nabla\cdot\boldsymbol{v}=0$. Because of the discretization of the nabla operator we have $\frac{1}{\Delta r}(v_1[p,q]-v_1[p-1,q]+v_2[p,q]-v_2[p,q-1])=0$. This means that the outgoing velocities at each node sum up to zero since $v_1[p-1,q]$ and $v_2[p,q-1]$ denote incoming velocities and hence the sign has to be changed. Hence, it follows that the row-sums and column-sums of $\boldsymbol{C}$ are zero, i.e., $\boldsymbol{C}\mathbf{1}=\mathbf{0}$ and $\mathbf{1}^T\boldsymbol{C}=\mathbf{0}$.



rotation fields $\nabla \times \boldsymbol{v} = c$ regular rotation fields

Fig. 1. Examples of different rotational velocity fields.

4. BLENDING IN WSN WITH UNIFORM GRID TOPOLOGY

4.1. Relation between AC and advection-diffusion

We can interpret the discrete advection-diffusion process (3) as an AC algorithm by interpreting the sampling positions as sensor positions and the field samples as AC states. Specifically, comparing (1) and (3) formally yields $\boldsymbol{W} = \boldsymbol{I} - \alpha\boldsymbol{D} - \zeta\boldsymbol{C}$. The update equation (3) requires only local computation, i.e., each sampling point which is equal to one sensor node uses the values of the direct neighboring sampling points (four in total, because there are two in each direction). Hence, the updates are processed on a 4-regular toroidal (or periodic) graph where the points are located on a regular grid. A comparison of the graph Laplacian and the discretized Laplacian operator (∇^2) of the advection-diffusion equation reveals that they are equal, i.e., $\boldsymbol{L} = \boldsymbol{D}$. Additionally, if $\alpha = \epsilon$ the advection-diffusion process (with $\zeta = 0$) is equal to AC with constant weights. For the case $\zeta \neq 0$, there is an additional component $\zeta\boldsymbol{C}\boldsymbol{x}[k]$ in the update equation. Because of $\boldsymbol{C}\mathbf{1} = \mathbf{0}$ and $\mathbf{1}^T\boldsymbol{C} = \mathbf{0}$ this term does not influence the average of the state vector and the fact that any constant vector is a fixed point of AC. The term $\zeta\boldsymbol{C}$ acts like a blender on the data and influences the convergence. However, if ζ is too large, the iterations become unstable.

4.2. Design of the velocity field

In [9] the velocity field was designed according to the condition $\boldsymbol{v}(r_1, r_2, t) = [v_1(r_2, t)\ v_2(r_1, t)]^T$, which simplifies the derivation of theoretical results but restricts the setting to toroidal topologies. For practical WSN with boundaries, this is not feasible and hence we need different velocity fields. A natural choice are rotational vector fields. Inspired by physics it is possible to generate vector fields that fulfill $\nabla \times \boldsymbol{v} = c$ (where $c \in \mathbb{R}$ is arbitrary) under the constraint that $\nabla \cdot \boldsymbol{v} = 0$. Some examples are shown in Fig. 1, where it is seen that the absolute value of the velocity decreases when approaching the center. Since the design of such rotational fields is involved, we use a modification which we call regular rotational vector field. Here, each velocity vector has the same length and each vector points in one of the two main directions. Realizations of regular rotational fields can be found in Fig. 1. It is seen that the vectors in the center have the same length as those at the boundary and therefore entail stronger advection. It is also possible to change the velocity direction of some of the cycles, which intuitively provides an improved blending. Performance results are provided in Section 6.

4.3. Convergence

In this section we provide sufficient conditions for the convergence of AC with advection for regular rotation fields. To that end, we follow [10], where the so-called von Neumann analysis [15] is applied to test the stability of the FTCS method. With this analysis it can be checked whether any averaging error $\boldsymbol{x}[k] - \bar{s}\mathbf{1}$ decays to zero. Since the error can be decomposed into Fourier modes with non-zero spatial frequency, it suffices to consider those modes separately. The Fourier modes are given by

$$m[p, q, k] = \xi^k \exp\{\iota(p\theta_1 + q\theta_2)\}\,,$$

where ι denotes the imaginary unit and $\theta_m = k_m\Delta x$, $m = 1, 2$, with k_m denoting the wave number (hence, $-\pi \leq \theta_m \leq \pi$). Inserting these modes into the local discretized update equation (cf. [10], which is similar to Section 4.2), we obtain

$$\begin{aligned}\xi =& 1 - 2\iota(\zeta_1 \sin(\theta_1) + \zeta_2 \sin(\theta_2)) \\ &+ 2\alpha(\cos(\theta_1) + \cos(\theta_2) - 2)\,,\end{aligned}$$

The values ζ_1 and ζ_2 represent the velocities in the two spatial directions. To ensure convergence we need to guarantee that $|\xi|^2 < 1$ for all θ_1 and θ_2, which means that all spatial frequency modes decay as the iterations progress. The results of [10] require that $\alpha < 1/4$ and $\zeta_1^2 + \zeta_2^2 < \alpha/2$, which are sufficient and necessary conditions for von Neumann stability for the case that α and ζ_m are spatially constant and that the sampling points lie on a torus.

In our case, we have the FTCS method and assume regular rotational velocity fields (but still on a torus). For this setting, we arrive at the same result under the assumption that one of the two ζ is zero since only two of the possible four velocities around each node are zero and these two velocities have to be equal and are represented by the nonzero ζ. Hence, we obtain $\alpha < 1/4$ and $\zeta^2 \leq \alpha/2$ as sufficient conditions for convergence (the first one is also necessary). We will see in Fig. 2 that the real stability bound described by the second condition is a bit off from these sufficient conditions, which can be explained by the fact that the cycles on which spatial waves are propagated and amplified through advection have limited size and therefore in particular small spatial frequencies are attenuated by the topology.

5. BLENDING IN WSN WITH ARBITRARY TOPOLOGY

So far we considered rotational fields on regular grid graphs, whose practical importance is limited. Hence, we next attempt to generalize the idea of blending the data via rotational advection fields to arbitrary WSNs in order to improve the AC performance. To this end we start with an AC diffusion matrix $\boldsymbol{D}$ and add an optimized advection matrix $\boldsymbol{C}$ to improve performance. First, we show how to analytically construct the matrix $\boldsymbol{C}$ for any given graph and velocity vector and then we present an approach to design appropriate velocity fields.

The velocity vector $\boldsymbol{v}$ has $|\mathcal{E}|$ elements, where each element defines the velocity for one edge (the sign defines the direction). To construct the advection matrix $\boldsymbol{C}$ we need the incidence matrix of the underlying graph $\mathcal{G}$as defined in [8]. The incidence matrix $\boldsymbol{B}$ comes with a permutation matrix $\boldsymbol{P}$ such that the adjacency matrix of the graph can be written as $\boldsymbol{A} = \boldsymbol{B}\boldsymbol{P}\boldsymbol{B}$ and the graph Laplacian reads $\boldsymbol{L} = \boldsymbol{B}\boldsymbol{B}^T - \boldsymbol{B}\boldsymbol{P}\boldsymbol{B}^T$. We restrict to the case where the permutation matrix is defined as $\boldsymbol{P} = \boldsymbol{I} \otimes (\boldsymbol{E} - \boldsymbol{I})$, where $\boldsymbol{E}$ denotes the all one matrix. Additionally we need the matrix $\boldsymbol{F} = \boldsymbol{I} \otimes [1\ -1]^T$. With these matrices it is possible to construct proper advection matrices $\boldsymbol{C}$ according to

$$\boldsymbol{C} = \boldsymbol{B}\boldsymbol{P}\,\mathrm{diag}\{\boldsymbol{F}\boldsymbol{v}\}\boldsymbol{B}^T\,. \tag{4}$$

In general the incompressible flow condition is not automatically fulfilled for any such $\boldsymbol{C}$, i.e., it has to ensured via the design of the

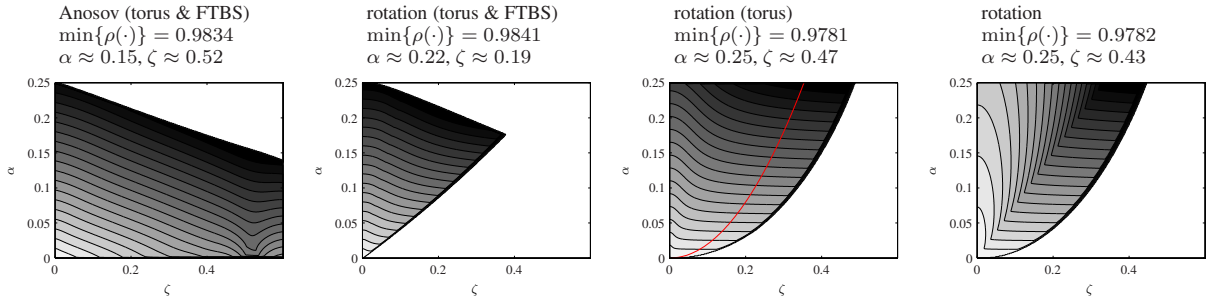


Fig. 2. *Spectral radius for different advection and diffusion parameters on uniform grid WSN with* 625 *nodes (the additional line in the third plot indicates the theoretical sufficient condition for convergence).*

velocity vector $\boldsymbol{v}$. The construction in (4) shows that $\boldsymbol{C}$ depends linearly on $\boldsymbol{v}$.

The velocity vector is obtained by solving the optimization problem,

$$\begin{aligned} &\text{minimize } f(\boldsymbol{v}) \\ &\text{subject to } \boldsymbol{C}\mathbf{1} = \mathbf{0}, \\ &\qquad \left\| \boldsymbol{I} - \boldsymbol{D}^{\text{AC}} - \boldsymbol{C} - \frac{1}{N}\mathbf{1}\mathbf{1}^T \right\|_2 < 1, \end{aligned} \tag{5}$$

where $f(\boldsymbol{v})$ is a suitable convex function (see below), chosen to increase the advection of the resulting algorithm. The first constraint ensures an incompressible flow and the second constraint enforces convergence. The matrix $\boldsymbol{D}$ has to be prescribed (e.g., using the standard designs mentioned in Section 2.2) and has to fulfill all AC convergence conditions. The stability constraint (5) can be replaced with the more stringent condition $0 \leq w_{ij} \pm v_{ij} \leq 1$ for all $(i,j) \in \mathcal{E}$, which is motivated by results in [10]. The resulting optimization problem contains only local conditions. We therefore expect that for specific target functions this problem can be solved in a distributed manner.

To obtain a suitable objective function $f(\boldsymbol{v})$ we again use the idea of rotation fields. Essentially, we impose an initial target velocity vector that mimics a rotational field and use as convex objective function the Euclidean distance to this target velocity field, i.e., $f(\boldsymbol{v}) = \|\boldsymbol{v} - \boldsymbol{v}_{\text{init}}\|_2$. We generate the target velocity vector $\boldsymbol{v}_{\text{init}}$ using the direction of each edge with respect to a circular reference field. In particular we consider edge l which links node i and node j. Their positions $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ are relative to the center of the region in which the nodes are distributed. Hence we obtain,

$$[\boldsymbol{v}_{\text{init}}]_l = \frac{2\boldsymbol{r}_i^T \boldsymbol{T} \boldsymbol{r}_j}{\text{E}\{|\boldsymbol{r}_i - \boldsymbol{r}_j|\}\text{E}\{|\boldsymbol{r}_i + \boldsymbol{r}_j|\}}, \qquad \boldsymbol{T} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix},$$

where $\text{E}\{\cdot\}$ denotes the expected value.

6. NUMERICAL RESULTS

The spectral radius $\rho(\boldsymbol{W} - \frac{1}{N}\mathbf{1}\mathbf{1}^T)$ is an indicator for the convergence speed for AC (cf. [6]). Therefore, we show the impact of the diffusion and advection component on the spectral radius in Fig. 2 for different advective flows in uniform grid WSN with 625 nodes. Besides Anosov flow [9] we used rotational velocity fields where 12 rotation rings are grouped into four groups of three, each group rotating in the opposite direction (see the rightmost field in Fig. 1). It is seen that for FTBS discretization the regular rotational flow yields almost the same minimum spectral radius as the periodically changing Anosov flow, but for our discretization proposed in Section 4.2, the minimum spectral radius decreases slightly. For the non-toroidal geometry with boundaries, the minimum spectral radius hardly changes. Moreover we can conclude that the proposed sampling performs best with maximum diffusion ($\alpha \approx 1/4$) and advection. Numerical MSE results (not shown) corroborate these findings and demonstrate even stronger performance improvements with our proposed method.

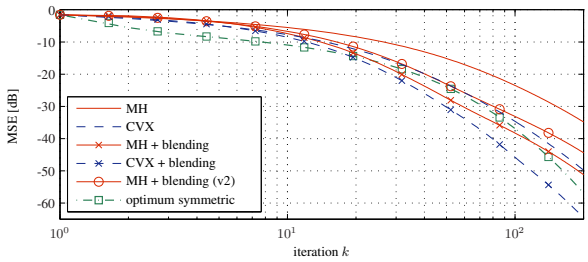


Fig. 3. *Convergence behavior for different weights for random geometric graphs with* $I = 100$ *and initial low-pass fields with cut-off frequency* 0.1.

We next consider WSN modeled by random geometric graphs with 100 nodes (each node has a communication range of 0.2). The sensors measure and average a low-pass field using various AC algorithms with different weight designs, augmented with the blending of Section 5. In Fig. 3 we have plotted the MSE (averaged over 500 scenarios) versus the number of iterations. It is seen that blending improves the performance for MH and CVX weights by about 10dB after 100 iterations (similar gains were observed also for other low-pass fields with different characteristic). Even applying our modified stability constraint (labeled 'v2' in Fig. 3) yields excellent performance. Comparing the curves to the per step MSE-optimum results of [8] let us conclude that the advection approach (with CVX weights) clearly outperforms the best symmetric weights. We also tested blending for dynamic AC [16] and observed a similar performance increase (results not shown).

7. CONCLUSIONS

We have shown that through an appropriate discretization method it is possible to augment classical AC schemes with rotational flows, thereby achieving blending schemes that achieve a faster mixing of the data. Compared to [9], our approach requires neither toroidal geometries nor a time-varying flow. Our numerical results verify the superiority of rotational flows. In particular their use in general WSNs provides a tremendous performance increase and even outperforms the per-step optimum symmetric weights.

REFERENCES

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

[2] V. Schwarz and G. Matz, "Distributed reconstruction of time-varying spatial fields based on consensus propagation," in *Proc. IEEE ICASSP-2010*, Dallas, TX, March 2010, pp. 2926–2929.

[3] J. N. Tsitsiklis, *Problems in Decentralized Decision making and Computation*, Ph.D. thesis, Massachusetts Institute of Technology, Dec. 1984.

[4] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*, Englewood Cliffs, NJ: Prentice-Hall, 1989.

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

[6] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[7] V. Schwarz and G. Matz, "Nonlinear average consensus based on weight morphing," in *Proc. IEEE ICASSP-2012*, Kyoto, JP, March 2012, pp. 3129–3132.

[8] V. Schwarz and G. Matz, "Mean-square optimal weight design for average consensus," in *Proc. IEEE SPAWC-2012*, Cesme, TR, June 2012, pp. 374–378.

[9] S. Sardellitti, M. Giona, and S. Barbarossa, "Fast distributed average consensus algorithms based on advection-diffusion processes," *IEEE Trans. Signal Processing*, vol. 58, no. 2, pp. 826–842, Feb. 2010.

[10] A. C. Hindmarsh, P. M. Gresho, and D. F. Griffiths, "The stability of explicit Euler time-integration for certain finite difference approximations of the multi-dimensional advection-diffusion equation," *International Journal for Numerical Methods in Fluids*, vol. 4, no. 9, pp. 853–897, 1984.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 2nd edition, 1989.

[12] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," *SIAM Review*, vol. 46, no. 4, pp. 667–689, Dec. 2004.

[13] Suhas V. Patankar, *Numerical Heat Transfer and Fluid Flow*, Hemisphere, Washington, DC, 1980.

[14] H. Lomax, T. H. Pulliam, and Zingg, *Fundamentals of computational fluid dynamics*, vol. 246, Springer Berlin, Berlin, GER, 2001.

[15] J. G. Charney, R. Fjörtoft, and J. von Neumann, "Numerical integration of the barotropic vorticity equation," *Tellus*, vol. 2, no. 4, pp. 237–254, 1950/2010.

[16] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, issue 2, pp. 322–329, Feb. 2009.