Computing Outside the Box: Average Consensus over Dynamic Networks

Bernadette Charron-Bost
Département d'informatique de l'ENS, ENS, CNRS, PSL University, Paris, France

Patrick Lambein-Monette[1]
Université Paris Cité, CNRS, IRIF, F-75013, Paris, France

Abstract

Networked systems of autonomous agents, and applications thereof, often rely on the control primitive of *average consensus*, where the agents are to compute the average of private initial values. To provide reliable services that are easy to deploy, average consensus should continue to operate when the network is subject to frequent and unpredictable change, and should mobilize few computational resources, so that deterministic, low powered, and anonymous agents can partake in the network.

In this stringent adversarial context, we investigate the implementation of average consensus by distributed algorithms over networks with bidirectional, but potentially short-lived, communication links. Inspired by convex recurrence rules for multi-agent systems, and the Metropolis average consensus rule in particular, we design a deterministic distributed algorithm that achieves asymptotic average consensus, which we show to operate in polynomial time in a synchronous temporal model.

The algorithm is easy to implement, has low space and computational complexity, and is fully distributed, requiring neither symmetry-breaking devices like unique identifiers, nor global control or knowledge of the network. In the fully decentralized model that we adopt, to our knowledge, no other distributed average consensus algorithm has a better temporal complexity.

Our approach distinguishes itself from classical convex recurrence rules in that the agent's values may sometimes leave their previous convex hull. As a consequence, our convergence bound requires a subtle analysis, despite the syntactic simplicity of our algorithm.

2012 ACM Subject Classification Theory of computation → Distributed algorithms; Computing methodologies → Distributed artificial intelligence; Networks → Sensor networks; Networks → Mobile networks; Networks → Network dynamics

Keywords and phrases average consensus, dynamic networks, distributed algorithms, iterated averaging, Metropolis

Digital Object Identifier 10.4230/LIPIcs.SAND.2022.10

Related Version *Extended Version*: https://arxiv.org/abs/2010.05675

Acknowledgements Patrick Lambein-Monette would like to thank his doctoral jury for stimulating discussions and remarks regarding previous versions of this material.

1 Introduction

1.1 Asymptotic average consensus

We consider a networked system of n *agents* – the generic term we use to denote the autonomous nodes of the network – denoted by the integer labels $1, \dots, n$. Agent i begins with an *input* value $\mu_i \in \mathbb{R}$, and maintains an *estimate* $x_i(t)$ of an objective. The input represents the agent's private observation of some aspect of its environment, which we assume to be taken arbitrarily from the domain of the problem; for example, the input may be a temperature reading, or the agent's initial position in space or velocity, if it is mobile. The

[1] Correponding author

estimate represents some aspect of the environment affected by the agent; depending on the system, it may simply be a local variable in the agent's memory, or it may directly represent some external parameter like the agent's heading or altitude.

Here, we focus on (asymptotic) *average consensus*, a control primitive widely studied by the distributed control community, where the estimates are made to *achieve asymptotic consensus* on the average of the input values – that is, to jointly converge towards the same limit $\overline{\mu} \coloneqq \frac{1}{n}\sum_i \mu_i$. The problem of computing an average is central to many applications in distributed control: let us cite sensor fusion and data aggregation [37, 27, 36], distributed optimization and machine learning [24, 28, 26], collective motion [32, 30], and more [13, 8, 12]. More generally, an average consensus primitive can be used to compute the relative frequency of the input values [16], and as such allows for the distributed computation of other statistical measures, for example the *mode* – the value with the highest support.

We study the problem of designing distributed algorithms for average consensus in the adversarial context of *dynamic networks*, where the communication links joining the agents change over time. Indeed, average consensus primitives are often needed in inherently dynamic settings, that static models fail to adequately describe. For a few examples, let us cite mobile ad-hoc networks, where links change as external factors cause the agents to move in space; autonomous vehicular networks, where agents are in control of their motion; or peer-to-peer networks, where constant arrivals and departures cause the network to reconfigure.

Specifically, we study distributed algorithms in a fully decentralized context: all agents start in the same state, run the same local algorithm, receive no global information about the system, only manipulate local variables, and interact with the system exclusively by exchanging messages with neighboring agents in the instantaneous communication graph. These constraints preclude the use of many standard solutions where the agents receive unique identifiers, where an agent is designated as a leader, or where tll agentshey initially agree on a bound on the network's degree or size. Moreover, we adopt a standard *local broadcast* communication model, particularly suited to modeling wireless networks, in which agents cast their messages without knowledge of their eventual recipients, and in particular cannot individually address their neighbors.

These conditions make it extremely hard to compute functions of the input values $\mu_1, \ldots, \mu_n$: on general *fixed* directed networks, deterministic distributed algorithms are only capable of computing functions that depend on the *set* of the input values $\{\mu_1, \ldots, \mu_n\}$, but not on their *multi-set* [17]. In particular, this precludes the distributed computation of the average. Here, we only consider networks with bidirectional communication links. Under this condition, the problem is rather simple if we assume a static communication graph [37, 5], in which case we can even deploy efficient solutions [31, 28] relying on spectral properties of the underlying graph. The problem is obviously much harder in a dynamic setting, which, for example, forbids the use of such sophisticated spectral techniques.

1.2 Contribution

A standard approach to asymptotic consensus has agents regularly adjust their estimates as a convex combination of those of their neighbors [10, 33], defined by a *convex recurrence rule*. We adopt a standard model of *synchronized rounds*, where this is expressed as a recurrence relation taking the generic form $x_i(t) = \sum_{j\in\mathcal{N}_i(t)} a_{ij}(t)x_j(t-1)$, where the weights $a_{ij}(t)$ are taken to form a convex combination, and the sum is over an agent's incoming neighbors in the communication graph at round t.

While asymptotic consensus is guaranteed as long as the never permanently splits [22], the estimates do not, in general, converge towards the average $\overline{\mu}$; reaching *average* consensus usually requires additionally enforcing *symmetric* weights $a_{ij}(t) = a_{ji}(t)$. Here, we study *distributed algorithms* for average consensus, i.e., we are interested in devising an algorithm that produces such weights through local computations only, in a fully decentralized manner.

For a simple example, average consensus comes easily by picking the weights $a_{ij}(t) = \frac{1}{n}$ when agents $i \neq j$ are neighbors in round t, and $a_{ii}(t) = 1 - \frac{\deg_i(t)-1}{n}$. However, this scheme might be simple to *describe*, but getting the agents to use these weights clearly requires getting them to know n, which is itself a serious distributed computing problem.

We will argue that the Metropolis rule [37], defined by the weights $a_{ij}(t) = \frac{1}{\max(\deg_i(t),\deg_j(t))}$ for any two $i \neq j$ neighbors in round t, breaks down over dynamic networks because of similar, albeit subtler, issues. We then propose a symmetric recurrence rule that *is* implementable over dynamic bidirectional networks, that we show to produce average consensus over any sufficiently connected network. The issues faced by the Metropolis rule are overcome by making the rule sometimes break convexity, which allows for keeping the average of the estimates constant even though the network changes unpredictably.

The temporal convexity of our distributed algorithm is polynomial, namely with a bound in $\mathrm{O}(n^4 \log n)$, whereas the *theoretical* complexity bound of the Metropolis rule is of $\mathrm{O}(n^2 \log n)$ [5]. To the best of our knowledge, this is the first deterministic algorithm that achieves asymptotic average consensus over bidirectional dynamic networks without any centralized input or symmetry-breaking assumptions. We note in passing that there exist *randomized* algorithms that are efficient in bandwidth and memory and converge in $\mathrm{O}(n)$ rounds to a good approximation of the average $\overline{\mu}$ with high probability [6, 20, 23].

We dub our distributed algorithm *MaxMetropolis*. Compared to the Metropolis rule, the change that we propose is deceptively simple: in the expression of the Metropolis weights, we replace the degree $\deg_i(t)$ with the value $\overline{\deg}_i(t-1) = \max\{\deg_i(1), \dots, \deg_i(t-1)\}$. However, the resulting rule is no longer *convex* – the estimates $x_i(t)$ may sometimes leave the convex hull of the set $\{x_1(t-1), \dots, x_n(t-1)\}$ – which makes the analysis substantially harder than in the purely convex case. Interestingly, although such "bad", convexity-breaking rounds, can happen at an arbitrarily late stage in the execution, we are able to bound the convergence time *independently* of when bad rounds occur – that is, once our target error threshold has been reached, disagreement in the system can still increase in later bad rounds, but not enough to break the threshold again.

1.3 Related works

Average consensus itself is at the center of a large body of works: among many others, let us cite [33, 34, 8, 19, 35, 37, 25, 3, 13, 28, 14], and see [26] for a recent overview of the domain. The approach based on doubly stochastic matrices in particular has been studied in depth, notably in [25, 29], with an analytical approach that focuses on aspects such as the temporal complexity and tolerance to quantization, whereas we address issues of a distributed nature, in particular the implementation of rules by distributed algorithms. We also note earlier work on random walks by Avin et al., who showed that dynamic networks can present considerable obstacles to mixing, in stark contrast with the well-behaved static case. Although their proposed solution is not directly implementable in our model, as it leverages global information (a bound over n), their study nonetheless deeply influenced the current work.

Of interest to our argument, we note that [35] looks for the *fixed* affine weights that optimize the speed of convergence towards average consensus over a given fixed graph, and find that the weights can often be negative. Our algorithm is itself able to solve average

consensus over dynamic networks precisely because it is sometimes allowed to use negative weights. When compared with our approach, the important difference is that we consider dynamic graphs and focus on distributed implementation of the recurrence rules, while the weights obtained in [35] are given by a centralized optimization problem, and are incompatible with a distributed approach.

A number of strategies aim at speeding up convex recurrence rules over static networks by having the agents learn what amounts to spectral elements of the graph Laplacian [4], and can result in linear-time convergence [31]. As is the case here, these represent distributed methods by which the agents learn structural properties of the communication graph. However, these methods rely on centralized symmetry-breaking crutches like unique identifiers, and their memory and computation footprint is much greater than ours, with agents computing and memorizing, in each round, the kernels of Hankel matrices of dimension $\Theta(n) \times \Theta(n)$. In contrast, our method can be used by anonymous agents, requires $\lceil \log n \rceil$ additional bits of memory and bandwidth, and has a trivial computational overhead.

2 Preliminaries

2.1 Mathematical toolbox

Let us fix some notation. If k is a positive integer, we denote by $[k]$ the set $\{1, \dots, k\}$. If any set $S \subset \mathbb{R}$ is non-empty and bounded, we denote its *diameter* by $\operatorname{diam} S \coloneqq \max S - \min S$.

A *directed graph* (or simply *graph*) $G = (V, E)$, with vertices in V and edges in $E \subseteq V \times V$, is called *reflexive* when $(i, i) \in E$ for all $i \in V$; G is *bidirectional* when $(i, j) \in E \iff (j, i) \in E$ for all $i, j \in V$; and G is *strongly connected* when directed paths join any pair of vertices – or simply *connected* when G is bidirectional.

All graphs that we consider here will be reflexive, bidirectional, and connected graphs of the form $G = ([n], E)$. In such a graph, the vertices linked to some vertex i form its *neighborhood* $\mathcal{N}_i(G) \coloneqq \{j \in [n] \,|\, (j, i) \in E\}$, and the count of its neighbors is its *degree* $\deg_i(G) \coloneqq |\mathcal{N}_i(G)|$. By definition, the degree is at most n, and in a reflexive graph it is at least 1.

We consistently denote matrices and vectors in bold italic style: upper case for matrices (e.g., $\boldsymbol{A}$) and lower case for vectors (e.g., $\boldsymbol{u}$), with their individual entries in regular italic style, (e.g., A_{ij}, u_k). The shorthand $\boldsymbol{v}^{\mathbb{N}}$ denotes the infinite vector sequence $\boldsymbol{v}(0), \boldsymbol{v}(1), \dots$.

The graph $G_{\boldsymbol{A}} = ([n], E)$ *associated* to a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is defined by $(j, i) \in E \iff A_{ij} \neq 0$ for all $i, j \in [n]$. The matrix $\boldsymbol{A}$ is said to be *irreducible* when $G_{\boldsymbol{A}}$ is strongly connected.

Given a vector $\boldsymbol{v} \in \mathbb{R}^n$, we write $\operatorname{diam} \boldsymbol{v}$ to mean the diameter of the *set* $\{v_1, \dots, v_n\}$ of its entries. The diameter constitutes a seminorm over $\mathbb{R}^n$; we call *consensus vectors* those of null diameter.

A matrix or a vector with non-negative (resp. positive) entries is itself called *non-negative* (resp. positive). A vector is called *stochastic* if its entries are non-negative sum to 1.

A *matrix* $\boldsymbol{A}$ is *stochastic* if its rows are all *stochastic* – that is, if $\boldsymbol{A}\mathbf{1} = \mathbf{1}$ – and any matrix that satisfies the condition $\boldsymbol{A}\mathbf{1} = \mathbf{1}$ will be said to be *affine*. We say that a matrix $\boldsymbol{A}$ is *doubly stochastic* when both $\boldsymbol{A}$ and $\boldsymbol{A}^{\intercal}$ are stochastic.

We denote the mean value of a vector $\boldsymbol{v} \in \mathbb{R}^n$ by $\langle \boldsymbol{v} \rangle \coloneqq \frac{1}{n} \sum_i v_i$. Doubly stochastic matrices play a central role in the study of average consensus, as multiplying any vector $\boldsymbol{v}$ by a doubly stochastic matrix $\boldsymbol{A}$ preserves its average – that is, $\langle \boldsymbol{A}\boldsymbol{v} \rangle = \langle \boldsymbol{v} \rangle$.

For any matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, we can arrange its n eigenvalues $\lambda_1, \dots, \lambda_n$, counted with their algebraic multiplicities, in decreasing order of magnitude: $|\lambda_1| \geqslant |\lambda_2| \geqslant \cdots \geqslant |\lambda_n|$.

Under this convention, the *spectral radius* of the matrix $\boldsymbol{A}$ is the quantity $\rho_{\boldsymbol{A}} \coloneqq |\lambda_1|$, and its *spectral gap* is the quantity $\gamma_{\boldsymbol{A}} \coloneqq |\lambda_1| - |\lambda_2|$. In particular, a stochastic matrix has a spectral radius of 1, which is itself an eigenvalue for the eigenvector $\mathbf{1}$.

2.2 Computing model

We consider a networked system of n agents, denoted $1, 2, \dots, n$. Computation proceeds in *synchronized rounds* that are communication closed, in the sense that no agent receives messages in round t that are sent in a different round. In each round $t \in \mathbb{N}_{>0}$, each agent i successively

1. broadcasts a *single* message $m_i(t)$ determined by its state at the beginning of round t
2. receives *some* messages among $m_1(t), \dots, m_n(t)$
3. undergoes an internal transition to a new state
4. produces a *round output* $x_i(t) \in \mathbb{R}$ and proceeds to round $t+1$.

The agents receiving agent i's message $m_i(t)$ are unknown to agent i at the time of emission, in step 1. Communications that occur in round t are modeled by a directed graph $\mathbb{G}(t) \coloneqq ([n], E(t))$, called the round t *communication graph*, which may change from one round to the next. We assume each communication graph $\mathbb{G}(t)$ to be reflexive, as an agent always has access to its own messages without delay or transmission loss.

Messages to be sent in step 1 and state transitions in step 3 are determined by a *sending* and a *transition* functions, which together define the *local algorithm* for agent i. Collected together, the local algorithms of all agents in the system constitute a *distributed algorithm*. We posit no *a priori* global coordination or knowledge of the agents: in particular, we assume no leader, no unique identifiers, and no initial agreement on global parameters such as n. An agent's computations only involve its own local variables in memory.

An *execution* of a distributed algorithm is a sequence of rounds, as defined above, with each agent running the corresponding local algorithm. We assume that all agents start simultaneously in round 1, since the algorithms under our consideration are robust to asynchronous starts, retaining the same time complexity as when the agents start simultaneously. Indeed, asynchronous starts only induce an initial transient period during which the network is disconnected, which cannot affect the convergence and complexity results of algorithms driven by convex recurrence rules.

In any execution of a distributed algorithm, the entire sequence $\boldsymbol{x}^{\mathbb{N}}$ is determined by the input vector $\boldsymbol{\mu}$ and the patterns of communications in each round t, i.e., the sequence of communication graphs $\mathbb{G} \coloneqq (\mathbb{G}(t))_{t \geqslant 1}$, called the *dynamic communication graph* of the execution, and so we write $\boldsymbol{x}^{\mathbb{N}} = \boldsymbol{x}^{\mathbb{N}}(\mathbb{G}, \boldsymbol{\mu})$. When the dynamic graph $\mathbb{G}$ is understood, we let $\mathcal{N}_i(t)$ and $\deg_i(t)$ respectively stand for $\mathcal{N}_i(\mathbb{G}(t))$ and $\deg_i(\mathbb{G}(t))$. As no confusion can arise, we will sometimes identify an agent with its corresponding vertex in the communication graph, and speak of the degree or neighborhood of an *agent* in a round of an execution.

We call a *network class* a set of dynamic graphs; given a class $\mathfrak{C}$, we denote by $\mathfrak{C}_{|n}$ the subclass $\{\mathbb{G} \in \mathfrak{C} \,|\, |\mathbb{G}| = n\}$. Here, our investigation will revolve around the class $\mathfrak{B}_{\mathrm{C}}$ of dynamic graphs of the following sort.

▶ **Assumption 1.** *In each round $t \in \mathbb{N}_{>0}$, the communication graph $\mathbb{G}(t)$ is reflexive, bidirectional, and connected.*

3 Recurrence rules for consensus

We distinguish local algorithms, as defined above, from the *recurrence rules* that they implement: the latter are recurrence relations that only describe how the estimates $x_i(t)$ change over time, while the former specifies the *distributed implementation* of such rules in the system, through local interactions. This discrepancy is apparent in the Metropolis rule, whose distributed implementation over dynamic networks is problematic due to its dependence on "knowledge at distance two".

3.1 Affine recurrence rules

Definition

Here, we focus on algorithmic solutions to the average consensus problem whose executions realize recurrence relations of the general form

$$x_i(t) = \sum_{j \in \mathcal{N}_i(t)} a_{ij}(t)\, x_j(t-1), \tag{1}$$

where the time-varying weights $a_{ij}(t)$ satisfy the affine constraint $\sum_{j\in\mathcal{N}_i(t)} a_{ij}(t) = 1$ and may depend on the dynamic graph $\mathbb{G}$ and the input values $\mu_1, \dots, \mu_n$. We refer to such relations as *affine recurrence rules*, and we say that a distributed algorithm *implements* the rule, insisting again that *a distributed algorithm is distinct from the rule it implements.*

Because of the constraint $\sum_{j\in\mathcal{N}_i(t)} a_{ij}(t) = 1$, the self-weights satisfy $a_{ii}(t) = 1 - \sum_{j\in\mathcal{N}_i(t)\setminus\{i\}} a_{ij}(t)$. An affine recurrence rule is thus fully specified by the weights $a_{ij}(t)$ assigned to an agent's proper neighbors $j \neq i$.

The affine rule of Equation (1) is equivalent to the vector equation $\boldsymbol{x}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t-1)$, where $A_{ij}(t) = a_{ij}(t)$ when i and j are neighbors in round t, and $A_{ij}(t) = 0$ otherwise. The affinity constraint then corresponds to the condition $\boldsymbol{A}(t)\mathbf{1} = \mathbf{1}$.

Convexity and convergence

We call the rule *convex* when all weights are non-negative – equivalently, when all matrices $\boldsymbol{A}(t)$ are stochastic. By and large, the study of affine recurrence rules focuses on that of convex recurrence rules, which guarantee convergence under mild conditions. We recall a standard convergence result, found under various forms in the literature, see for example [7, 33, 18, 22].

▶ **Proposition 2.** *Assume that the weights of Equation* (1) *admit a uniform positive lower bound* α*:* $a_{ij}(t) \geqslant \alpha > 0$ *for all* t*,* i*, and* $j \in \mathcal{N}_i(t)$*. Under Assumption 1, the vectors* $\boldsymbol{x}(t)$ *converge to a consensus vector.*

We speak of *uniform convexity* when such a parameter α exists, and we note that in this case asymptotic consensus is actually ensured by conditions much weaker than Assumption 1: for bidirectional interactions, it is enough that the network never become permanently split [22, Theorem 1].

Remark that Proposition 2 says nothing of the *value* of the consensus; affine recurrence rules for *average* consensus are typically designed to produce matrices that are doubly stochastic. By enforcing the invariant $\langle \boldsymbol{x}(t) \rangle = \langle \boldsymbol{x}(t-1) \rangle$, this makes the initial average $\overline{\mu}$ the only admissible consensus value.

The *convergence time* of a single sequence $\boldsymbol{z}^{\mathbb{N}}$, given by $\mathrm{T}(\varepsilon; \boldsymbol{z}^{\mathbb{N}}) \coloneqq \inf\{t \in \mathbb{N} \,|\, \forall \tau \geqslant t\colon \operatorname{diam} \boldsymbol{z}(\tau) \leqslant \varepsilon\}$, measure its progress towards asymptotic consensus. For a rule or an algorithm, we consider the more helpful *worst-case relative convergence time* over a class $\mathfrak{C}$: for a system of n agents, it is defined by

$$\mathrm{T}(\varepsilon; n, \mathfrak{C}) \coloneqq \sup_{\boldsymbol{\mu} \in \mathbb{R}^n} \sup_{\mathbb{G} \in \mathfrak{C}_{|n}} \mathrm{T}\big(\varepsilon \cdot \operatorname{diam} \boldsymbol{\mu}; \boldsymbol{x}^{\mathbb{N}}(\mathbb{G}, \boldsymbol{\mu})\big), \tag{2}$$

where we drop the class $\mathfrak{C}$ if it is clear from the context.

We recall the following bounds for uniformly convex recurrence rules over the class $\mathfrak{B}_{\mathrm{C}}$: when all matrices are *doubly stochastic*, the convergence time is in $\mathrm{O}(\alpha^{-1} n^2 \log {}^{n}\!/_{\varepsilon})$ [25, Theorem 10]. In the common case that $\alpha = \Theta({}^{1}\!/_{n})$, all rules are known to admit executions that do not converge before $\Omega(n^2 \log {}^{1}\!/_{\varepsilon})$ rounds over the fixed line graph with n vertices [29, Theorem 6.1].

3.2 Consensus and average consensus rules

The EqualNeighbor rule

The prototypical example of a convex recurrence rule is the *EqualNeighbor* rule, where an agent assigns the equal weights to all its neighbors, itself included:

$$x_i(t) = \frac{1}{\deg_i(t)} \sum_{j \in \mathcal{N}_i(t)} x_j(t). \tag{3}$$

We can mechanically derive an algorithm implementing the EqualNeighbor rule: in each round t, broadcast one's latest estimate $x_i(t-1)$, and pick as new estimate $x_i(t)$ the arithmetic mean of the incoming values. Since $\deg_i(t) \leqslant n$, this rule admits ${}^{1}\!/_{n}$ as a parameter of uniform convexity, and for a dynamic graph of $\mathfrak{B}_{\mathrm{C}}$, Proposition 2 shows that any solution to Equation (3) converges to a consensus vector.

Clearly, the EqualNeighbor rule does *not* solve the *average* consensus problem on the entire class $\mathfrak{B}_{\mathrm{C}}$, as the weights are generally not symmetric, unless each communication graph $\mathbb{G}(t)$ is *regular* – that is, if all its vertices have the same degree.

The Metropolis rule

In [37], Xiao et al. investigate the problem of *distributed sensor fusion* with the help of an average consensus primitive. For that, they describe the "maximum-degree" rule, parametrized with an integer $N \geqslant 1$, defined by the constant weights $a_{ij}(t) = {}^{1}\!/_{N}$ for any agents $i \neq j$ neighbors in round t.

The authors note that this rule solves average consensus over the class $\cup_{n \leqslant N} \mathfrak{B}_{\mathrm{C}|n}$, but remark that *implementing* this rule hinges on the agents initially agreeing on the bound N, embedding an assumption of centralized control. This makes the "maximum-degree" rule inapplicable over truly decentralized systems – indeed, our communication model does not generally allow for the distributed computation of such a bound N [1]. Xiao et al. go on suggesting the alternative rule:

$$x_i(t) = x_i(t-1) + \sum_{j \in \mathcal{N}_i(t)} \frac{x_j(t-1) - x_i(t-1)}{\max(\deg_i(t), \deg_j(t))}, \tag{4}$$

generally referred to as the *Metropolis* rule, as it is inspired from the Metropolis-Hastings method [15, 21].

Analytically, this rule is appealing, as it was recently shown [5] to display a worst-case convergence time of $\mathrm{O}(n^2 \log n)$ over the entire class $\mathfrak{B}_{\mathrm{C}}$ – making it the fastest rule known to us to solve either consensus or average consensus on that class. From a computational perspective, it is argued in [37] that the Metropolis rule is better suited for decentralized systems, as it only leverages "local" knowledge. Indeed, agents can implement this rule knowing only, in each round, their own degrees in the current communication graph and that of their neighbors – compared to the initial agreement over $N \geqslant n$ required of the "maximum-degree" rule.

Unfortunately, local algorithms cannot implement the Metropolis rule over dynamic networks. The rule is only "local" in the weak sense that an agent's next estimate $x_i(t)$ depends on information present *within distance* 2 of agent i in the communication graph $\mathbb{G}(t)$, which is not local *enough* when the network is subject to change.

Indeed, since agent $j \in \mathcal{N}_i(t)$ only learns its round t degree $\deg_j(t)$ at the *end* of round t – by counting its incoming messages – it cannot share this information with other agents before the following round. Any distributed implementation of the Metropolis rule would therefore require communication links that evolve at a slow and regular pace; one can imagine a network whose topology can only change once every k rounds, when $t \equiv 0 \bmod k$, e.g., at even rounds.

When the network is subject to *unpredictable* changes, the situation is even worse: we need to warn all agents, ahead of time, about any upcoming topology change. In effect, this amounts to having a global synchronization signal precede every change in the communication topology. For a topology change in round t_0, this differs little from starting an *entirely new execution* with new input values $\mu'_1 = x_1(t_0 - 1), \ldots, \mu'_n = x_n(t_0 - 1)$. To paraphrase, given a sufficiently stable communication network, one "can" implement the Metropolis rule over dynamic networks; however, the execution is fully decentralized only as long as no topology change actually occurs.

We note that, although we have covered the Metropolis rule here, other average consensus rules typically face similar problems, even when expressingly designed for dynamic networks. As an example, while the Metropolis rule can be implemented with a two-message protocol – e.g., on a communication graph that changes every other round, and with all agents agreeing on the parity of the round number, see e.g., [9] for a discussion – the rules given in [29, Algorithm 8.2] and [25, Section IV.A] involve a three-message protocol. Their implementation thus requires more network stability, and a stronger agreement, than Metropolis.

4 The MaxMetropolis algorithm

4.1 A symmetric affine rule

Symmetrizing

Let us briefly recall the idea of the Metropolis-Hastings [15, 21] method: given a positive stochastic vector $\boldsymbol{\pi}$, the method turns a stochastic matrix $\boldsymbol{A}$ – usually viewed as the transition matrix of a reversible Markov chain – into another stochastic matrix $\boldsymbol{A'}$ with stationary distribution $\boldsymbol{\pi}$, by picking off-diagonal entries as $A'_{ij} = \min\left(A_{ij}, \frac{\pi_j}{\pi_i} A_{ji}\right)$. When $\boldsymbol{\pi}$ is the constant vector $\left(\frac{1}{n}, \frac{1}{n}, \cdots, \frac{1}{n}\right)$, we get the simpler transform $M(-)$, defined entry-wise by:

$$\forall i, j \in [n] \colon [M(\boldsymbol{A})]_{ij} = \begin{cases} \min(A_{ij}, A_{ji}) & j \neq i \\ 1 - \sum_{k \neq i} \min(A_{ik}, A_{ki}) & j = i. \end{cases} \tag{5}$$

Let us call this transform the *Metropolis-Hastings symmetrization*; as an example, the symmetrization of the EqualNeighbor matrix yields the Metropolis matrix. We can make a few remarks: for any matrix $\boldsymbol{A}$, the matrix $M(\boldsymbol{A})$ is affine and symmetric by construction, and for any $j \neq i$ we have $[M(\boldsymbol{A})]_{ij} \leqslant A_{ij}$ and therefore $[M(\boldsymbol{A})]_{ii} \geqslant A_{ii}$. In particular, if the matrix $\boldsymbol{A}$ is stochastic with positive diagonal entries, then so is $M(\boldsymbol{A})$; if we can use Proposition 2 to establish the convergence of the system $\boldsymbol{x}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t-1)$, then necessarily the system $\boldsymbol{y}(t) = M(\boldsymbol{A}(t))\boldsymbol{y}(t-1)$ also converges, and achieves average consensus.

Bound learning

To apply the Metropolis-Hastings symmetrization while avoiding the aforementioned limitations of the Metropolis rule, let us temporarily assume that each agent $i \in [n]$ initially knows an upper bound $q_i \geqslant 1$ over its degree throughout the execution, i.e., $q_i \geqslant \deg_i(t)$ for all $t \geqslant 1$.

In this case, an agent may broadcast in each round the pair $\langle q_i, x_i(t-1)\rangle$ to its neighbors, and adjust its estimate as

$$x_i(t) = x_i(t-1) + \sum_{j \in \mathcal{N}_i(t)} \frac{x_j(t-1) - x_i(t-1)}{\max(q_j, q_i)}; \tag{6}$$

we easily see that this rule produces symmetric weights ($a_{ij}(t) = a_{ji}(t)$) and has a uniform convexity parameter of $1/\max_i q_i$. For a dynamic graph of $\mathfrak{B}_{\mathrm{C}}$, any solution $\boldsymbol{z}^{\mathbb{N}}$ of Equation (6) converges to a consensus vector, by Proposition 2, and therefore achieves asymptotic average consensus, since the weights are symmetric. Using e.g., the aforementioned result of [25, Theorem 10], we can show that the convergence time behaves as $\mathrm{O}(\max_i q_i \cdot n^2 \log {}^{n}\!/_{\varepsilon})$, which is polynomial in n when the bounds q_i themselves are.

Obviously, assuming such bounds q_i supposes that the agents dispose of information about the dynamic structure of the network ahead of the execution, which our model explicitly disallows. Instead of *assuming* such bounds, we next show that we can solve the average consensus problem for the class $\mathfrak{B}_{\mathrm{C}}$ by making agents learn good bounds over time in a manner consistent with our symmetric and local model.

To this effect, for each agent i we let $\overline{\deg}_i(t) \coloneqq \max\{\deg_i(1), \dots, \deg_i(t)\}$ for any round t. For a dynamic graph in $\mathfrak{B}_{\mathrm{C}|n}$, the value $\overline{\deg}_i(t) \in [2, n]$ is weakly increasing with t, and therefore stabilizing: we have $\overline{\deg}_i(t) = \overline{\deg}_i \coloneqq \max_{\tau \geqslant 1} \deg_i \tau$ for all rounds t beyond some round t_i^*, Thus, by keeping track of $\overline{\deg}_i(t)$, agent i will eventually hold a bound on its *future* degrees for the rest of the execution, which may be used to implement Equation (6), not for the whole interval $[1, \infty[$, but on all but finitely many rounds.

Moreover, we have by definition $\overline{\deg}_i(t) \geqslant \deg_i(t)$, so that using $\overline{\deg}_i(t)$ in place of q_i in Equation (6) produces a convex rule – even though $\overline{\deg}_i(t)$ may be inferior to agent i's *future* degree. Unfortunately, the weights $\frac{1}{\max\left(\overline{\deg}_i(t), \overline{\deg}_j(t)\right)}$ cannot be computed in a local manner: since $\overline{\deg}_i(t)$ depends on $\deg_i(t)$, the issues of the Metropolis rule apply here as well, as an agent cannot communicate its degree to its neighbors at the time they need the information.

We overcome this obstacle with a small, but crucial adjustment: building the round t weights using the *latest known bound* $\overline{\deg}_i(t-1)$ in place of $\overline{\deg}_i(t)$ allows us to conform to the stringent *locality* constraints by sacrificing the *convexity* of the rule. Specifically, we propose the *MaxMetropolis* algorithm – given in Algorithm 1, – a deterministic distributed algorithm which solves the average consensus problem over the class $\mathfrak{B}_{\mathrm{C}}$ in polynomial time, by implementing the rule

$$x_i(t) = x_i(t-1) + \sum_{j \in \mathcal{N}_i(t)} \frac{x_j(t-1) - x_i(t-1)}{\max\left(\overline{\deg}_j(t-1), \overline{\deg}_i(t-1)\right)}. \tag{7}$$

Algorithm 1 The MaxMetropolis algorithm, code for agent i.

Input: $\mu_i \in \mathbb{R}$

Initially:

$x_i \leftarrow \mu_i$;

$q_i \leftarrow 2$;

In each round do:

`send` $m_i = \langle x_i, q_i \rangle$;

`receive` $m_{j_1}, \dots, m_{j_d}$; ▷ d `neighbors`

$x_i \leftarrow x_i + \sum_{k=1}^{d} \frac{x_{j_k} - x_i}{\max(q_i, q_{j_k})}$;

$q_i \leftarrow \max(q_i, d)$;

`output` x_i ;

The weights are clearly symmetric, and so any solution to Equation (7) satisfies the invariant $\langle \boldsymbol{x}(t+1) \rangle = \langle \boldsymbol{x}(t) \rangle$. Moreover, by construction, there exists a round t^* after which we have $\overline{\deg}_i(t-1) = \overline{\deg}_i \geqslant \deg_i(t)$; the assumptions of Proposition 2 are then satisfied over the infinite interval $[t^*, \infty[$. Taken together, these observations immediately give us that MaxMetropolis is an average consensus distributed algorithm for the class $\mathfrak{B}_{\mathrm{C}}$.

On the other hand, in contrast with the Metropolis rule, the MaxMetropolis rule offers no guarantee of convexity: we easily see that if, for example, $\deg_i(t)$ is much larger than $\overline{\deg}_i(t-1)$, $x_i(t)$ may leave the convex hull of $\{x_j(t-1) \,|\, j \in \mathcal{N}_i(t)\}$, and in fact may even leave the convex hull of $\{x_j(t-1) \,|\, j \in [n]\}$. Such convexity-breaking rounds can occur late in the execution, and our main analytical difficulty will be to show that these "late bad rounds" cannot introduce too much noise in the system once a given degree of agreement has been reached.

▶ **Theorem 3.** *The MaxMetropolis algorithm solves the average consensus problem in all of its executions over the class $\mathfrak{B}_{\mathrm{C}}$. For a system of n agents and an error threshold of $\varepsilon > 0$, the convergence time is bounded by $\mathrm{T}(\varepsilon; n) = \mathrm{O}(n^4 \log n/\varepsilon)$.*

4.2 Temporal complexity of the MaxMetropolis algorithm

To prove Theorem 3, we need to introduce a few technical results borrowed from [5], where they are given a more general and detailed exposition. In the following, we denote by $\sigma(-)$ the sample standard deviation: $\sigma(\boldsymbol{x}) \coloneqq \sqrt{\sum_i (x_i - \langle \boldsymbol{x} \rangle)^2}$. The crux of the proof is to dominate $\sigma(\boldsymbol{x}(t))$ with a geometrically decreasing sequence, taking care when handling matrices with possibly negative entries.

▶ **Lemma 4.** *For any vector $\boldsymbol{v} \in \mathbb{R}^n$, we have*

$$\sqrt{2/n}\,\sigma(\boldsymbol{v}) \leqslant \operatorname{diam} \boldsymbol{v} \leqslant 2\,\sigma(\boldsymbol{v}). \tag{8}$$

The inequalities are strict if, and only if, the vectors $\boldsymbol{v}$ and $\mathbf{1}$ are independent.

Proof. Developing the definition of the standard deviation, we have $\sigma(\boldsymbol{v}) = \sqrt{\frac{1}{2}\sum_{i \neq j} (v_i - v_j)^2}$, which yields the left-hand side inequality. Moreover, without loss of generality we can assume $\langle \boldsymbol{v} \rangle = 0$, in which case $\sigma(\boldsymbol{v}) = \|\boldsymbol{v}\|$; the right-hand side inequality then follows from the classic bounds $\operatorname{diam} - \leqslant 2\,\|-\|_\infty$ and $\|-\|_\infty \leqslant \|-\|$. ◀

The following lemma is a restatement of a standard variational characterization of the eigenvalues of the matrix $\mathbf{I} - \boldsymbol{A}^\intercal \boldsymbol{A}$; see e.g., [11] for an in-depth treatment of the question.

▶ **Lemma 5.** *Let $\boldsymbol{A}$ denote a doubly stochastic matrix, irreducible and with positive diagonal entries. For any vector $\boldsymbol{v}$, we have*

$$\sigma(\boldsymbol{A}\boldsymbol{v}) \leqslant \sqrt{1 - \gamma_{\boldsymbol{A}^\intercal \boldsymbol{A}}}\, \sigma(\boldsymbol{v}); \tag{9}$$

in the particular case where $\boldsymbol{A}$ is symmetric, we have $\sigma(\boldsymbol{A}\boldsymbol{v}) \leqslant (1 - \gamma_{\boldsymbol{A}})\, \sigma(\boldsymbol{v})$.

Finally, we will rely on the following spectral bound, given in [25, Lemma 9].

▶ **Lemma 6.** *Let $\boldsymbol{A}$ be a stochastic matrix, with smallest positive entry α. If $\boldsymbol{A}$ is symmetric, irreducible, and has positive diagonal entries, then we have*

$$\gamma_{\boldsymbol{A}} \geqslant \frac{\alpha}{n(n-1)}. \tag{10}$$

With Lemmas 4–6, we can turn to the proof of Theorem 3.

Proof of Theorem 3. Let us fix a dynamic graph $\mathbb{G} \in \mathfrak{B}_{\mathrm{C}}$ with $n \geqslant 2$ vertices, and define

$$\begin{gathered} \overline{\deg}_i(t) \coloneqq \max_{\tau \leqslant t} \deg_i \tau, \quad \overline{\deg}_i \coloneqq \sup_{t \geqslant 1} \deg_i(t), \quad \overline{\deg}_{\mathbb{G}} \coloneqq \max_{i \in [n]} \overline{\deg}_i, \quad \text{and} \\ \mathcal{K} \coloneqq \{t \geqslant 1 \,|\, \exists i \colon \overline{\deg}_i(t-1) < \overline{\deg}_i(t)\}, \end{gathered} \tag{11}$$

where by convention we set $\deg_i(0) = 2$ so that the set $\mathcal{K}$ is properly defined. By definition, each sequence $\overline{\deg}_i(t)$ is weakly increasing with t, and has $\overline{\deg}_i$ for limit. Since $\deg_i(t) \leqslant n$, there are at most $\overline{\deg}_i$ rounds with $\overline{\deg}_i(t-1) < \overline{\deg}_i(t)$. The set $\mathcal{K}$ is therefore finite, with cardinal $\delta \coloneqq |\mathcal{K}| \leqslant \sum_i \overline{\deg}_i$. We let $t^* \coloneqq \max \mathcal{K} + 1$; by construction, in all rounds $t \geqslant t^*$ we have $\overline{\deg}_i(t) = \overline{\deg}_i$.

By an immediate induction, we see that, in any execution of the MaxMetropolis algorithm over the dynamic communication graph $\mathbb{G}$, the sequence of estimate vectors satisfies the recurrence relation $\boldsymbol{x}(t) = \boldsymbol{A}(t)\, \boldsymbol{x}(t-1)$, where the affine MaxMetropolis matrix $\boldsymbol{A}(t)$ is given for off-diagonal entries $i \neq j$ by

$$A_{ij}(t) = \begin{cases} \dfrac{1}{\max\big(\overline{\deg}_i(t-1), \overline{\deg}_j(t-1)\big)} & j \in \mathcal{N}_i(t) \\ 0 & j \notin \mathcal{N}_i(t), \end{cases} \tag{12}$$

and $\boldsymbol{x}(0) = (\mu_1, \dots, \mu_n)$ is given by the input values of the execution.

Equation (12), shows that the affine matrix $\boldsymbol{A}(t)$ is symmetric, and thus for any vector $\boldsymbol{v}$ we have $\langle \boldsymbol{A}(t)\boldsymbol{v} \rangle = \langle \boldsymbol{v} \rangle$. This is true for all $t \geqslant 1$, and so $\langle \boldsymbol{x}(t) \rangle = \overline{\mu}$ is an invariant of the execution. If we show asymptotic consensus, then the consensus value is necessarily the initial average $\overline{\mu}$.

As a result of the Metropolis-Hastings symmetrization, the diagonal entries of the matrix $\boldsymbol{A}(t)$ satisfy

$$A_{ii}(t) \geqslant 1 - \frac{\deg_i(t) - 1}{\overline{\deg}_i(t-1)}, \tag{13}$$

which gives in particular $A_{ii}(t) \geqslant 1/n$ when $t \notin \mathcal{K}$. The vector sequence $(\boldsymbol{x}(t))_{t \geqslant t^*}$ thus satisfies the assumptions of Proposition 2 for the uniform convexity parameter $\alpha = 1/n$, and so $\boldsymbol{x}(t)$ converges to a consensus vector. As already discussed, the limit value is necessarily

the initial average $\overline{\mu}$, and the system achieves asymptotic average consensus. This holds for any dynamic graph $\mathbb{G} \in \mathfrak{B}_{\mathrm{C}}$ and arbitrary input values $\mu_1, \dots, \mu_n$, and thus MaxMetropolis is an average consensus algorithm for the class $\mathfrak{B}_{\mathrm{C}}$.

It remains to show the polynomial convergence bound $\mathrm{T}(\varepsilon; n) = \mathrm{O}(n^4 \log n/\varepsilon)$. We start with the remark that the diagonal entry $A_{ii}(t)$ can be negative in a round t during which $\overline{\deg}_i(t) > \overline{\deg}_i(t-1)$. Because of this, the estimate $x_i(t)$ might end up outside the range of the previous estimates $\{x_1(t), \dots, x_n(t)\}$. As a consequence, rounds $t \in \mathcal{K}$ are "bad" rounds, where the system may move away from consensus, delaying the eventual convergence. In the class $\mathfrak{B}_{\mathrm{C}}$, there is no uniform upper bound on the value of t^*, and such convexity-breaking rounds may occur arbitrarily late in the execution. Our challenge is therefore to show that, in finite time, the system reaches a given degree of agreement which cannot be undone in later "bad" rounds. We do this by accounting, from the start, the total delay that can be accrued in rounds $t \in \mathcal{K}$.

We follow the variations of the sample standard deviation $S(t) \coloneqq \sigma(\boldsymbol{x}(t))$ from one round to the next, distinguishing on whether $t \in \mathcal{K}$ or not.

Case $t \notin \mathcal{K}$. By Equation (13), the irreducible matrix $\boldsymbol{A}(t)$ has positive diagonal entries, and thus has a positive spectral gap. By Lemma 5, we have

$$\forall t \notin \mathcal{K}\colon S(t) \leqslant (1 - \gamma_{\boldsymbol{A}(t)}) \cdot S(t-1). \tag{14}$$

Case $t \in \mathcal{K}$. Here, the matrix $\boldsymbol{A}(t)$ may have negative diagonal entries. It need not be a stochastic matrix, and indeed its spectral radius $\rho_{\boldsymbol{A}(t)}$ is possibly greater than 1. However, as a symmetric matrix, the matrix $\boldsymbol{A}(t)$ is diagonalizable, and thus we have $\|\boldsymbol{A}(t)\boldsymbol{v}\| \leqslant \rho_{\boldsymbol{A}(t)} \cdot \|\boldsymbol{v}\|$ for any vector $\boldsymbol{v}$. For the particular case $\boldsymbol{v} = \boldsymbol{x}(t-1) - \overline{\mu}\mathbf{1}$, this results in

$$\forall t \in \mathcal{K}\colon S(t) \leqslant \rho_{\boldsymbol{A}(t)} \cdot S(t-1). \tag{15}$$

Equation (15) actually holds for all $t \geqslant 1$, but it is strictly worse than Equation (14) for rounds $t \notin \mathcal{K}$.

Thus we let

$$\kappa(t) \coloneqq \begin{cases} \rho_{\boldsymbol{A}(t)} & t \in \mathcal{K}, \\ 1 - \gamma_{\boldsymbol{A}(t)} & t \notin \mathcal{K}, \end{cases} \tag{16}$$

and we can summarize Equations (14) and (15) by $\forall t \geqslant 1\colon S(t) \leqslant \kappa(t) \cdot S(t-1)$. By induction, we then have $S(t) \leqslant \prod_{\tau \leqslant t} \kappa(\tau) \cdot S(0)$, and, applying Lemma 4 twice, we get

$$\forall t \geqslant 1\colon \operatorname{diam} \boldsymbol{x}(t) \leqslant 2\sqrt{n} \prod_{\tau \leqslant t} \kappa(\tau) \cdot \operatorname{diam} \boldsymbol{\mu}. \tag{17}$$

We are interested in the asymptotic behavior of $2\sqrt{n} \prod_{\tau \leqslant t} \kappa(\tau)$.

In order to bound the spectral radius $\rho_{\boldsymbol{A}(t)}$, we let $\nu_{t,i} \coloneqq 1 - \min(0, A_{ii}(t))$, and $\nu_t \coloneqq \max_i \nu_{t,i}$. Let us show that $\rho_{\boldsymbol{A}(t)} \leqslant \nu_t^2$: pick any eigenvalue $\lambda \in \operatorname{Sp} \boldsymbol{A}(t)$. By construction, the quantity $\left(1 + \frac{\lambda - 1}{\nu_t}\right)$ is an eigenvalue of the stochastic matrix $\frac{1}{\nu_t}(\boldsymbol{A}(t) + (\nu_t - 1)\mathbf{I})$, and so is less than 1 in absolute value. We have $1 - 2\,\nu_t \leqslant \lambda \leqslant 1$, and so $|\lambda| \leqslant 2\nu_t - 1$. Using the basic inequality $x^2 - 2\,x + 1 \geqslant 0$, we have $|\lambda| \leqslant \nu_t^2$, and therefore $\rho_{\boldsymbol{A}(t)} \leqslant \nu_t^2$ since this holds for any $\lambda \in \operatorname{Sp} \boldsymbol{A}(t)$.

For any $t \in \mathcal{K}$ and $i \in [n]$, we have $\sum_{j \neq i} A_{ij}(t) \leqslant \frac{\overline{\deg}_i(t)-1}{\overline{\deg}_i(t-1)} \leqslant \frac{\overline{\deg}_i(t)}{\overline{\deg}_i(t-1)}$. Since $\overline{\deg}_i(t)$ is weakly increasing with t, we have in turn $\nu_{t,i} \leqslant \frac{\overline{\deg}_i(t)}{\overline{\deg}_i(t-1)}$, from here we have

$$\begin{aligned}
\prod_{t\in\mathcal{K}} \rho_{\boldsymbol{A}(t)} &\leqslant \Big(\prod_{t\in\mathcal{K}} \max_{i\in[n]} \nu_{t,i}\Big)^2 && \Big\downarrow \nu_{t,i} \geqslant 1 \\
&\leqslant \Big(\prod_{t\in\mathcal{K}} \prod_{i\in[n]} \nu_{t,i}\Big)^2 && \Big\downarrow \nu_{t,i} \leqslant \tfrac{\overline{\deg}_i(t)}{\overline{\deg}_i(t-1)} \\
&\leqslant \Big(\prod_{i\in[n]} \prod_{t\in\mathcal{K}} \tfrac{\overline{\deg}_i(t)}{\overline{\deg}_i(t-1)}\Big)^2 && \Big\downarrow \overline{\deg}_i(t) = \overline{\deg}_i(t-1) \text{ when } t \notin \mathcal{K} \\
&\leqslant \Big(\prod_{i\in[n]} \prod_{t\geqslant 1} \tfrac{\overline{\deg}_i(t)}{\overline{\deg}_i(t-1)}\Big)^2 \\
&= \Big(\prod_{i\in[n]} \tfrac{\overline{\deg}_i}{2}\Big)^2 = 2^{-2n}\varpi^2,
\end{aligned}$$

where $\varpi \coloneqq \prod_{i\in[n]} \overline{\deg}_i$.

From here, we let $\gamma \coloneqq \inf_{t\notin\mathcal{K}} \gamma_{\boldsymbol{A}(t)}$, and we have

$$\begin{aligned}
\prod_{\tau\leqslant t} \kappa(\tau) &= \Big(\prod_{\tau\in[1,t]\cap\mathcal{K}} \kappa(\tau)\Big)\Big(\prod_{\tau\in[1,t]\setminus\mathcal{K}} \kappa(\tau)\Big) && \Big\downarrow \kappa(\tau \in \mathcal{K}) \geqslant 1 \\
&\leqslant \Big(\prod_{\tau\in\mathcal{K}} \rho_{\boldsymbol{A}(t)}\Big)\Big(\prod_{\tau\in[1,t]\setminus\mathcal{K}} (1-\gamma_{\boldsymbol{A}(t)})\Big) \\
&\leqslant 2^{-2n}\varpi^2(1-\gamma)^{t-\delta}.
\end{aligned}$$

As a consequence, given any error threshold $\varepsilon > 0$, the estimates are contained in a ball of diameter $(\varepsilon \cdot \operatorname{diam} \boldsymbol{\mu})$ at the latest in round $t_\varepsilon \leqslant \delta + \gamma^{-1} \log(2^{-2n+1}\varpi^2\sqrt{n}/\varepsilon)$. From Lemma 6, we have $\gamma^{-1} \leqslant n(n-1)\overline{\deg}_{\mathbb{G}}$, and using $\varpi \coloneqq \prod_{i\in[n]} \overline{\deg}_i$ and $\delta \coloneqq |\mathcal{K}| \leqslant \sum_{i\in[n]} \overline{\deg}_i - 2n$, we get:

$$t_\varepsilon \leqslant \sum_i \overline{\deg}_i - 2n + n(n-1)\overline{\deg}_{\mathbb{G}}\left(2\sum_i \log \overline{\deg}_i - (2\,n-1)\log 2 - \log \varepsilon\right), \tag{18}$$

which, using the fact that $\overline{\deg}_i \leqslant n$, finally gives us $t_\varepsilon = \mathrm{O}(n^4 \log n/\varepsilon)$. ◀

Compared to the $\mathrm{O}(n^2 \log n/\varepsilon)$ convergence time of the Metropolis rule, the latter asymptotic bound is worse by a factor $n\,\overline{\deg}_{\mathbb{G}}$. From the proof, we can give a rough analysis of this factors: the factor n represents the delay due to broken convexity, as each agent individually induces a delay of $\log \overline{\deg}_i$. The factor $\overline{\deg}_{\mathbb{G}}$ comes from the fact that, whereas the Metropolis rule always selects the *best* possible off-diagonal weights – that is, the largest ones, – the MaxMetropolis rule makes conservative choices so as to allow for a decentralized algorithmic implementation that only breaks convexity finitely many times.

Improvements to the MaxMetropolis approach, based for example on adjusting the parameters q_i downwards in pursuit of faster mixing, must therefore be considered with extreme care, as gains due to larger weights might result in greater delays due to broken convergence.

5 Conclusion

In this paper, we have presented the MaxMetropolis algorithm, a parsimonious distributed algorithm for average consensus that operates in polynomial time over connected bidirectional dynamic networks, without resorting to any centralized crutch like unique identifiers, a designated leader, or global information on the network.

Our solution has many potential uses, given that average consensus primitives underpin many applications studied in distributed control. In contrast with the classic approaches used in this domain, we take an algorithmic stance, grounded in the theory of anonymous computation [1, 2, 17] and of the algorithmic study of dynamic networks [20]. We argue that the fundamental convex recurrence rule for average consensus, namely, the Metropolis rule, cannot be implemented in a fully distributed and decentralized setting when the network is subject to unpredictable change. Our solution consists in relaxing the *convexity* constraint, resulting in an *affine* recurrence rule for average consensus that is algorithmically implementable in any networked multi-agent system with a time-varying communication graph, under the sole constraint of bidirectional links and permanent connectivity.

In the long version of our paper, we will relax the latter assumption and show that $(B \geqslant 1)$-bounded connectivity – where it is only each *matrix product* $\boldsymbol{A}(t+B-1)\cdots\boldsymbol{A}(t)$ that is assumed irreducible – only delays our convergence bound by a factor B. An open question is whether one can design a fully decentralized average consensus algorithm that doesn't break the convex hull of the estimates, or whether that is impossible.

References

1 Dana Angluin. Local and global properties in networks of processors (extended abstract). In R. E. Miller, S Ginsburg, W. A. Burkhard, and R. J. Lipton, editors, *Proceedings of the twelfth annual ACM symposium on Theory of computing - STOC '80*, pages 82–93. ACM Press, 1980. doi:10.1145/800141.804655.

2 Paolo Boldi and Sebastiano Vigna. An effective characterization of computability in anonymous networks. In Jennifer Welch, editor, *DISC 2001: Distributed Computing*, volume 2180 of *Lecture Notes in Computer Science*, pages 33–47. Springer Berlin Heidelberg, 2001. doi:10.1007/3-540-45414-4_3.

3 Florence Bénézit, Vincent D. Blondel, Patrick Thiran, John N. Tsitsiklis, and Martin Vetterli. Weighted gossip: Distributed averaging using non-doubly stochastic matrices. In *2010 IEEE International Symposium on Information Theory*, pages 1753–1757, June 2010. doi:10.1109/ISIT.2010.5513273.

4 Themistoklis Charalambous, Michael G. Rabbat, Mikael Johansson, and Christoforos N. Hadjicostis. Distributed Finite-Time Computation of Digraph Parameters: Left-Eigenvector, Out-Degree and Spectrum. *IEEE Transactions on Control of Network Systems*, 3(2):137–148, 2016. doi:10.1109/TCNS.2015.2428411.

5 Bernadette Charron-Bost. Geometric Bounds for Convergence Rates of Averaging Algorithms. *Information and Computation*, 2022. (To appear). arXiv:2007.04837.

6 Bernadette Charron-Bost and Patrick Lambein-Monette. Randomization and quantization for average consensus. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 3716–3721. IEEE, December 2018. doi:10.1109/CDC.2018.8619817.

7 Samprit Chatterjee and Eugene Seneta. Towards Consensus: Some Convergence Theorems on Repeated Averaging. *Journal of Applied Probability*, 14(1):89–97, 1977. doi:10.2307/3213262.

8 George Cybenko. Dynamic load balancing for distributed memory multiprocessors. *Journal of Parallel and Distributed Computing*, 7(2):279–301, 1989. doi:10.1016/0743-7315(89)90021-X.

9 Louis Penet de Monterno, Bernadette Charron-Bost, and Stephan Merz. Synchronization modulo k in dynamic networks. In *Stabilization, Safety, and Security of Distributed Systems - 23rd International Symposium, SSS 2021, Virtual Event, November 17-20, 2021, Proceedings*, volume 13046 of *Lecture Notes in Computer Science*, pages 425–439. Springer, 2021. doi:10.1007/978-3-030-91081-5_28.

10 Morris H. DeGroot. Reaching a Consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974. doi:10.2307/2285509.

11 Persi Diaconis and Daniel Stroock. Geometric bounds for eigenvalues of markov chains. *The Annals of Applied Probability*, 1(1):36–61, February 1991. doi:10.1214/aoap/1177005980.

12 Michael Dinitz, Jeremy Fineman, Seth Gilbert, and Calvin Newport. Load balancing with bounded convergence in dynamic networks. In *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pages 1–9, Atlanta, GA, USA, 2017. IEEE. doi:10.1109/INFOCOM.2017.8057000.

13 Alejandro D. Dominguez-Garcia, Stanton T. Cady, and Christoforos N. Hadjicostis. Decentralized optimal dispatch of distributed energy resources. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 3688–3693. IEEE, December 2012. ZSCC: 0000166. doi:10/ggd8nx.

14 Balazs Gerencser and Julien M. Hendrickx. Push-sum with transmission failures. *IEEE Transactions on Automatic Control*, 64(3):1019–1033, March 2019. doi:10.1109/TAC.2018.2836861.

15 Wilfred Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, April 1970. doi:10.1093/biomet/57.1.97.

16 Julien M. Hendrickx, Alex Olshevsky, and John N. Tsitsiklis. Distributed anonymous discrete function computation. *IEEE Transactions on Automatic Control*, 56(10):2276–2289, October 2011. doi:10.1109/TAC.2011.2163874.

17 Julien M. Hendrickx and John N. Tsitsiklis. Fundamental limitations for anonymous distributed systems with broadcast communications. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 9–16. IEEE, September 2015. doi:10.1109/ALLERTON.2015.7446980.

18 Ali Jadbabaie, Jie Lin, and A. Stephen Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003. doi:10.1109/TAC.2003.812781.

19 David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491, October 2003. doi:10/fcmmkg.

20 Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *Proceedings of the 42nd ACM symposium on Theory of computing - STOC '10*, page 513. ACM Press, 2010. doi:10.1145/1806689.1806760.

21 Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, March 1953. doi:10.2172/4390578.

22 Luc Moreau. Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50(2):169–182, 2005. doi:10.1109/TAC.2004.841888.

23 Damon Mosk-Aoyama and Devavrat Shah. Computing separable functions via gossip. In *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*, PODC '06, pages 113–122. ACM Press, July 2006. doi:10.1145/1146381.1146401.

24 Angelia Nedić and Alex Olshevsky. Distributed optimization over time-varying directed graphs. *IEEE Transactions on Automatic Control*, 60(3):601–615, 2014. doi:10/f63582.

25 Angelia Nedić, Alex Olshevsky, Asuman Ozdaglar, and John N. Tsitsiklis. On Distributed Averaging Algorithms and Quantization Effects. *IEEE Transactions on Automatic Control*, 54(11):2506–2517, 2009. doi:10.1109/TAC.2009.2031203.

26 Angelia Nedić, Alex Olshevsky, and Michael G. Rabbat. Network Topology and Communication-Computation Tradeoffs in Decentralized Optimization. *Proceedings of the IEEE*, 106(5):953–976, 2018. doi:10.1109/JPROC.2018.2817461.

27 Reza Olfati-Saber and Jeff S. Shamma. Consensus filters for sensor networks and distributed sensor fusion. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 6698–6703, December 2005. doi:10/c338d4.

28 Alex Olshevsky. Linear Time Average Consensus and Distributed Optimization on Fixed Graphs. *SIAM Journal on Control and Optimization*, 55(6):3990–4014, 2017. doi:10.1137/16M1076629.

29 Alex Olshevsky and John N. Tsitsiklis. Convergence Speed in Distributed Consensus and Averaging. *SIAM Review*, 53(4):747–772, 2011. doi:10.1137/110837462.

30 W. Ren. Consensus strategies for cooperative control of vehicle formations. *IET Control Theory & Applications*, 1(2):505–512, 2007. doi:10.1049/iet-cta:20050401.

31 Shreyas Sundaram and Christoforos N. Hadjicostis. Finite-Time Distributed Consensus in Graphs with Time-Invariant Topologies. In *2007 American Control Conference*, pages 711–716, 2007. doi:10.1109/ACC.2007.4282726.

32 Ichiro Suzuki and Masafumi Yamashita. Distributed Anonymous Mobile Robots: Formation of Geometric Patterns. *SIAM Journal on Computing*, 28(4):1347–1363, 1999. doi:10.1137/S009753979628292X.

33 John N. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1984. URL: https://www.mit.edu/~jnt/Papers/PhD-84-jnt.pdf.

34 John N. Tsitsiklis, Dimitri P. Bertsekas, and Michael Athans. Distributed Asynchronous Deterministic and Stochastic Gradient Optimization Algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986. doi:10.1109/TAC.1986.1104412.

35 Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53(1):65–78, 2004. doi:10.1016/j.sysconle.2004.02.022.

36 Lin Xiao, Stephen Boyd, and Seung-Jean Kim. Distributed average consensus with least-mean-square deviation. *Journal of Parallel and Distributed Computing*, 67(1):33–46, 2007. doi:10/bmq2t4.

37 Lin Xiao, Stephen Boyd, and Sanjay Lall. A Scheme for Robust Distributed Sensor Fusion Based on Average Consensus. In *Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, pages 63–70, Los Angeles, CA, USA, 2005. IEEE. doi:10.1109/IPSN.2005.1440896.