

Which Distributed Averaging Algorithm Should I Choose for my Sensor Network?

Patrick Denantes^{†*}, Florence Bénézit*, Patrick Thiran*, Martin Vetterli*

*School of IC, EPFL, Lausanne CH-1015, Switzerland

[†]Associate Institute for Signal Processing, TUM, Munich D-80290, Germany

Abstract

Average consensus and gossip algorithms have recently received significant attention, mainly because they constitute simple and robust algorithms for distributed information processing over networks. Inspired by heat diffusion, they compute the average of sensor networks measurements by iterating local averages until a desired level of convergence. Confronted with the diversity of these algorithms, the engineer may be puzzled in his choice for one of them. As an answer to his/her need, we develop precise mathematical metrics, easy to use in practice, to characterize the convergence speed and the cost (time, message passing, energy...) of each of the algorithms. In contrast to other works focusing on time-invariant scenarios, we evaluate these metrics for ergodic *time-varying* networks. Our study is based on Oseledec's theorem, which gives an almost-sure description of the convergence speed of the algorithms of interest.

We further provide upper bounds on the convergence speed. Finally, we use these tools to make some experimental observations illustrating the behavior of the convergence speed with respect to network topology and reliability in both average consensus and gossip algorithms.

I. INTRODUCTION

A. Problem statement.

We study two classes of iterative distributed algorithms, which compute the average of measurements in a sensor network: *average consensus* and *gossip algorithms*. The sensors and the connections between them are unreliable in general, thus the network structure varies over time. While average consensus is a synchronized algorithm, where at each iteration, all the nodes of the network update their current estimate by computing a weighted average of the estimates of their neighbors, gossip algorithms are asynchronous; at each iteration, only one random node wakes up and randomly chooses another node. The two nodes exchange their estimates and update them to the average of the two. Both algorithms are designed to deal with unstructured, unreliable networks, and do not require any knowledge of the network structure or size at the nodes.

In both algorithm classes, one can think of numerous specific algorithms, with a number of design parameters, e.g. weighing factors. A careful choice of the design parameters is crucial, because they have a strong influence on the performance of the algorithms. The goal of this paper is to provide a precise framework to decide on which algorithm to choose in order to obtain a given performance. The critical features this framework requires are an accurate definition for the algorithm's cost and speed of convergence, as well as an easy way to measure these quantities. The challenge here is to harness the time-variability of the network. The ambition is to provide performance and cost assessment metrics, which are well-defined and well-behaved even in the presence of randomness in the network.

In this paper, "network" refers to any kind of network, wired or wireless, with static or mobile nodes, and especially to networks with changing topology. These changes over time may be due to interference, noise, node failures or sleep modes, etc.

B. Related work.

Average consensus in static, time-invariant networks has received considerable attention. In particular, optimizing the convergence speed over the averaging weights is a well-known problem called "fastest

mixing chain” [1], [2]. In this context, mixing chain theory [3], [4] states that the speed of convergence is governed by the second largest eigenvalue in magnitude $\lambda_2(\mathbf{W})$ of the weight matrix \mathbf{W} (the matrix gathering the averaging weights). The fastest mixing chain is the matrix \mathbf{W} which minimizes $\lambda_2(\mathbf{W})$. As shown in [1], the optimal weights for a given fixed network can be computed with a semidefinite program. This weight optimization can be useful in reliable networks, and we explain in Section II-C how to adapt the weights in case of link failures. However, the studies cited above only considered synchronous algorithms in the case where the network structure remains fixed over time.

When considering time-varying networks and/or randomized gossip algorithms however, the weight matrix \mathbf{W} changes over time and the mixing chain theory cannot be applied anymore [5]. For gossip algorithms, a notion of convergence speed called ϵ -averaging time has been introduced in [6], [7]. It is defined for any $\epsilon > 0$ as the earliest time at which the estimates are ϵ -close (in 2-norm) to the true average with probability greater than $1 - \epsilon$. Although this probabilistic definition seems to be well suited to study randomized algorithms, evaluating probabilities numerically may be difficult in practice, as it requires numerous samples and thus multiple experiments.

Further, Kashyap and al. [8] consider another type of randomized averaging algorithms, in which the values exchanged between the nodes are constrained to be integers. In this way, the finite capacity of the transmission channels is taken into account. Still further, Blondel and al. [9] studied the case of delays in the transmission, i.e. the values exchanged between nodes are evaluated only at later iterations. In both cases, only fundamental stability and convergence results are presented. Overall, little work has been done for a quantitative study of convergence speed, which would facilitate the comparison of different averaging algorithms, especially in the application relevant case of time-varying network topologies.

C. Contributions and outline of the paper

In Section II, we first describe the two classes of averaging algorithms (average consensus and gossip), along with the time-varying model of sensor network studied in this paper, where random link failures and/or the node wake-up process require us to use a sequence of randomly time-varying weight matrix $\mathbf{W}(t)$. We assume this sequence to be stationary and ergodic, but not necessarily independent and identically distributed (i.i.d). We also summarize the conditions for convergence of these algorithms in this section.

Next, we study the convergence speed of these algorithms in Section III. In the decrease of an error term over the iterations of the averaging algorithm, we distinguish a steady-state or stationary phase from an initial transient phase. Applying Oseledec’s theorem, we prove that the error decreases exponentially fast to zero in the steady-state phase, at a *contraction rate* γ that is almost surely (a.s.) a *deterministic* constant, despite the randomness of the sequence of weight matrices. Moreover, this rate does not depend on the norm used to measure the error, and furthermore, characterizes similarly the speeds of convergence in mean, in mean square and with probability 1. A third appealing feature is that γ can be easily measured for finite sized networks using a single run of the algorithm, unlike the notion of ϵ -averaging times described in [6], [7]. Finally, other metrics such as energy, power consumption, amount of transmitted messages are directly related to the contraction rate and enjoy therefore the same four properties. We gather these metrics under a general notion of consensus cost, defined as the cost (i.e. time, energy, etc) spent to divide the error by a factor e in the stationary regime.

In Section IV, we compute two upper bounds on the contraction rate γ , in the case of an i.i.d. sequence of weight matrices $\mathbf{W}(t)$.

We give some practical guidelines to measure consensus cost and apply them to real data in Section V. We use data gathered on SensorScope [10], [11], a wireless sensor network deployed in the Communication Systems building at EPFL, to compute the consensus cost (time) in a real world time-varying network. The results show that the Sensorscope network fairly accurately matches the ergodic model, and therefore illustrate the practical applicability of our analysis.

How to choose the best strategy for a given network is explained in Section VI, where we treat several representative scenarios. This comparison is not exhaustive, but shows how the results of Sections III and IV can be used as methodological tools to evaluate, in practice, the performance of different distributed averaging algorithms in sensor networks.

II. AVERAGE CONSENSUS AND GOSSIP MODEL SETUP.

A. Set up and notations.

We consider a time-varying network of n nodes, whose goal is to make available to each node the average value of the measurements of all nodes in the network, or at least a good approximation of it. To this end, at time slot t (t is discrete), the nodes can communicate with each other over all currently active graph edges, or communication links.

We will restrict the exchanged messages to contain only a current estimate $x_i(t)$ of the sending node i , and in some cases limited information about the degree of the sending node and its neighbors. At each time step t , every node i may perform an update operation of its estimate $x_i(t)$ of the overall average. This operation is linear, and relies only on the current average estimates from node i and from its neighbors. The update equation for node i at time t then reads, for $1 \leq i \leq n$,

$$x_i(t+1) = w_{ii}(t)x_i(t) + \sum_{j \in \mathcal{N}_i(t)} w_{ij}(t)x_j(t), \quad (1)$$

where $w_{ij}(t)$ are the weighing factors gathered in a weight matrix $\mathbf{W}(t)$ such that $\mathbf{x}(t+1) = \mathbf{W}(t)\mathbf{x}(t)$, where $\mathbf{x}(t) = [x_1(t); \dots; x_n(t)]^T$. The weights values are set according to averaging algorithms described later on, and $\mathcal{N}_i(t)$ is the current active neighborhood of node i , i.e. the set of nodes which have an active link to node i at time t . $x_i(0)$ is the initial measurement at node i and $x_{ave} := \mathbf{1}^T \mathbf{x}(0)/n = \mathbf{1}^T \mathbf{x}(t)/n$ denotes the true average, where $\mathbf{1} = [1; \dots; 1]^T$ is the vector with all ones.

B. Conditions for convergence to true average.

We denote by \mathbf{J}_n the $n \times n$ averaging matrix with all elements $J_{k,l} = 1/n$, and by $\|\cdot\|_2$ the spectral norm. It is also useful to define the matrix $\mathbf{A}(t) = \mathbf{W}(t) - \mathbf{J}_n$ and the vector of the estimation errors $\epsilon(t) = \mathbf{x}(t) - x_{ave}\mathbf{1}$. The algorithm converges almost surely (a.s.) if $\mathbb{P}[\lim_{t \rightarrow \infty} \epsilon(t) = 0] = 1$. There are two *necessary* conditions for convergence:

$$\begin{aligned} \mathbf{1}^T \mathbf{W}(t) &= \mathbf{1}^T \\ \mathbf{W}(t) \mathbf{1} &= \mathbf{1}, \end{aligned} \quad (2)$$

which respectively ensure that the average is preserved at every iterations, and that $\mathbf{1}$ is a fixed point. Conditions for convergence in expectation and in mean square can be found in [6]. We present here *sufficient* conditions for a.s. convergence and convergence in second moment, in the case where $\{\mathbf{W}(t)\}_{t \geq 0}$ is stationary and ergodic:

- Conservation properties: conditions (2).
- Contraction property: $\|\mathbf{W}(t)\|_2 \leq 1$.
- Connectivity property: $\mathbb{E}[T_\eta] < \infty$, where $T_\eta := \inf_t \{t \geq 1 : \prod_{p=0}^t \mathbf{W}(t-p) \geq \eta > 0\}$ is a stopping time. In other words, there can be isolated nodes at any iteration, but every node has to eventually connect to the network, which has to be jointly connected.

This result was recently proved in [12].

C. Different averaging strategies.

There are two main classes of distributed averaging algorithms:

- 1) *Synchronous algorithms* or average consensus. All the nodes activate at each time slot t , communicate with their neighbors and update their current state. There are three main average consensus strategies for the choice of the weights.

a) *Uniform weights*: [5]

$$w_{ij}(t) = \begin{cases} \alpha & \text{if } j \in \mathcal{N}_i(t) \\ 1 - \alpha |\mathcal{N}_i(t)| & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

where α is a constant small enough for the scheme to be stable, and $|\cdot|$ denotes cardinality.

b) *Metropolis weights*: [5]

$$w_{ij}(t) = \begin{cases} \frac{1}{1 + \max\{|\mathcal{N}_i(t)|, |\mathcal{N}_j(t)|\}} & \text{if } j \in \mathcal{N}_i(t) \\ 1 - \sum_{k \in \mathcal{N}_i(t)} w_{ik}(t) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

c) *Adapted optimal weights*: Suppose that, when fully working, the network is a known graph G . If, at time t , link failures occur in G , then our network at time t is only a subgraph of G . According to the fastest mixing chain theory, we can compute the optimal weight matrix Ω^G for the graph G : Ω^G performs with optimal speed on the *time-invariant* graph G . Since link failures change over time, our network is *time-varying*, and we cannot use these weights as such; however, we want our time-varying weights to be inspired by the optimal time-invariant weights. An easy way to take advantage of our knowledge of the optimal time-invariant weights is given here, and we show in Section VI-B that this strategy can considerably speed up the averaging process in reliable networks. We adapt the weights ω_{ij}^G (the coefficients of the weight matrix Ω^G) so that they respect the link failures and yet fulfill conditions (2). More precisely, for all working links (ij) , we take $w_{ij} = \omega_{ij}^G$. If link (kl) , existing in G , fails, then $w_{kl} = 0$ instead of $w_{kl} = \omega_{kl}^G$, because node k and node l cannot exchange their estimates anymore. To keep the overall sum of estimates unchanged, we adapt w_{kk} and w_{ll} such that the weights at each node always sum to 1. This can be achieved locally at each node, and adapted optimal weights are thus a distributed averaging strategy.

$$w_{ij}(t) = \begin{cases} \omega_{ij}^G & \text{if } j \in \mathcal{N}_i(t) \\ 1 - \sum_{k \in \mathcal{N}_i(t)} w_{ik}(t) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

2) *Asynchronous algorithms* or gossip algorithms. At each time slot t , one node only activates. It performs an averaging scheme with one other node. In the most common gossip algorithm [13], the currently active node i chooses one of its neighbors j and they both update their estimates x_i and x_j with their average $(x_i + x_j)/2$:

$$\begin{aligned} w_{ij}(t) &= w_{ji}(t) = w_{ii}(t) = w_{jj}(t) = 1/2 \\ w_{kk}(t) &= 1 && \text{if } k \neq i, j \\ w_{kl}(t) &= 0 && \text{on all other edges.} \end{aligned} \quad (6)$$

In geographic gossip, the information is routed through the network to allow also non-neighboring nodes to average their values [14]. A possible extension could allow three or more nodes to average their values at each step [15]. Another variation would be to choose weights other than $1/2$ in pairwise gossip for example. See Section VI-C for further details.

D. Stationary and ergodic networks

It is important to notice that all described strategies present *time-varying* averaging weights. Moreover, these changes are *random*. That is, we can see the sequence of averaging matrices as the realization of a random process $\{\mathbf{W}(t)\}_{t \geq 0}$. From there, it might seem difficult to define a *deterministic* convergence speed, which is observed not only in a particular run, but repeatedly in almost every¹ realization of the process $\{\mathbf{W}(t)\}_{t \geq 0}$. We show in Section III that this is in fact possible, when this process satisfies two conditions: stationarity and ergodicity. Stationarity means invariance under time shifts. Ergodicity usually comes with stationarity, but is a little bit more subtle. Refer to the book of Walters [16] for a precise definition. These conditions are actually very broad, and easily satisfied by most network models.

In particular, let us see how they translate when using the considered averaging algorithms:

1) *Synchronous algorithms*: The algorithms *try* to perform the same operation at every iteration. The only source of time-variability, and therefore of randomness, is in the variations of the network topology. These variations are due to a number of factors, e.g. node failures, transmission channel quality, node sleeping modes, etc. Thus, the combination of all these effects should be stationary and ergodic over time.

2) *Asynchronous algorithms*: Here, randomness is additionally introduced by the algorithm itself. But this process can easily be controlled to be stationary and ergodic, for example if nodes wake up and choose their neighbor in an i.i.d. manner.

Whether $\{\mathbf{W}(t)\}_{t \geq 0}$ is actually stationary and ergodic in real life is the object of Section V-B, where we run gossip on actual wireless sensor data. In the simulations, we use a model where links fail independently with probability p at each iteration t . Thus, the $\mathbf{W}(t)$ are all i.i.d., and this is a special case of a stationary and ergodic process.

As a conclusion, unlike previous work, this paper embraces two different sources of randomness, the network topology variations and the randomized algorithm itself, which we aggregate into a single stochastic process $\{\mathbf{W}(t)\}_{t \geq 0}$. Our analysis is valid as long as this process is stationary and ergodic, a condition satisfied by most natural network and algorithm definitions.

III. CONSENSUS COST AND CONSENSUS TIME.

In Section II-B we recalled the conditions for a consensus algorithm to converge, i.e. for the error $\epsilon(t) = \mathbf{x}(t) - x_{ave}\mathbf{1}$ to become arbitrarily small when the number of iterations t is taken large enough. From now on, we assume the a.s. convergence conditions to be satisfied, and we study the cost of convergence itself, in terms of number of iterations, energy, power, etc., as we will see in Section VI-A. Let $C(t)$ be the total cost of the algorithm up to iteration t . We assume the increments of $C(t)$ at each iteration to be a nonnegative function of $\mathbf{W}(t)$, so that the iteration costs are also stationary and ergodic. We want to express the error $\epsilon(t)$ as a function of $C(t)$, independently of the initial measurement $\mathbf{x}(0)$ and of the particular realizations of the network and algorithm. In this section, we prove that, if $\{\mathbf{W}(t)\}_{t \geq 0}$ is *stationary* and *ergodic*, then, for large t , there is a *deterministic* constant C_c , called consensus cost, such that the error $\|\epsilon(t)\| \approx \exp(-C(t)/C_c)$. C_c only depends on the algorithm considered and on the network's statistics.

Definition 1: Consensus cost Let

$$C_c = \lim_{t \rightarrow \infty} -C(t) / \log \|\epsilon(t)\|, \quad (7)$$

where $\|\cdot\|$ denotes any norm equivalent to a p -norm, $p \in [1, \infty]$. Whenever this limit exists, we call consensus cost its value C_c .

The particular case where the consensus cost is time ($C(t) = t$) leads to the following definition:

¹That is, with probability 1.

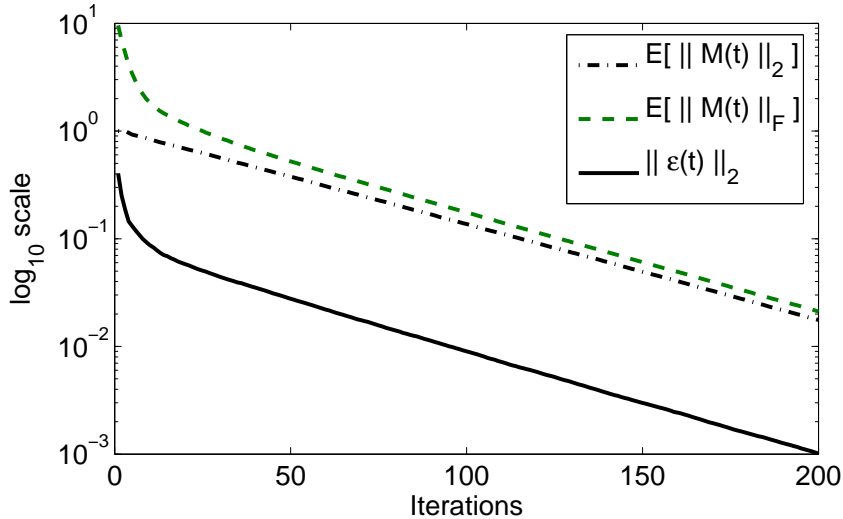


Fig. 1. Residual error for Metropolis average consensus on a random geometric graph with 200 nodes and link probability failure $p = 0.9$. The figure represents in a logarithmic scale the residual error of the estimates as well as the residual error of the averaging matrix in norm 2 and Frobenius norm.

Definition 2: Consensus time Whenever the limit:

$$T_c = \lim_{t \rightarrow \infty} -t / \log \|\epsilon(t)\|. \quad (8)$$

exists, it is called consensus time.

Theorem 1: If $\{\mathbf{W}(t)\}_{t \geq 0}$ is stationary and ergodic, then the limits (7) and (8) exist. Moreover, C_c and T_c are a.s. non-random, independent of the norm $\|\cdot\|$, and $C_c = E[C(1)]T_c$.

This theorem is illustrated by simple observations on Fig. 1. The residual error $\epsilon(t)$ in a logarithmic scale (solid line) slowly decreases linearly after a faster transient phase. Moreover, we notice that the averaging matrix residual $\mathbf{M}_t := \prod_{k=1}^t \mathbf{W}(t-k) - \mathbf{J}_n = \prod_{k=1}^t \mathbf{A}(t-k)$ decreases in expectation with the same rate, which suggests the decreasing rate is deterministic and does not depend on the initial measurement $\mathbf{x}(0)$. Showing this and defining a contraction rate γ as the slope of this linear stationary regime is the object of the next two theorems, which will be used to prove Theorem 1.

A. Contraction rate of the averaging matrix.

It is well known [17, page 299] that for any submultiplicative matrix norm $\|\cdot\|$ (for all A, B , $\|AB\| \leq \|A\|\|B\|$), $\lim_{t \rightarrow \infty} \|\mathbf{A}^t\|^{1/t} = \rho(\mathbf{A})$. If $\rho(\mathbf{A}) \neq 0$, we can rewrite this as $\lim_{t \rightarrow \infty} \frac{1}{t} \log \|\mathbf{A}^t\| = \log \rho(\mathbf{A})$. How does this extend to the case where $\mathbf{A}(t)$ varies over time? Is there any way to specify or to compute an asymptotic contraction rate of *matrices*

$$\lim_{t \rightarrow \infty} \frac{1}{t} \log \left\| \prod_{k=1}^t \mathbf{A}(t-k) \right\|, \quad (9)$$

and when is this quantity well-defined in the first place?

Theorem 2: (Furstenberg and Kesten's Theorem. [18]) Let $\{\mathbf{A}(t)\}_{t \geq 0}$ be a stationary and ergodic sequence of $n \times n$ matrices, and let $\|\cdot\|$ be any submultiplicative matrix norm. If

$$E [\max(0, \log \|\mathbf{A}(0)\|)] < \infty, \quad (10)$$

then, if $\mathbf{M}_t := \prod_{k=1}^t \mathbf{A}(t-k)$ for any t , the limit

$$\lim_{t \rightarrow \infty} \frac{1}{t} \log \|\mathbf{M}_t\|$$

exists, and, is equal with probability 1 to

$$\gamma := \lim_{t \rightarrow \infty} \frac{1}{t} \mathbb{E} [\log \|\mathbf{M}_t\|]. \quad (11)$$

Furstenberg and Kesten's theorem shows that, as we assume $\{\mathbf{A}(t)\}_{t \geq 0}$ to be stationary and ergodic, the asymptotic contraction rate γ is well-defined and is a constant with probability 1. Note that in our case, $\{\mathbf{W}(t)\}_{t \geq 0}$ are contracting matrices because of converging conditions (II-B), which implies that $\{\mathbf{A}(t)\}_{t \geq 0}$ are bounded matrices, and hence condition (10) is fulfilled.

B. Contraction rate of the average estimates.

It is now legitimate to ask how the estimation error *signal* $\|\epsilon(t)\| = \|\mathbf{M}_t \mathbf{x}(0)\|$ behaves. The next theorem [19], [16], [20], [21] states that the error contracts with the same rate γ as in Theorem 2 for almost every initial measurement $\mathbf{x}(0)$. In other words, Theorem 2 can be extended from matrices to signals.

Theorem 3: (Oseledec's Theorem. [19], [21]) Let $\|\cdot\|$ be a norm on \mathbb{R}^n equivalent to a p -norm, with $p \in [1, \infty]$, let $\{\mathbf{A}(t)\}_{t \geq 0}$ be a stationary and ergodic sequence of matrices satisfying condition (10), and $\mathbf{M}_t := \prod_{k=1}^t \mathbf{A}(t-k)$. Then, with probability 1, there is a proper subspace V of \mathbb{R}^n such that, for all $\mathbf{x} \in \mathbb{R}^n \setminus V$,

$$\lim_{t \rightarrow \infty} \frac{1}{t} \log \|\mathbf{M}_t \mathbf{x}\| = \gamma, \quad (12)$$

where γ is defined by (11) in Theorem 2.

Theorem 3 shows therefore that $(\log \|\epsilon(t)\|)/t \rightarrow \gamma$. In the introductory statement, "almost every" starting point \mathbf{x} means all $\mathbf{x} \in \mathbb{R}^n \setminus V$, i.e. all \mathbf{x} outside a proper linear subspace of \mathbb{R}^n . Note that if $\mathbf{A}(t) = \mathbf{A}$ is a constant (i.e. degenerate random variable), then e^γ is the absolute value of the largest eigenvalue of \mathbf{A} in magnitude. It is also interesting to see that γ does not depend on the norm $\|\cdot\|$, and that the subspace V is random. In this paper, we work only with one coefficient γ , which characterizes the speed of the slowest component of the signal, but a full version of the theorem states the existence of other smaller coefficients, called Lyapunov exponents, which affect the transient phase.

Proof: (Theorem 1) Theorems 2 and 3 prove that every average consensus or gossip algorithm converges with some characteristic, deterministic contraction rate γ , asymptotically in time, if the underlying sequence of weights matrices form a stationary and ergodic stochastic process. To define properly consensus time, it suffices to notice that, under these conditions, (8) becomes $T_c = -1/\gamma$. Moreover,

$$\begin{aligned} C_c &= \lim_{t \rightarrow \infty} -C(t)/\log \|\epsilon(t)\| \\ &= \lim_{t \rightarrow \infty} \frac{C(t)}{t} \cdot \lim_{t \rightarrow \infty} \frac{-t}{\log \|\epsilon(t)\|} \\ &= \mathbb{E}[C(1)]T_c. \end{aligned}$$

The last equality comes from the ergodicity of the network, which implies the ergodicity of the cost at each iteration. ■

IV. UPPER BOUNDS ON CONTRACTION RATE.

Theorems 2 and 3 prove the existence of deterministic contraction rate γ , consensus cost and consensus time but do not provide any method to compute them, other than evaluating the limit (11) or (12). In this section, we present upper bounds on $\gamma = -1/T_c$, under the stronger condition that the sequence of weight matrices is i.i.d.

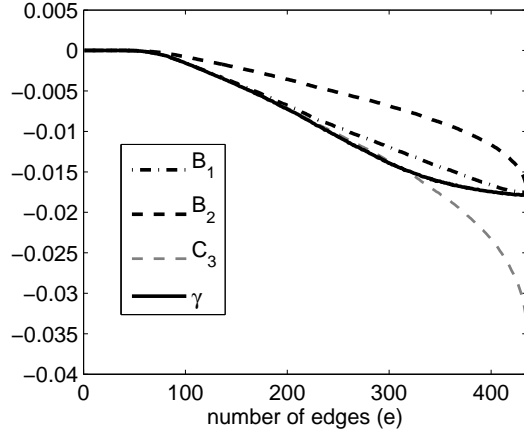


Fig. 2. Impact of the network’s connectivity on the behavior of $B_1 := \log(\rho(\mathbb{E}[\mathbf{A} \otimes \mathbf{A}]))/2$, $B_2 := \log(\lambda_1(\mathbb{E}[\mathbf{A}^T \mathbf{A}]))/2$ and $C_3 := \log \lambda_2 \mathbb{E}[\mathbf{W}]$. Pairwise gossip was run on a random geometric network of 30 nodes and increasing connecting radius, and thus number of edges.

A. A first bound $B_1 := \log(\rho(\mathbb{E}[\mathbf{A} \otimes \mathbf{A}]))/2$.

Theorem 4: Let $\{\mathbf{A}(t)\}_{t \geq 0}$ be a sequence of i.i.d. random matrices in $\mathbb{R}^{n \times n}$ satisfying condition (10), and γ its contraction rate. Then, γ is bounded from above

$$\gamma \leq B_1 := \frac{1}{2} \log \rho (\mathbb{E}[\mathbf{A}(0) \otimes \mathbf{A}(0)]),$$

where ρ denotes the spectral radius, and \otimes the Kronecker product.

The proof is given in Appendix.

B. A looser but simpler bound $B_2 := \log(\lambda_1(\mathbb{E}[\mathbf{A}^T \mathbf{A}]))/2$.

This bound is strongly inspired by previous work [6].

Theorem 5: Let $\{\mathbf{A}(t)\}_{t \geq 0}$ be a sequence of i.i.d. random matrices in $\mathbb{R}^{n \times n}$ satisfying condition (10), and γ its contraction rate. Then, γ is bounded from above

$$\gamma \leq B_2 := \frac{1}{2} \log \lambda_1 (\mathbb{E}[\mathbf{A}^T(0)\mathbf{A}(0)]),$$

where λ_1 denotes the largest eigenvalue.

The proof is given in Appendix. As we can see in Fig. 2, this bound is very loose, except for very well connected networks. The proof of Theorem 4 requires two inequalities, one of which we expect to be an equality, whereas the proof of Theorem 5 uses an infinite number of inequalities. This explains why B_1 is much tighter.

In most gossip algorithms, the matrix \mathbf{W} is a projection matrix: $\mathbf{W}\mathbf{W} = \mathbf{W}$. Indeed, if the same pair of nodes successively average their estimates twice, the second averaging round is useless. In that case, Theorem 5 becomes $\gamma \leq \frac{1}{2} \log \lambda_1 (\mathbb{E}[\mathbf{A}]) = \frac{1}{2} \log \lambda_2 (\mathbb{E}[\mathbf{W}])$. This enables us to adapt previous results based on the estimation of $\lambda_2 (\mathbb{E}[\mathbf{W}])$. These results bound ϵ -averaging time in gossip algorithms running on random geometric graphs of increasing size n , constructed by placing the n nodes uniformly on a unit square, and by connecting any pair of nodes if their distance is smaller than $r(n) = \sqrt{\alpha \log n/n}$, for some constant α large enough to ensure graph connectivity. In this setting, for any function f , if $T_{ave}(\epsilon) = O(f(n) \log(\epsilon^{-1}))$, then $T_c = O(f(n))$. Consequently, if we consider the number of exchanged messages as the cost function C , pairwise gossip with direct neighbors behaves in $C_c = O(n^2/\log n)$ messages [22]; geographic gossip, which averages any pair of nodes at the expense of routing their

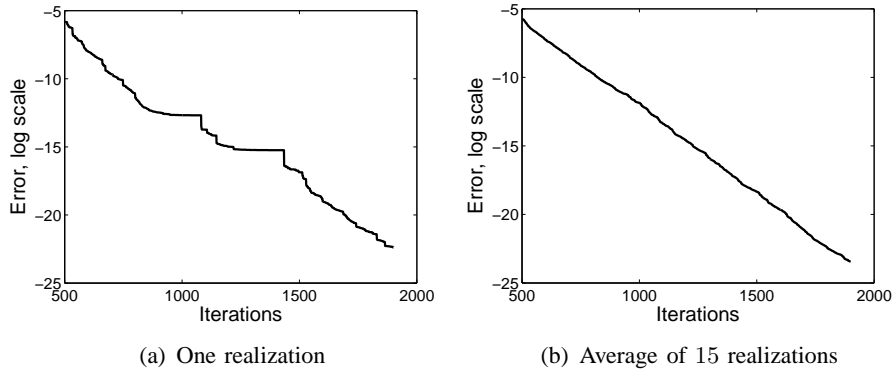


Fig. 3. Geographic gossip with path averaging on a random geometric graph. Links fail i.i.d. in time and space. Fig. (a) represents $\log \|\epsilon(t)\|$ from $t = 500$ to 1900. We reproduced the experiment 15 times and averaged the 15 results to get Fig (b) . In this case we prefer to measure γ on this averaged figure.

estimates, runs with $C_c = O(n^{1.5}/\sqrt{\log n})$ messages [14]; and finally geographic gossip with averaging along the routing path achieves $C_c = O(n)$ messages [15]. Unfortunately, in general, $\lambda_2(\mathbb{E}[\mathbf{W}])$ is *not* a bound on γ .

V. EXPERIMENTAL OBSERVATIONS

A. Practical guidelines to measure contraction rate γ , consensus time T_c and consensus cost C_c .

The method to measure γ , T_c and C_c comes directly from their definitions: $\gamma = \lim_{t \rightarrow \infty} \log \|\epsilon(t)\| / t$, $T_c = -1/\gamma$ and $C_c = \lim_{t \rightarrow \infty} -C(t) / \log \|\epsilon(t)\|$. One should run the algorithm with a known initial signal $\mathbf{x}(0)$, stop the algorithm at some large t_f and measure the error $\epsilon(t_f)$. Then one can compute $\log \|\epsilon(t_f)\| / t_f$ to get an estimate of γ , and consequently of T_c , or measure $C(t_f)$ and calculate $-C(t_f) / \log \|\epsilon(t_f)\|$ as an approximation of C_c . There are two sources of error when estimating these quantities at a finite time t_f : the deviation induced by the transient phase, and noisy excursions from the linear trend of the curve (Fig. 3(a)) introduced by the randomness of the sequences of time-varying matrices, especially when there are relatively isolated nodes in the network which do not participate often in the averaging process. Two tricks, which can be combined, improve the accuracy of the measurements:

- 1) *Erase the transient phase:* Take an extra measurement at an intermediate iteration t_i after the transient phase, and estimate the slope of $\log \|\epsilon(t)\|$ only in the stationary regime: $\gamma \simeq (\log \|\epsilon(t_f)\| - \log \|\epsilon(t_i)\|) / (t_f - t_i)$. Similarly, $C_c \simeq (C(t_f) - C(t_i)) / (\log \|\epsilon(t_f)\| - \log \|\epsilon(t_i)\|)$.
- 2) *Reduce noise:* Run the algorithm several times and estimate γ as the slope of the average of the errors: $\gamma \simeq \langle \log \|\epsilon(t_f)\| \rangle / t_f$, where $\langle \rangle$ denotes averaging over runs. See Fig. 3 for an example. Then $C_c \simeq (-1/\gamma) \langle C(t_f) \rangle / t_f$.

B. Confronting theory with data from SensorScope.

We used data from SensorScope [10], which is a sensor network of approximately 20 static nodes, to validate the theoretical results provided in section III. As explained in the description of SensorScope provided in [11], basic tinyOS multihop routing is used to gather at the sink a picture of the network's connectivity. Every 15 minutes, every node sends the list of its current neighbors. Because this reporting procedure is not reliable, we could only use the data from 16 nodes. On these nodes, we perform pairwise gossip: when a node i communicates its neighbor list to the sink, we uniformly choose one node j in the list, and update nodes i and j 's states to $(x_i + x_j)/2$.

The results are shown in Fig. 4 and are very encouraging. As shown in Fig. 4(b), $\log \|\epsilon(t)\|$ does decrease linearly on average. Realizations taken one by one are more noisy for 2 reasons. First, some nodes have

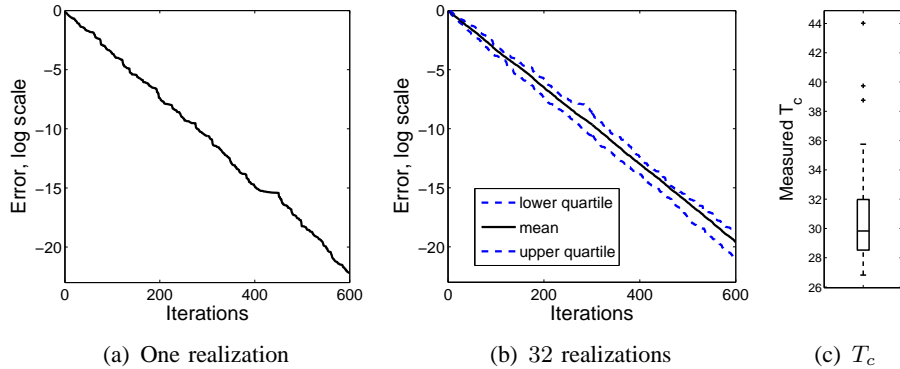


Fig. 4. Pairwise gossip on SensorScope. In (a), (b): the error $\epsilon(t) = \|x(t) - x_{ave}\|_2$ in a log scale, and in (c): consensus time. Pairwise gossip was run on preexistent data measured on a wireless sensor network of 16 nodes at EPFL. Each of the 32 realizations were run with different initial measurements chosen randomly according to a normal distribution.

less neighbors than others, so their participation to the algorithm is unevenly allocated, creating some noisy excursion effects, as mentioned in V-A. Second, Fig. 4(a) represents a day where data reliably reached the sink, but on other days, we do not receive the entire data. Besides amplifying the previously cited noise phenomenon, this also introduces distortion from one day to the other, and the measured consensus times T_c spread out (Fig. 4(c)). Despite that, consensus time concentrates between 28 and 32 iterations, for most of our experiments. We thus conclude that our analysis reasonably holds on SensorScope and that, on this network, gossip algorithm divides the error by a factor e every ≈ 30 iterations. In further work, we will implement pairwise gossip directly in the network, in order to study precisely its ergodicity without distortions due to incomplete data.

VI. CHOICE OF STRATEGY

A. Choice of cost function.

Choosing the strategy will depend on the choice of the cost function C because *one should choose the strategy that minimizes the consensus cost C_c .*

The absolute time of convergence, in contrast with the number of iterations, depends on the clock model used in the network for the algorithm. For example, the duration d of one iteration may be shorter in pairwise gossiping than in average consensus. Consequently, in order to compare strategies on a time criterion, one should compare their absolute consensus times $T_c^{abs} = E[d]T_c$.

From an information processing point of view, it is interesting to know how efficient the successful pairwise exchanges of messages are, as if the network was wired. We call $\mathcal{M}(t)$ their number after t iterations and \mathcal{M}_c the *consensus number of message exchanges*. Note that $\mathcal{M}(t)$ can be measured by counting the total number of messages received by the nodes.

If we are interested in power consumption, we have to use the broadcast model. In average consensus, one node emits the same message to all its neighbors with only one transmission. Let $\mathcal{E}(t)$ be the energy used by all nodes, taken here as the number of messages broadcasted, up to iteration t . We accordingly define *consensus energy* \mathcal{E}_c , which can have a behavior very different from that of \mathcal{M}_c .

B. Examples of comparison between strategies.

a) Gossip or Metropolis?: We want to know whether we should use pairwise gossip or Metropolis weights, depending on the network reliability. We model our sensor network as a random geometric graph with $n = 50$ nodes and connection radius $r = 0.18$, and the reliability of our network is indexed by p , the probability of link failure. We assume the links failures to be i.i.d. in time and in space. Simulations (Fig.

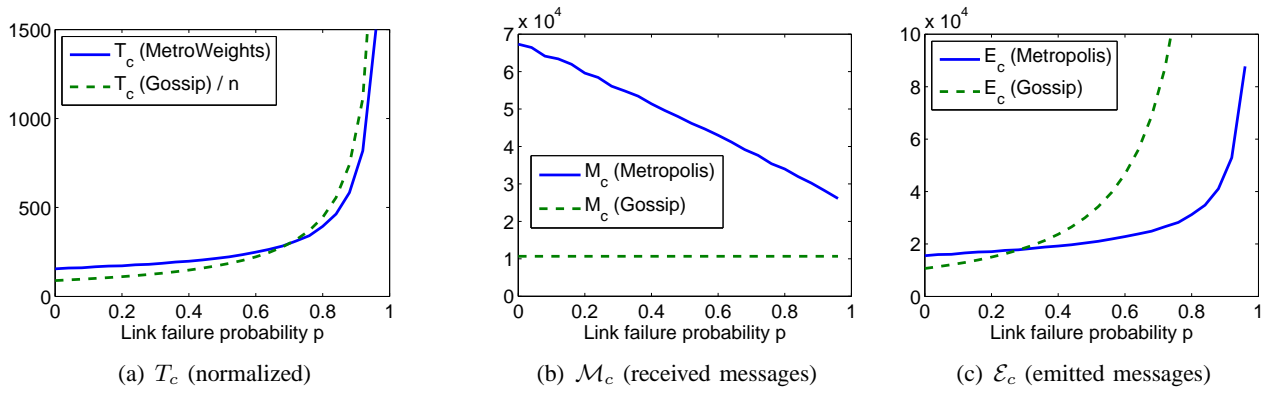


Fig. 5. These experiments have been run on a random geometric graph of 50 nodes on a unit square, with radius 0.18. Fig (a) shows Metropolis and gossip consensus times, the latter being scaled by the number of nodes. In Fig (b) we see that the pairwise communications in Metropolis are more efficient if the network is less reliable. Fig (c) shows that gossip spends less energy than Metropolis when the network is more reliable.

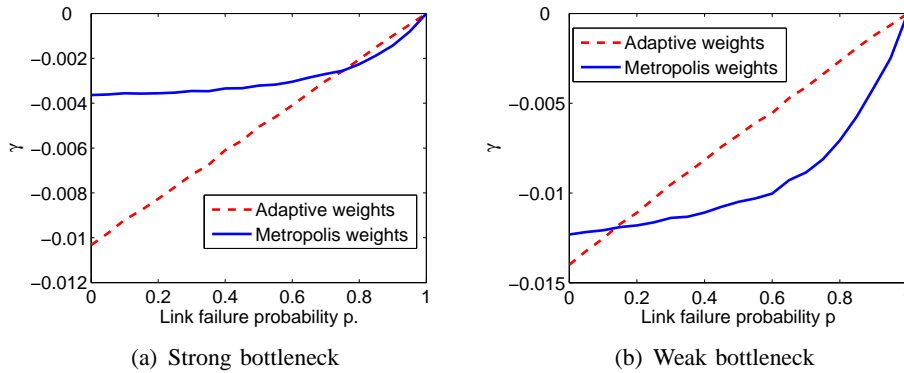


Fig. 6. These two experiments were run on trees with 31 nodes. The root of tree (a) has 3 children that have 9 children each, whereas the root of tree (b) has 10 children that have 2 children each. The root is a much bigger bottleneck in tree (a) than in tree (b). This bottleneck in tree (a) is well taken care of by the weight optimization, whereas Metropolis weights already perform well on tree (b). Therefore the threshold probability is much larger in (a) than in (b).

5) show that in reliable networks, it is better to use pairwise gossiping whereas unreliable networks work better with the Metropolis algorithm. Indeed, at each iteration, the Metropolis algorithm takes advantage of all the existing links, whereas the gossip algorithm bets on one link. If it is off, no averaging occurs and there is a loss of time and energy. The weakness of Metropolis weights in reliable networks is the redundancy of its messages, an effect most visible when considering the received messages (Fig. 5(b)).

b) Metropolis or adapted optimal weights?: Choosing between two synchronous strategies is easier, because they have the same clocks and their message passing schemes are very similar. In our simulations, we compare their contraction rates γ . If the network is reliable (low failure probability p), it is worth computing the optimized weights and adapt them when links fail, as in (5). On the contrary, if p is large, the adapted weights are too low, considering the smaller number of active links, and Metropolis weights perform better. An example is given in Fig. 6, which shows that the threshold probability separating the best alternatives heavily depends on network topology.

C. Choice of pairwise gossip weights.

In previous work, pairwise gossip assumed the weight $\alpha = 0.5$ as mentioned in (6). Here, we show that this weight choice is not the best strategy for regular graphs and random geometric graphs, although

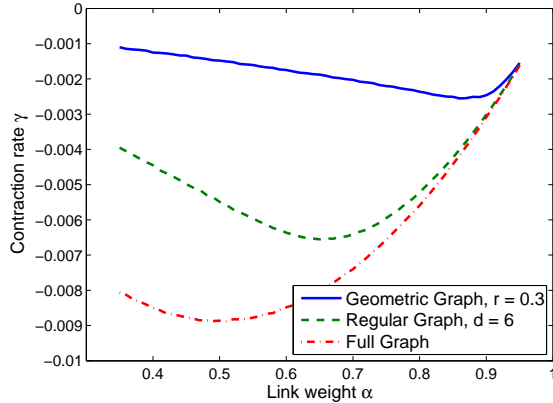


Fig. 7. Search for optimal weights in pair-wise gossip (Eq. (13)). These curves were obtained by averaging the results over 500 graphs of each type. All the graphs have 60 nodes.

it is the best strategy for fully connected graphs. We propose to extend pairwise gossip in the following way. If node i wakes up and calls node j ,

$$\begin{aligned}
 w_{ij}(t) &= w_{ji}(t) = \alpha \\
 w_{ii}(t) &= w_{jj}(t) = 1 - \alpha \\
 w_{kk}(t) &= 1 && \text{if } k \neq i, j \\
 w_{kl}(t) &= 0 && \text{on all other edges.}
 \end{aligned} \tag{13}$$

As shown in Fig. 7, except some special cases including the full graph, it is better to have $\alpha = \alpha_0 > 0.5$. The explanation is intuitive. In a general graph, some nodes are more isolated than others. Even though the nodes wake up the same amount of times in average, they are not called by other nodes on a fair basis. This implies that relatively isolated nodes participate to the algorithm less often than well connected nodes. So capturing a proportion $\alpha > 0.5$ of the value of one’s neighbor enables the “slow” nodes to catch up by giving a large part of their value to the “fast” nodes. However, there is a trade off to be found. If α is too large, the algorithm loses efficiency. In our experiment, we consider graphs with 60 nodes. In this case, the optimal weight for regular graphs of degree $d = 6$ is $\alpha_0 = 0.65$, and $\alpha_0 = 0.87$ for random geometric graphs with radius $r = 0.3$. Finding how the optimal α behaves with the number of nodes and more generally with the topology of graphs is yet to be explored.

VII. SUMMARY AND OPEN PROBLEMS

In this paper, we first provide novel tools that allow to compare performance of different averaging strategies in arbitrary, ergodic time-varying networks. The principal parameter used is the *contraction rate* γ , which can be related to a Lyapunov exponent governing the decay rate of the error vector norm.

A first advantage of this metric is that it can be computed very easily in practice, with a few simulation runs only and using a random initial measurement vector. Theorem 3 proves that the contraction rate γ is a deterministic constant with probability one with respect to realizations of the time-varying network, and for almost any starting point (i.e. initial measurement vector).

More importantly, contraction rate γ is the key parameter to compute consensus cost. The natural definition of γ is with respect to time, or the number of algorithm iterations. But it can readily be modified in order to consider e.g. the number of sent messages or the energy spent. Thus, depending on the cost function to optimize, we consider either a time constant T_c or a cost C_c , respectively characterizing the amount of time or any other resource needed to achieve a certain “consensus” between the nodes in

the network. The values T_c and \mathcal{E}_c may then be used to compare averaging costs of different averaging strategies.

Furthermore, we derive upper bounds on the contraction rate for the case of link failure patterns i.i.d. in time. Although being either loose (B_2) or strenuous to compute numerically (B_1), these bounds have the advantage to require the analysis of a single matrix only, instead of an infinite sequence of matrices. Consequently, they may help to get more insight in averaging dynamics from an analytical point of view, rather than by simulations. According to empirical observations, B_1 seems to be fairly tight in many cases. Still, proving whether and under which conditions this bound is asymptotically tight for large networks remains an open problem.

Finally, we showed a few examples where we used the newly derived tools to analyze different network setups and averaging strategies. We compared pairwise gossiping with synchronized averaging using Metropolis weights, considering different cost functions such as time, the number of transmitted messages, and the number of received messages. Finally, we highlighted the fact that efficiency of pairwise gossip may be improved significantly by putting more weight on the exchanged values than on one's own.

All these observations can help optimizing and improving existing averaging algorithms, and are only examples on how the presented performance measures may be used. We are convinced they will prove very useful in evaluating future averaging algorithms, and comparing them to existing ones.

ACKNOWLEDGMENTS

Many thanks to Prof. Wolfgang Utschick from TU Munich, who made this work possible by sending Patrick Denantes to Lausanne for his Master's Thesis, and to H. Nguyen for providing us the SensorScope data. The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

VIII. APPENDIX.

Proof: (Theorem 4) Theorem 2 gives us the choice of the submultiplicative matrix norm and we choose the Frobenius norm. We denote by m_{ij} the i -th row, j -th column element of \mathbf{M}_t . Then, by Jensen's inequality (15),

$$\gamma = \lim_{t \rightarrow \infty} \frac{1}{t} \mathbb{E} \left[\frac{1}{2} \log \|\mathbf{M}_t\|_F^2 \right] \quad (14)$$

$$\leq \lim_{t \rightarrow \infty} \frac{1}{2t} \log \mathbb{E} [\|\mathbf{M}_t\|_F^2] \quad (15)$$

$$= \lim_{t \rightarrow \infty} \frac{1}{2t} \log \mathbb{E} \left[\sum_{i=1}^n \sum_{j=1}^n m_{ij}^2(t) \right]. \quad (16)$$

We now express $\|\mathbf{M}_t\|_F^2$ as a function $f(\cdot)$ of another matrix, namely $\mathbf{M}_t \otimes \mathbf{M}_t$. Formally, f is thus defined as

$$f : \mathbb{R}^{n^2 \times n^2} \rightarrow \mathbb{R} : \mathbf{B} \mapsto \sum_{i=1}^n \sum_{j=1}^n b_{i+n(i-1), j+n(j-1)} \quad (17)$$

where $b_{k,l}$ is the k -th row, l -th column element of \mathbf{B} . In other words, $f(\mathbf{B})$ sums n^2 specially chosen elements of the $n^2 \times n^2$ matrix \mathbf{B} . Besides the equality

$$f(\mathbf{M} \otimes \mathbf{M}) = \|\mathbf{M}\|_F^2, \quad (18)$$

this function has the desirable property of being linear:

$$f(\alpha \mathbf{A} + \beta \mathbf{B}) = \alpha f(\mathbf{A}) + \beta f(\mathbf{B}). \quad (19)$$

Going back to equation (15), replace $\|\mathbf{M}\|_F^2$ according to (18). Then, using property (19), invoke linearity of expectation to interchange $f(\cdot)$ and $\mathbb{E}[\cdot]$:

$$\begin{aligned}\gamma &\leq \lim_{t \rightarrow \infty} \frac{1}{2t} \log \mathbb{E} [f(\mathbf{M}_t \otimes \mathbf{M}_t)] \\ &= \lim_{t \rightarrow \infty} \frac{1}{2t} \log f(\mathbb{E} [\mathbf{M}_t \otimes \mathbf{M}_t]) \\ &= \lim_{t \rightarrow \infty} \frac{1}{2t} \log f\left(\mathbb{E} \left[\prod_{p=1}^t \mathbf{A}(t-p) \otimes \prod_{p=1}^t \mathbf{A}(t-p) \right]\right) \\ &= \lim_{t \rightarrow \infty} \frac{1}{2t} \log f\left(\mathbb{E} \left[\prod_{p=1}^t (\mathbf{A}(t-p) \otimes \mathbf{A}(t-p)) \right]\right).\end{aligned}$$

Note that we used the distributive property of the Kronecker product, $(\mathbf{AC}) \otimes (\mathbf{BD}) = (\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D})$. Now use the i.i.d. property of the matrices $\mathbf{A}(t)$ to interchange the matrix product and expectation

$$\begin{aligned}\gamma &\leq \lim_{t \rightarrow \infty} \frac{1}{2t} \log f\left(\prod_{p=1}^t \mathbb{E}[\mathbf{A}(t-p) \otimes \mathbf{A}(t-p)]\right) \\ &= \lim_{t \rightarrow \infty} \frac{1}{2t} \log f(\mathbb{E}^t [\mathbf{A}(0) \otimes \mathbf{A}(0)]).\end{aligned}$$

To simplify the notation in the following, we define $\mathbf{B} := \mathbb{E}[\mathbf{A}(0) \otimes \mathbf{A}(0)]$. Using linearity of f once again, we have

$$\begin{aligned}\gamma &\leq \lim_{t \rightarrow \infty} \frac{1}{2t} \log f(\mathbf{B}^t) \\ &= \lim_{t \rightarrow \infty} \frac{1}{2t} \left(\log \|\mathbf{B}^t\| + \log f\left(\frac{\mathbf{B}^t}{\|\mathbf{B}^t\|}\right) \right) \\ &= \lim_{t \rightarrow \infty} \left(\frac{1}{2} \log \|\mathbf{B}^t\|^{\frac{1}{t}} + \frac{1}{2t} \log f\left(\frac{\mathbf{B}^t}{\|\mathbf{B}^t\|}\right) \right).\end{aligned}$$

It is a well known fact (Gelfand's formula, see e.g. [17]) that for any p -norm, $p \in \{1, 2, \dots, \infty\}$,

$$\lim_{t \rightarrow \infty} \|\mathbf{B}^t\|_p^{\frac{1}{t}} = \rho(\mathbf{B}) := \max\{|\lambda| : \lambda \in \lambda(\mathbf{B})\}.$$

To bound the last term, we choose $p = \infty$, which yields to $f\left(\frac{\mathbf{B}^t}{\|\mathbf{B}^t\|_\infty}\right) \leq n^2$. We can now conclude:

$$\begin{aligned}\gamma &\leq \lim_{t \rightarrow \infty} \left(\frac{1}{2} \log \|\mathbf{B}^t\|^{\frac{1}{t}} + \frac{1}{2t} \log n^2 \right) \\ &= \frac{1}{2} \log \lim_{t \rightarrow \infty} \|\mathbf{B}^t\|^{\frac{1}{t}} + 0 \\ &= \frac{1}{2} \log \rho(\mathbf{B}).\end{aligned}$$

Proof: (Theorem 5) Let $\mathbf{y}(t) = \mathbf{x}(t) - x_{ave}\mathbf{1}$. For any nonrandom choice of $\mathbf{y}(0) \in \mathbb{R}^n \setminus \{0\}$,

$$\begin{aligned}\mathbb{E} [\mathbf{y}^T(t)\mathbf{y}(t)] &= \mathbb{E} [\mathbb{E} [\mathbf{y}^T(t-1)\mathbf{A}^T(t-1)\mathbf{A}(t-1)\mathbf{y}(t-1)|\mathbf{y}(t-1)]] \\ &= \mathbb{E} [\mathbf{y}^T(t-1)\mathbb{E} [\mathbf{A}^T(t-1)\mathbf{A}(t-1)] \mathbf{y}(t-1)] \\ &\leq \lambda_1^t (\mathbb{E} [\mathbf{A}^T(0)\mathbf{A}(0)]) \mathbf{y}^T(0)\mathbf{y}(0).\end{aligned}$$

■

Then, by Jensen's inequality,

$$\begin{aligned} \mathbb{E} \frac{1}{2t} \left[\log \left(\frac{\mathbf{y}^T(t)\mathbf{y}(t)}{\mathbf{y}^T(0)\mathbf{y}(0)} \right) \right] &\leq \frac{1}{2t} \log \left(\frac{\mathbb{E} [\mathbf{y}^T(t)\mathbf{y}(t)]}{\mathbf{y}^T(0)\mathbf{y}(0)} \right) \\ &\leq \frac{1}{2} \log \lambda_1 (\mathbb{E} [\mathbf{A}^T(0)\mathbf{A}(0)]) . \end{aligned}$$

Now, we choose $\mathbf{y}(0) = \hat{\mathbf{y}}(0)$ such that $\mathbb{P}(\hat{\mathbf{y}}(0) \in \mathbb{R}^n \setminus V) = 1$, with V defined as in Theorem 3. For such a $\hat{\mathbf{y}}(0)$, $\lim_{t \rightarrow \infty} \frac{1}{t} \log \frac{\|\hat{\mathbf{y}}(t)\|_2}{\|\hat{\mathbf{y}}(0)\|_2} = \gamma$ with probability 1. Consequently, there is a bounded C such that $\frac{1}{t} \log \frac{\|\hat{\mathbf{y}}(t)\|_2}{\|\hat{\mathbf{y}}(0)\|_2} < C$ for all t . Then, by dominated convergence,

$$\begin{aligned} \gamma &= \mathbb{E} \lim_{t \rightarrow \infty} \frac{1}{t} \log \frac{\|\hat{\mathbf{y}}(t)\|_2}{\|\hat{\mathbf{y}}(0)\|_2} \\ &= \lim_{t \rightarrow \infty} \mathbb{E} \frac{1}{t} \log \frac{\|\hat{\mathbf{y}}(t)\|_2}{\|\hat{\mathbf{y}}(0)\|_2} \\ &\leq \frac{1}{2} \log \lambda_1 (\mathbb{E}[\mathbf{A}^T(0)\mathbf{A}(0)]) . \end{aligned}$$

■

REFERENCES

- [1] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing Markov chain on a graph," *SIAM Rev.*, vol. 46, no. 4, pp. 667–689 (electronic), 2004.
- [2] —, "Fastest mixing markov chain on a path," *The American Mathematical Monthly*, vol. 113, no. 1, pp. 70–74, 2006.
- [3] P. Brémaud, *Markov chains*, ser. Texts in Applied Mathematics. New York: Springer-Verlag, 1999, vol. 31, gibbs fields, Monte Carlo simulation, and queues.
- [4] J. Rosenthal, "Convergence rates of markov chains," *SIAM Rev.*, vol. 37, no. 3, pp. 387–405, 1995.
- [5] L. Xiao, S. Boyd, and S. Lall, "Distributed average consensus with time-varying metropolis weights," submitted to *Automatica*, June 2006.
- [6] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.
- [7] —, "Gossip algorithms: Design, analysis and applications," in *Proc. IEEE Infocom 2005*, vol. 3, Miami, March 2005, pp. 1653–1664.
- [8] A. Kashyap, T. Başar, and R. Srikant, "Quantized consensus," *Automatica*, vol. 43, no. 7, pp. 1192–1203, 2007.
- [9] V. Blondel, J. Hendrickx, A. Olshevsky, and J. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *CDC-ECC '05. 44th IEEE Conference on*, 2005, pp. 2996–3000.
- [10] T. Schmid, H. Dubois-Ferrière, and M. Vetterli, "SensorScope: Experiences with a Wireless Building Monitoring Sensor Network," in *Workshop on Real-World Wireless Sensor Networks (REALWSN'05)*.
- [11] H. Nguyen and P. Thiran, "Using End-to-End Data to Infer Lossy Links in Sensor Networks," in *IEEE Infocom 2006*, 2006. [Online]. Available: <http://www.ieee-infocom.org/>
- [12] P. Denantes, "Performance of averaging algorithms in time-varying networks," Master's thesis, TUM / EPFL, June 2007, available on <http://people.epfl.ch/patrick.denantes>.
- [13] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Analysis and optimization of randomized gossip algorithms," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, December 2004, pp. 5310–5315.
- [14] A. G. Dimakis, A. D. Sarwate, and M. J. Wainwright, "Geographic gossip: efficient aggregation for sensor networks," in *IPSN '06*. New York, NY, USA: ACM Press, 2006, pp. 69–76.
- [15] F. Bénézit, A. Dimakis, P. Thiran, and M. Vetterli, "Geographic gossip with path averaging is order optimal," submitted to *Allerton*, 2007.
- [16] P. Walters, *An introduction to ergodic theory*, ser. Graduate Texts in Mathematics. New York: Springer-Verlag, 1982, vol. 79.
- [17] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge: Cambridge University Press, 1990, corrected reprint of the 1985 original.
- [18] H. Furstenberg and H. Kesten, "Products of random matrices," *Ann. Math. Statist.*, vol. 31, pp. 457–469, 1960.
- [19] V. I. Oseledec, "A multiplicative ergodic theorem. Characteristic Ljapunov, exponents of dynamical systems," *Trudy Moskov. Mat. Obšč.*, vol. 19, pp. 179–210, 1968.
- [20] D. Ruelle, "Ergodic theory of differentiable dynamical systems," *Inst. Hautes Études Sci. Publ. Math.*, no. 50, pp. 27–58, 1979.
- [21] M. S. Raghunathan, "A proof of Oseledec's multiplicative ergodic theorem," *Israel J. Math.*, vol. 32, no. 4, pp. 356–362, 1979.
- [22] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Mixing times for random walks on geometric random graphs," in *ALENEX/ANALCO*. SIAM, 2005.