Automatic Performance Estimation for Decentralized Optimization

Sébastien Colla and Julien M. Hendrickx

***Abstract*— We present a methodology to automatically compute worst-case performance bounds for a large class of first-order decentralized optimization algorithms. These algorithms aim at minimizing the average of local functions that are distributed across a network of agents. They typically combine local computations and consensus steps. Our methodology is based on the approach of Performance Estimation Problem (PEP), which allows computing the worst-case performance and a worst-case instance of first-order optimization algorithms by solving an SDP. We propose two ways of representing consensus steps in PEPs, which allow writing and solving PEPs for decentralized optimization. The first formulation is exact but specific to a given averaging matrix. The second formulation is a relaxation but provides guarantees valid over an entire class of averaging matrices, characterized by their spectral range. This formulation often allows recovering a posteriori the worst possible averaging matrix for the given algorithm. We apply our methodology to three different decentralized methods. For each of them, we obtain numerically tight worst-case performance bounds that significantly improve on the existing ones, as well as insights about the parameters tuning and the worst communication networks.**

I. Introduction

The goal of this paper is to develop a methodology that automatically provides numerically tight performance bounds for first-order decentralized methods on convex functions and to demonstrate its usefulness on different existing methods. Decentralized optimization has received an increasing attention due to its useful applications in large-scale machine learning and sensor networks, see e.g. [1] for a survey. In decentralized methods for separable objective functions, we consider a set of agents $\{1,\dots,N\}$, working together to solve the following optimization problem:

$$\underset{x\,\in\,\mathbb{R}^d}{\text{minimize}} \quad f(x)=\frac{1}{N}\sum_{i=1}^{N} f_i(x), \tag{1}$$

where $f_i:\mathbb{R}^d\to\mathbb{R}$ is the private function locally held by agent i. To achieve this goal, each agent i holds its own version x_i of the decision variable $x\in\mathbb{R}^d$. Agents perform local computations and exchange local information with their neighbors to come to an agreement on the minimizer x^* of the global function f. Exchanges of information often take the form of an average consensus step on some quantity, e.g., on the x_i. This corresponds to a multiplication by an averaging matrix $W\in\mathbb{R}^{N\times N}$, typically assumed symmetric and *doubly stochastic*, i.e., a nonnegative matrix whose rows and columns sum to one. This matrix W indicates both the topology of the network of agents and the weights they are using during the average consensus. Therefore, we call it network or averaging matrix, without distinction.

One of the simplest decentralized optimization method is the distributed (sub)gradient descent (DGD) [2] where agents successively perform an average consensus step (2) and a local gradient step (3):

$$y_i^k=\sum_{j=1}^{N} w_{ij}x_j^k, \tag{2}$$

$$x_i^{k+1}=y_i^k-\alpha^k\nabla f_i(x_i^k), \tag{3}$$

for step-sizes $\alpha^k>0$. Numerous other methods rely on the interplay of gradient and average consensus steps, such as EXTRA [3], DIGing [4], and NIDS [5]. The convergence results of DIGing holds for time-varying network matrices and the algorithm can also be extended to handle directed communications [4]. Different other algorithms can handle directed communications [6], [7]. There also are accelerated decentralized methods, such as the accelerated distributed Nesterov gradient descent (Acc-DNGD) [8]. Furthermore, average consensus is used in different distributed primal dual methods such as DDA [9], MSDA [10] or MSPD [11], though not directly in all, e.g., D-ADMM [12], [13]. We note in particular recent results have reached optimal performance for certain specific performance criteria and classes of functions [14], [11]. However, many questions and challenges remain open in the design and analysis of decentralized optimization method. For further information on the topic, we refer the reader to the introduction of this recent thesis [15] in the field.

The quality of an optimization method is often evaluated *via* a worst-case guarantee. Having accurate performance bounds for a decentralized algorithm is important to correctly understand the impact of its parameters and the network topology on its performance and to compare it fairly with other algorithms. However, obtaining these guarantees using theoretical proofs can often be a challenging task, requiring combining the impact of the optimization component and the interconnection network, sometimes resulting in bounds that are conservative or overly complex. For example, we will show in Section IV that the existing performance bounds for DGD or DIGing are significantly worse than the actual worst-cases.

S. Colla and J. M. Hendrickx are with the ICTEAM institute, UCLouvain, Louvain-la-Neuve, Belgium. S. Colla is supported by the French Community of Belgium through a FRIA fellowship (F.R.S.-FNRS). J. M. Hendrickx is supported by the "RevealFlight" Concerted Research Action (ARC) of the Federation Wallonie-Bruxelles and by the Incentive Grant for Scientific Research (MIS) "Learning from Pairwise Comparisons" of the F.R.S.-FNRS. Email addresses: sebastien.colla@uclouvain.be, julien.hendrickx@uclouvain.be

A. Contributions

We propose a new analysis methodology allowing to compute numerically tight worst-case performance bounds of decentralized optimization methods. This methodology is based on an alternative computational approach that finds a worst-case performance guarantee of an algorithm by solving an optimization problem, known as the performance estimation problem (PEP), see Section II for more details. The PEP approach has led to many results in centralized optimization, see e.g. [16], [17], but it has never been exploited in decentralized optimization. The current PEP framework lacks ways of representing the communications between the agents. Therefore, we propose two formulations of the average consensus steps that can be embedded in a solvable PEP. Both formulations are presented in Section III. The first one uses a given averaging matrix W and is exact. It can be used in PEP with any matrix W and leads to performance bounds that are tight, but specific to the given matrix. However, these bounds are too specific to understand the general behavior of the algorithm. Our second formulation thus considers entire spectral classes of symmetric matrices, as often found in the literature. This allows the PEP problem to obtain spectral upper bounds on the performance that are valid over an entire spectral class of network matrices and to look for the worst matrix in the given class. Although this formulation is a relaxation, we observe tight results in most cases, see Section IV. This spectral formulation is our main methodological contribution.

Using these two new formulations, the PEP approach can be applied directly to a large class of first-order decentralized algorithms, in a wide range of settings and problems. We demonstrate these new formulations by analyzing the worst-case performance of DGD [1], DIGing [4] and Acc-DNGD [8] in Section IV. For all three algorithms, the spectral formulation leads to tight spectral performance bounds and we observe that these bounds are actually independent of the number of agents N (when $N \geq 2$). For DGD and DIGing, these spectral bounds significantly improve on the existing theoretical ones. For DIGing, we also show how to use the PEP approach to obtain linear convergence rate guarantees on an infinite horizon. For Acc-DNGD, we use our tool to analyze the impact of the parameters and nuance the conjecture from [8] about the asymptotic convergence rate $\mathcal{O}(\frac{1}{K^2})$, where K is the total number of iterations.

One of the advantages of this new methodology is its great flexibility, allowing to answer many different questions, for example, by changing the class of functions or the performance criterion. In future works, this tool can easily be used to analyze the effect of inexact communications or gradient computations. This methodology could also consider performance criteria that not only account for efficiency but also robustness against errors in gradient computations or communications in order to help designing algorithms with the best efficiency-robustness trade-off.

A preliminary version of these results was published in [18]. Our main new contributions with respect to [18] are (i) the proof of the necessary constraints used in our spectral formulation for any dimension d of the variables $x_i \in \mathbb{R}^d$; (ii) the technique to find the worst averaging matrix based on the worst-case solutions of the spectral formulation; (iii) a wider demonstration of the methodology by analyzing DIGing and Acc-DNGD algorithms in addition of DGD.

B. Related work

An alternative approach with similar motivations for automated performance evaluation and inspired by dynamical systems concepts is proposed in [19]. Integral quadratic constraints (IQC), usually used to obtain stability guarantees on complex dynamical systems, are adapted to obtain sufficient conditions for the convergence of optimization algorithms. It provides infinite-horizon linear rates of convergence, based on relatively small size problems, but it only applies when the convergence is geometric. In comparison, the PEP approach computes the worst-case performance on a finite horizon, and therefore, the size of the problem grows with the limit of the horizon. However, this allows PEP to analyze non-geometric convergence and the impact of time-varying properties. Unlike PEP, the IQC approach offers no a priori guarantee of tightness, though it turns out to be tight in certain situations.

An application of the IQC methodology to decentralized optimization is presented in [20] and is also exploited for designing a new decentralized algorithm that achieves a faster worst-case linear convergence rate in the smooth strongly convex case. This IQC formulation uses problems whose size is independent of the number of agents N and it considers a fixed framework of decentralized algorithms that embeds a lot of explicit first-order methods such as DIGing [4], EXTRA [3] and NIDS [5]. This methodology cannot be directly applied to DGD, nor to smooth convex functions or any other situation that does not have a geometric convergence. Our PEP approach applies directly to any decentralized method (implicit or explicit) that involves consensus steps in the form of a matrix product, without the need to cast the method into a specific framework. This makes PEP very modular and user-friendly, in particular with the PESTO toolbox [21].

II. General PEP approach

In principle, a tight performance bound on an algorithm could be obtained by running it on every single instance - function and initial condition - allowed by the setting considered and selecting the worst performance obtained. This would also directly provide an example of "worst" instance if it exists. The performance estimation problem (PEP) formulates this abstract idea as a real optimization problem that maximizes the error measure of the algorithm result, over all possible functions and initial conditions allowed [22]. This optimization problem is inherently infinite-dimensional, as it contains a continuous function among its variables. Nevertheless, Taylor et al. have shown [16], [17] that PEP can be solved exactly using an SDP formulation, for a wide class of centralized first-order algorithms and different classes of functions.

The main ingredients of a centralized PEP are: (i) a performance measure $\mathcal{P}$, e.g. $f(x^k) - f(x^*)$; (ii) a class of functions $\mathcal{F}$, e.g. the class $\mathcal{F}_{\mu,L}$ of μ-strongly convex and L-smooth functions; (iii) an optimization method $\mathcal{M}$ on class $\mathcal{F}$; (iv) a set $\mathcal{I}^0$ of initial conditions, e.g. $\|x^0 - x^*\|^2 \leq 1$. These ingredients are organized into an optimization problem as

$$\begin{aligned} \sup_{f,x^0,\dots,x^K,x^*} \quad & \mathcal{P}\big(f, x^0, \dots, x^K, x^*\big) \\ \text{s.t.} \quad & f \in \mathcal{F}, \qquad x^* \in \operatorname{argmin} f, \\ & x^k \text{ are iterates from method } \mathcal{M} \text{ applied on } f, \\ & \mathcal{I}^0 \text{ holds.} \end{aligned} \tag{4}$$

To overcome the infinite dimension of variable f, the problem can be discretized into $\{\big(x^k, g^k, f^k\big)\}_{k\in I}$, where g^k and f^k are respectively the (sub)gradient and the function value of f at point k and $I = \{0, \dots, K, *\}$. Then, the constraint $f \in \mathcal{F}$ is replaced by interpolation conditions ensuring that there exists a function of class $\mathcal{F}$ which interpolates those data points $\{\big(x^k, g^k, f^k\big)\}_{k\in I}$, i.e. these values are consistent with an actual function. Such constraints are provided for many different classes of functions in [17, Section 3]. They are generally quadratics and potentially non-convex in the iterates and the gradients vectors, but they are linear in the scalar products of these and in the function values. The same holds true for most classical performance criteria and initial conditions. We can then consider these scalar products directly as decision variables of the PEP. For this purpose, we define a Gram matrix G that contains scalar products between all vectors, e.g. the iterates $x^k \in \mathbb{R}^d$ and subgradients $g^k \in \mathbb{R}^d$.

$$G = P^T P, \text{ with } P = \big[g^0 \dots g^K g^* x^0 \dots x^K x^*\big].$$

By definition, G is symmetric and positive semidefinite. Moreover, to every matrix $G \succeq 0$ corresponds a matrix P whose has a number of rows equal to rank G. We can show (see [17]) that the reformulation of (4) with G is lossless provided that we do not impose the dimension d in (4), and indeed look for the worst-case over all possible dimensions (imposing the dimension would correspond to adding a typically less tractable rank constraint on G). The idea is therefore to formulate a PEP of the form of (4) as an equivalent positive semidefinite program (SDP) using the Gram matrix G and the vector of functional values $f_v = [f^k]_{k\in I}$ as variables.

This SDP formulation is convenient because it can be solved numerically to global optimality, leading to a tight worst-case bound, and it also provides the worst-case solution over all possible problem dimensions.We refer the reader to [17] for more details about the SDP formulation of PEP, including ways of reducing the size of matrix G. However, the dimension of G always depends on the number of iterations K. From a solution G, f_v of the SDP formulation, we can construct a solution for the discretized variables $\{\big(x^k, g^k, f^k\big)\}_{k\in I}$, e.g. using the Cholesky decomposition of G. Since these points satisfy sufficient interpolation constraints, we can also construct a function from $\mathcal{F}$ interpolating these points.

Proposition 1 states sufficient conditions under which a PEP in the form of (4) can be formulated as an SDP. These conditions are satisfied in most PEP settings, allowing to formulate and solve the SDP PEP formulation for a large class of (implicit and explicit) first-order methods, with different classes of functions and performance criteria, see [17]. The proposition uses the following definition.

Definition 1 (Gram-representable [17]): Consider a Gram matrix G and a vector f_v, as defined above. We say that a constraint or an objective is linearly (resp. LMI) Gram-representable if it can be expressed using a finite set of linear (resp. LMI) constraints involving (part of) G and f_v.

Proposition 1 ([17, Proposition 2.6]): If the interpolation constraints of the class of functions $\mathcal{F}$, the satisfaction of the method $\mathcal{M}$, the performance measure $\mathcal{P}$ and the set of constraints $\mathcal{I}$, which includes the initial conditions, are linearly (or LMI) Gram-representable, then, computing the worst-case for criterion $\mathcal{P}$ of method $\mathcal{M}$ after K iterations on objective functions in class $\mathcal{F}$ with constraints $\mathcal{I}$ can be formulated as an SDP, with $G \succeq 0$ and f_v as variables.
This remains valid when the objective function is the sum of N sub-functions being each in a class of functions with linearly (or LMI) Gram representable interpolation constraints.

Remark: In [17], Definition 1 and Proposition 1 were only formulated for linearly Gram-representable constraints, but their extension to LMI Gram-representable constraints is direct. Such constraints appear in the analysis of consensus steps with spectral classes of network matrices.

PEP techniques allowed answering several important questions in optimization, see e.g. the list in [23], and to make important progress in the tuning of certain algorithms including the well-known centralized gradient-descent. It was further exploited to design optimal first-order methods: OGM for smooth convex optimization [24] and its extension ITEM for smooth strongly convex optimization [25].

It can also be used to deduce proofs about the performance of the algorithms [26]. It has been made widely accessible *via* a Matlab [21] and Python [27] toolbox. However, PEPs have never been used to study the performance of decentralized methods.

III. Representing consensus steps in PEP

The main missing block to develop a PEP formulation for decentralized methods is to find a proper way of representing the consensus steps of the methods in the SDP PEP. In PEP for decentralized methods, we must find the worst-case for each of the N local functions f_i and the N sequences of local iterates $x_i^0 \dots x_i^K$ $(i = 1, \dots, N)$. The same techniques as in Section II can be applied to discretize the problem, using proper interpolation conditions on each local function. We also want to use a Gram matrix of scalar products to reformulate it as an SDP that can be solved efficiently. Therefore, according to Proposition 1, we need to use linearly Gram representable performance criterion, classes of functions and initial conditions. Moreover, we also need to find a linearly or LMI Gram-representable way of expressing the

updates of the decentralized method to be analyzed. There is currently no representation of the consensus steps that can be embedded in the SDP formulation. Therefore, the rest of this section proposes ways to represent agent interactions in the SDP PEP formulation and thus provides the missing block for analyzing decentralized first-order optimization methods with PEP.

Agent interactions are part of any decentralized methods and often take place *via* a weighted averaging, which can be described as a consensus step of the following form, similar to that used in DGD (2):

$$y_i = \sum_{j=1}^{N} w_{ij} x_j, \qquad \text{for all } i \in \{1, \dots, N\}, \tag{5}$$

where x_j can represent any vector in $\mathbb{R}^d$ held by agent j, e.g., its local iterates in the case of DGD or something else in more advanced methods. Vector $y_i \in \mathbb{R}^d$ is an auxiliary variable that represents the result of the interaction and $W \in \mathbb{R}^{N\times N}$ is the averaging matrix. This form of communication is used in many decentralized methods such as DGD [2], DIGing [4], EXTRA [3], NIDS [5] and the results presented in this section can be exploited for all these methods.

When K different consensus steps are involved in the algorithm, we observe that the Gram matrix G of all scalar products contains the submatrix G_c:

$$G_c = P_c^T P_c, \tag{6}$$
$$\text{with } P_c = \left[x_0^0 \dots x_N^0 \dots x_0^K \dots x_N^K \; y_0^0 \dots y_N^0 \dots y_0^K \dots y_N^K\right].$$

We will see that the new constraints we propose to represent the consensus steps are linearly or LMI Gram-representable in G_c, and then also linearly or LMI Gram-representable in G.

A. Averaging matrix given a priori

When the averaging matrix is given *a priori*, the consensus step (5) corresponds to a set of linear equality constraints on y_i and x_j, which are equivalent to the constraints

$$\Big(y_i - \sum_{j=1}^{N} w_{ij} x_j\Big)^T \Big(y_i - \sum_{j=1}^{N} w_{ij} x_j\Big) = 0, \text{ for } i = 1, \dots, N.$$

These constraints only involve scalar products from G_c (6) and are thus linearly Gram-representable. They can therefore be used in the SDP formulation of a PEP, see Proposition 1. Alternatively, linear equality constraints (5) can also be used to substitute the values of y_i in the problem, allowing to reduce its size. This solution is also preferable for numerical reasons. In any case, this allows writing PEPs that provide exact worst-case performances for the given decentralized method and the specific averaging matrix given *a priori*. We call this the *exact PEP formulation*. It can be applied to any matrix W, and not only for symmetric or doubly stochastic ones. This can be useful for trying different network matrices and observing their impact on the worst-case performance of the algorithm. It will serve as an exact comparison baseline in the numerical experiments of Section IV. The next section presents a way of representing communications in PEP that allows obtaining more general performance guarantees, valid over entire classes of network matrices and not only for a specific one.

B. Averaging matrix as a variable

We now consider that the matrix W is not given *a priori*, but is *one of the decision variables* of the performance estimation problem with bounds on its possible eigenvalues. Hence, the PEP also looks for the worst averaging matrix among all the possible ones. Typically, the literature considers matrices that are symmetric, doubly-stochastic and whose all eigenvalues (except 1) are in a given range. We were unable to represent this class of matrices in the SDP PEP formulation, in particular the non-negativity assumption embedded in the doubly stochasticity. Therefore, we will consider a slightly more general class of matrices, where we relax this non-negativity assumption.

Definition 2 (Generalized doubly stochastic matrix [28]): A matrix $W \in \mathbb{R}^{N\times N}$ is *generalized doubly stochastic* if its rows and columns sum to one, i.e., if

$$\sum_{i=1}^{N} w_{ij} = 1, \qquad \sum_{i=1}^{N} w_{ji} = 1, \qquad \text{for } j = 1, \dots, N.$$

The resulting worst matrix of the PEP can thus have negative elements. However, the worst-case guarantees that it provides is still valid for non-negative matrices. Moreover, we note that most results from the literature exploiting spectral information of stochastic matrices do in fact not use the non-negativity of W and are thus really about generalized stochastic matrices, see e.g. [1], [3], [5]. We analyze the impact of this relaxation in the case of DGD in Section IV-A.

Formally, the search space for W is restricted by the following constraints, for each agent $i \in \{1, \dots, N\}$ and each consensus steps $k \in \{1, \dots, K\}$,

$$y_i^k = \sum_{j=1}^{N} w_{ij} x_j^k, \tag{7}$$
$$W \in \mathcal{W}(\lambda^-, \lambda^+), \tag{8}$$

where $\mathcal{W}(\lambda^-, \lambda^+)$ is the set of real, symmetric and generalized doubly stochastic $N \times N$ matrices that have their eigenvalues between λ^- and λ^+, except for $\lambda_1 = 1$:

$$\lambda^- \leq \lambda_N \leq \dots \leq \lambda_2 \leq \lambda^+ \quad \text{where } \lambda^-, \lambda^+ \in (1, 1)\ .$$

We do not have a direct way for representing constraints (7) and (8) in an LMI Gram-representable manner that can be embedded in an SDP PEP, hence we will use a relaxation of these constraints that we will see in Section IV is often close to tight. From constraints (7) and (8), we derive new necessary conditions involving only variables y_i^k and x_i^k, allowing to eliminate W from the problem. We also show that these new constraints can be expressed in terms of G_c (6) and are therefore Gram-representable.

Notations: Let $x_i^k, y_i^k \in \mathbb{R}^d$ be the local variables of agent i at iteration k of an algorithm. Let X^j, $Y^j \in \mathbb{R}^{N\times K}$ be matrices containing the j^{th} component of each of these local variables:

$$X^j_{i,k} = (x_i^k)_j \quad \text{and} \quad Y^j_{i,k} = (y_i^k)_j \qquad \text{for} \quad \begin{smallmatrix} j=1,\dots,d \\ i=1,\dots,N \\ k=1,\dots,K \end{smallmatrix}$$

Each column corresponds thus to a different consensus step k, and each row to a different agent i. Using this notation, the consensus steps constraints (7) can simply be written as d matrix equations: $Y^j = WX^j$ for each $j = 1,\dots,d$.

Moreover, we can stack each X^j and Y^j vertically in matrices $X, Y \in \mathbb{R}^{Nd\times K}$, and write all the consensus steps (7) with one matrix equation:

$$\underbrace{\begin{bmatrix} Y^1 \\ \vdots \\ Y^d \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} W & & 0 \\ & \ddots & \\ 0 & & W \end{bmatrix}}_{\tilde{W}} \underbrace{\begin{bmatrix} X^1 \\ \vdots \\ X^d \end{bmatrix}}_{X}.$$

The matrix $\tilde{W} \in \mathbb{R}^{Nd\times Nd}$ is block-diagonal repeating d times W and can be written as $\tilde{W} = (I_d \otimes W)$, where $I_d \in \mathbb{R}^{d\times d}$ is the identity matrix and $\otimes$ denotes the Kronecker product. We decompose matrices X and Y in average and centered parts:

$$X = (\overline{X} \otimes \mathbf{1}_N) + X_\perp, \qquad Y = (\overline{Y} \otimes \mathbf{1}_N) + Y_\perp,$$

where $\overline{X}$ and $\overline{Y}$ are agents average vectors in $\mathbb{R}^{d\times K}$, defined as $\overline{X}_{\cdot k} = \frac{1}{N}\sum_{i=1}^N x_i^k$, $\overline{Y}_{\cdot k} = \frac{1}{N}\sum_{i=1}^N y_i^k$ for $k = 1,\dots,K$ and $\mathbf{1}_N = [1\dots 1]^T \in \mathbb{R}^N$. The centered matrices $X_\perp, Y_\perp \in \mathbb{R}^{Nd\times K}$ have by definition an agent average of zero for each component and iteration: $(I_d \otimes \mathbf{1}_N)^T X_\perp = \mathbf{0}_{d\times K}$.

Gram-representable relaxation of (7) *and* (8)*:*

Theorem 2 (Consensus Constraints): If $Y^j = WX^j$, for every $j = 1,\dots,d$ and for a same matrix $W \in \mathcal{W}(\lambda^-, \lambda^+)$, i.e., if $Y = (I_d \otimes W)X$, then

(i) The matrices X^TY and $X_\perp^T Y_\perp$ are symmetric,

(ii) The following constraints are satisfied

$$\overline{X} = \overline{Y}, \tag{9}$$

$$\lambda^- X_\perp^T X_\perp \;\preceq\; X_\perp^T Y_\perp \;\preceq\; \lambda^+ X_\perp^T X_\perp, \tag{10}$$

$$(Y_\perp - \lambda^- X_\perp)^T (Y_\perp - \lambda^+ X_\perp) \;\preceq\; 0, \tag{11}$$

where the notations $\succeq$ and $\preceq$ denote respectively positive and negative semi-definiteness.

(iii) Constraints (9), (10), (11) are LMI Gram-representable.

Proof: First, we average elements from both sides of the assumption $Y^j = WX^j$ to obtain constraint (9):

$$\overline{Y}_{j\cdot} = \frac{\mathbf{1}^T Y^j}{N} = \frac{\mathbf{1}^T W X^j}{N} = \frac{\mathbf{1}^T X^j}{N} = \overline{X}_{j\cdot} \text{ for } j = 1,\dots,d$$

where $\mathbf{1}^T W = \mathbf{1}^T$ follows from W being generalized doubly stochastic, i.e., its rows and columns sum to one (see Definition 2).

In the sequel, we will consider all the components j at once, and then we use the notation $Y = \tilde{W}X$, where $\tilde{W} = I_d \otimes W$ is in $\mathbb{R}^{Nd\times Nd}$. Since $\tilde{W}$ is a block-diagonal matrix repeating d times W, if $W \in \mathcal{W}(\lambda^-, \lambda^+)$, then $\tilde{W} \in \mathcal{W}(\lambda^-, \lambda^+)$.

The symmetry of the matrix X^TY follows from the assumption $Y = \tilde{W}X$, with $\tilde{W}$ symmetric. The same argument shows the symmetry of $X_\perp^T Y_\perp$, because $Y = \tilde{W}X$ and $\overline{X} = \overline{Y}$ imply $Y_\perp = \tilde{W}X_\perp$.

Since the averaging matrix $\tilde{W}$ is real and symmetric, we can take an orthonormal basis $\mathbf{v_1},\dots,\mathbf{v_{Nd}}$ of eigenvectors, corresponding to real eigenvalues $\lambda_1 \geq \cdots \geq \lambda_{Nd}$. Since $\tilde{W}$ is composed of d diagonal blocks of the generalized doubly stochastic matrix W, it has the same eigenvalues as W but with a multiplicity d times larger for each of them. Therefore, its largest eigenvalues are $\lambda_j = 1$ $(j = 1,\dots,d)$ and corresponds to the eigenvectors $\mathbf{v_j} = [0\dots\mathbf{1}_N^T\dots 0]^T$ where the position for $\mathbf{1}_N$ corresponds to the position of the block j in $\tilde{W}$. These eigenvectors can be written in matrix form as $V_{1,\dots,d} = I_d \otimes \mathbf{1}_N$. By assumption, other eigenvalues are such that

$$\lambda^- \leq \lambda_i \leq \lambda^+ \text{ for } i = d+1,\dots,Nd, \text{ with } \lambda^-, \lambda^+ \in (-1, 1).$$

Let us now consider a combination $X_\perp z$ of the columns of the matrix $X_\perp$, for an arbitrary $z \in \mathbb{R}^K$. It can be decomposed in the eigenvector basis of $\tilde{W}$, and used to express the combination $Y_\perp z$ as well:

$$X_\perp z = \sum_{i=d+1}^{Nd} \gamma_i \mathbf{v}_i, \text{ and } Y_\perp z = \tilde{W}X_\perp z = \sum_{i=d+1}^{Nd} \gamma_i \lambda_i \mathbf{v}_i, \tag{12}$$

where γ_i are real coefficients. These coefficients for $\mathbf{v_1},\dots,\mathbf{v_d}$ are zero because $X_\perp z$ is orthogonal to these eigenvectors associated with eigenvalue $\lambda_j = 1$. Indeed $X_\perp z$ is centered with respect to the agents, for any component j or iteration k and then we have

$$(I_d \otimes \mathbf{1}_N)^T X_\perp = V_{1,\dots,d}^T X_\perp = \mathbf{0}_{d\times K}.$$

Using the decomposition (12) to compute the scalar product $z^T X_\perp Y_\perp z$ for any $z \in \mathbb{R}^K$ leads to the following scalar inequalities

$$\begin{aligned} z^T X_\perp^T Y_\perp z = \sum_{i=d+1}^{Nd} \gamma_i^2 \lambda_i &\geq \lambda^- z^T X_\perp^T X_\perp z, \\ &\leq \lambda^+ z^T X_\perp^T X_\perp z. \end{aligned}$$

Having these inequalities satisfied for all $z \in \mathbb{R}^K$, is equivalent to (10). In the same way, (11) is obtained by verifying that the following inequality holds for all $z \in \mathbb{R}^K$:

$$(Y_\perp z - \lambda^- X_\perp z)^T (Y_\perp z - \lambda^+ X_\perp z) \leq 0.$$

This can be done by substituting $X_\perp z$ and $Y_\perp z$ using equation (12), and by using the bounds on λ_i $(i = d+1,\dots,Nd)$. Finally, we prove part (iii) of the theorem. Constraint (9) is linearly (and thus also LMI) Gram-representable because it can be expressed using only elements of G_c (6), i.e. scalar product between x_i^k and y_j^k,

$$\left(\frac{1}{N}\sum_i x_i^k - y_i^k\right)^T \left(\frac{1}{N}\sum_j x_j^k - y_j^k\right) = 0 \quad \text{for } k = 1,\dots,K.$$

Constraints (10) and (11) are LMI Gram-representable because they are LMIs whose entries can be defined using only the entries of G_c, which is a submatrix of the full Gram

matrix G of scalar products. For example, the entry k,l of $X_\perp^T X_\perp$ can be expressed as the scalar product of columns k and l of $X_\perp$

$$\left(X_\perp^T\right)_{k\cdot}(X_\perp)_{\cdot l} = \sum_{i=1}^{N}\Big(x_i^k - \frac{1}{N}\sum_j x_j^k\Big)^T\Big(x_i^l - \frac{1}{N}\sum_j x_j^l\Big).$$

■

Using Theorem 2, we can relax constraints (7) and (8) and replace them by (9), (10) and (11), which are LMI Gram-representable. Then, Proposition 1 allows to write a relaxed SDP formulation of a PEP providing worst-case results valid for the entire spectral class of matrices $\mathcal{W}(\lambda^-,\lambda^+)$. We call this formulation the *spectral PEP formulation* and its results the *spectral worst-case*. This SDP formulation has matrix $G \succeq 0$ and vectors $f_{i,v}$ $(i = 1,...,N)$ as decision variables. The vector $f_{i,v}$ contains the function values of f_i at the different iterates x_i^k $(k = 1,...,K)$. The values in G correspond to the scalar products of the iterates (x_i^k, y_i^k) and the gradients (g_i^k) of the different agents.

When different averaging matrices are used for different sets of consensus steps, the constraints from Theorem 2 can be applied independently to each set of consensus steps.

Constraint (9) is related to the stochasticity of the averaging matrix and imposes that variable x has the same agents average as y, for each consensus step and for each component. Linear matrix inequality constraints (10) and (11) imply in particular scalar constraints for the diagonal elements. They correspond to independent constraints for each consensus step, i.e., for each column $x_\perp$ and $y_\perp$ of matrices $X_\perp$, $Y_\perp$:

$$\lambda^- x_\perp^T x_\perp \le x_\perp^T y_\perp \le \lambda^+ x_\perp^T x_\perp,$$
$$(y_\perp - \lambda^- x_\perp)^T(y_\perp - \lambda^+ x_\perp) \le 0.$$

These constraints imply in particular that

$$y_\perp^T y_\perp \le \lambda_{\max}^2\, x_\perp^T x_\perp, \quad \text{where } \lambda_{\max} = \max(|\lambda^-|,|\lambda^+|),$$

meaning that the disagreement between the agents, measured by $y_\perp^T y_\perp$ for y and $x_\perp^T x_\perp$ for x, is reduced by a factor $\lambda_{\max}^2 \in [0,1)$ after a consensus. But constraints (10) and (11) also allow linking different consensus steps to each other, *via* the impact of off-diagonal terms, in order to exploit the fact that these steps use the same averaging matrix.

We can also interpret constraints (10) and (11) as a sum over all the dimensions $j = 1,\dots,d$. Each term of this sum corresponds to the same constraint expression as (10) and (11), but applied only to $X_\perp^j$ and $Y_\perp^j \in \mathbb{R}^{N\times K}$. Indeed, any product involving $X_\perp$ or $Y_\perp \in \mathbb{R}^{Nd\times K}$ can be written as a sum over the dimensions d. For instance, for $X_\perp^T X_\perp$, we have

$$X_\perp^T X_\perp = \sum_{j=1}^{d}(X_\perp^j)^T X_\perp^j.$$

The following proposition show that constraint (10) is redundant when we consider a symmetric range of eigenvalue, i.e., when $\lambda^+ = -\lambda^-$.

Proposition 3 (Symmetric range of eigenvalues): If $\lambda^+ = -\lambda^- = \lambda \in [0,1]$, then LMI constraints (10) and (11) from Theorem 2 are equivalent to

$$Y_\perp^T Y_\perp \preceq \lambda^2 X_\perp^T X_\perp. \tag{13}$$

Proof: Constraint (13) is equivalent to (11) when $\lambda^+ = -\lambda^- = \lambda$. Constraint (10) with $\lambda^+ = -\lambda^- = \lambda$ becomes

$$-\lambda X_\perp^T X_\perp \;\preceq\; X_\perp^T Y_\perp \;\preceq\; \lambda X_\perp^T X_\perp, \tag{14}$$

We now show that constraint (14) is implied by (13), which achieves the proof. When $\lambda \ge 0$, constraint (13) is equivalent to the following bound on $\|Y_\perp z\|$, for any $z \in \mathbb{R}^K$,

$$\|Y_\perp z\| \le \lambda \|X_\perp z\| \qquad \text{for any } z \in \mathbb{R}^K. \tag{15}$$

Moreover, the scalar product between $X_\perp z$ and $Y_\perp z$ can be bounded, for any $z \in \mathbb{R}^K$, using the Cauchy-Schwarz inequality

$$-\|X_\perp z\|\;\|Y_\perp z\| \le (X_\perp z)^T(Y_\perp z) \le \|X_\perp z\|\;\|Y_\perp z\|. \tag{16}$$

We can combine inequality (16) with (15) and obtain

$$-\lambda z^T X_\perp^T X_\perp z \le z^T X_\perp^T Y_\perp z \le \lambda z^T X_\perp^T X_\perp z, \text{ for any } z \in \mathbb{R}^K,$$

which is equivalent to (14). ■

Proposition 3 means that when we consider a class of matrices with a symmetric range of eigenvalues, i.e., $\mathcal{W}(-\lambda,\lambda)$, we can remove constraint (10) from the spectral PEP formulation, without modifying its result. Other constraints from Theorem 2, including the constraints of symmetry should still be imposed.

Recovering the worst averaging matrix: Theorem 2 provides necessary constraints for describing a set of consensus steps $Y^j = WX^j$ that use an unknown averaging matrix W from a spectral class of symmetric generalized doubly-stochastic matrices $\mathcal{W}(\lambda^-,\lambda^+)$. But these constraints are not known to be sufficient; so even if matrices X and Y satisfy constraints (9), (10), (11), we do not know if there is a matrix $W \in \mathcal{W}(\lambda^-,\lambda^+)$ such that $Y^j = WX^j$ for all $j = 1,\dots,d$. The question of the existence of such a matrix can be expressed as follows: is the optimal cost of the following problem equal to 0?

$$\min_{\hat{W}\,\in\,\mathcal{W}(\lambda^-,\lambda^+)} \quad \|\tilde{Y} - \hat{W}\tilde{X}\|_F \tag{17}$$

where matrices $\tilde{X}, \tilde{Y} \in \mathbb{R}^{N\times Kd}$ stack the $X^j, Y^j \in \mathbb{R}^{N\times K}$ horizontally and thus reshape matrices X and Y. This reshape is needed for recovering a matrix $\hat{W}$ which is identical for every dimension j and that has appropriate size $(N \times N)$. Note that problem (17) can easily be solved since it is an SDP, as the constraint $\hat{W} \in \mathcal{W}(\lambda^-,\lambda^+)$ can be formulated as $\lambda^- I \preceq (\hat{W} - \frac{\mathbf{1}\mathbf{1}^T}{N}) \preceq \lambda^+ I$ together with $\hat{W}^T = \hat{W}$ and $\hat{W}\mathbf{1} = \mathbf{1}$. If the optimal cost of (17) is zero, the optimal value of $\hat{W}$ is a valid worst averaging matrix W. Alternatively, problem (17) without any constraints is a least-square problem whose solution is given by $\hat{W}_p = \tilde{Y}\tilde{X}^+$, where $\tilde{X}^+$ is the pseudo-inverse of $\tilde{X}$. If its remainder $\|\tilde{Y} - \hat{W}_p\tilde{X}\|_F$ is zero, then we check a posteriori that the constraint $\hat{W}_p \in \mathcal{W}(\lambda^-,\lambda^+)$ is satisfied. This was often

the case in our experiments from Section IV and allows to recover rapidly the value of the worst averaging matrix. Note, though, that when $\hat{W}_p$ does not satisfy the required conditions, there could still exists, in some cases, a different valid W, so one should then solve (17).

We have now all the elements to write and solve PEPs for decentralized optimization methods, including ways of representing the consensus steps in a Gram-representable manner, allowing to formulate the PEP as an SDP. In the next section, we demonstrate the methodology by analyzing 3 decentralized methods.

IV. Demonstration of our methodology

Using results from previous sections, we can build two PEP formulations for analyzing the worst-case performance of a large class of decentralized optimization methods: the exact and the spectral formulations. These formulations allow to obtain rapidly and automatically accurate numerical performance bound. We demonstrate the power of the methodology by focusing on the analysis of 3 selected algorithms. For each algorithm, we use the same settings (performance criterion, initial condition, class of functions,...) as its theoretical bound, in order to obtain comparable results. These particular settings are not a limitation from our PEP formulations, which allows to represent a larger diversity of situations.

A. Distributed (sub)gradient descent (DGD)

We consider K iterations of DGD described by (2) and (3), with constant step-size α, in order to solve problem (1), i.e., minimizing $f(x) = \frac{1}{N}\sum_{i=1}^N f_i(x)$, with x^* as minimizer of f. There are different studies on DGD; e.g., [29] shows that its iterates converge to a neighborhood of the optimal solution x^* when the step-size is constant. In the sequel, we take as baseline the results of a recent survey [1] providing a theoretical bound for the functional error at the average of all the iterates, valid when subgradients are bounded.

Theorem 4 (Performance of DGD [1, Theorem 8]):
Let $f_i, \dots, f_N \in \mathcal{F}_R$, i.e. convex local functions with subgradients bounded by R. Let x^0 be an identical starting point for all agents such that $\|x^0 - x^*\|^2 \leq D^2$. And let $W \in \mathcal{W}(-\lambda, \lambda)$ for some $\lambda \in [0, 1)$, i.e. W is a symmetric and generalized doubly stochastic matrix with eigenvalues $\lambda_2, \dots, \lambda_N \in [-\lambda, \lambda]$.
If we run DGD for K steps with a constant step-size $\alpha = \frac{1}{\sqrt{K}}$, then there holds[1]

$$f(x_{\text{av}}) - f(x^*) \leq \frac{D^2 + R^2}{2\sqrt{K}} + \frac{2R^2}{\sqrt{K}(1-\lambda)}, \qquad (18)$$

where $x_{\text{av}} = \frac{1}{N(K+1)}\sum_{i=1}^N \sum_{k=0}^K x_i^k$ is the average over all the iterations and all the agents.
Theorem 4 was stated in [1] for doubly stochastic matrix W but the proof never uses the non-negativity assumption and therefore it also holds for generalized doubly stochastic matrices. For comparison purposes, we will analyze the performance of DGD using our PEP formulations in exactly the same settings as Theorem 4. Our first PEP formulation searches for solutions to the following maximization problem:

$$\begin{aligned}
\max_{\substack{x^*, f_i, x_i^0, y_i^0 \dots x_i^K, y_i^K \\ \text{for } i=1,\dots,N}} \ & \frac{1}{N}\sum_{i=1}^N \left(f_i(x_{\text{av}}) - f_i(x^*)\right) && \text{(DGD}(W)\text{-PEP)} \\
\text{s.t. } & f_i \in \mathcal{F}_R && \forall i \\
& x^* = \operatorname*{argmin}_x \frac{1}{N}\sum_{i=1}^N f_i(x), \\
& \|x_i^0 - x^*\|^2 \leq D^2 \text{ and } \quad x_i^0 = x_j^0 && \forall i, j \\
& y_i^k = \sum_{j=1}^N w_{ij} x_j^k, && \forall i, k \quad (19) \\
& x_i^{k+1} = y_i^k - \alpha^k \nabla f_i(x_i^k), && \forall i, k
\end{aligned}$$

The objective function and the constraints of (DGD(W)-PEP) are all linearly Gram-representable and the problem can then be formulated as an SDP, according to Proposition 1. This is referred to as the *exact formulation* because it finds the exact worst-case performance of the algorithm for a specific given matrix W. The second formulation relaxes the consensus constraints imposed by equation (19) and replaces them with the constraints from Theorem 2, with $-\lambda^- = \lambda^+ = \lambda$. Those are LMI Gram-representable (see Theorem 2) and can then be used in the SDP formulation of PEP, according to Proposition 1. This formulation is referred to as the *spectral formulation* and provides *spectral worst-cases*, i.e., upper bounds on the worst-case performances of the algorithm, valid for any matrix $W \in \mathcal{W}(-\lambda, \lambda)$. These spectral bounds can thus be compared with the bound from Theorem 4.

In our experiments, we focus on the situation where $D = 1$ and $R = 1$, but the results obtained can be scaled up to general values using changes of variables, see Appendix I.

a) Impact of the number of agents N: In Fig. 1, we observe that the results of the spectral formulation are *independent* of the number of agents $N \geq 2$ in the problem. This is shown for K = 5 iterations and different spectral ranges. This observation has been confirmed for other values of K (10, 15, and 20). The theoretical performance bound from Theorem 4 is also independent of N. Therefore, in the sequel, we analyze the spectral formulation for $N = 3$, which keeps the computational complexity low while keeping some non-trivial network matrices.

b) Comparison with Theorem 4: We compare the spectral bound with the theoretical bound from Theorem 4 for different ranges of eigenvalues $[-\lambda, \lambda]$ for the network matrix. The value of $1 - \lambda$ corresponds to the algebraic connectivity of the network. Fig. 2 shows the evolution of both bounds with λ for $K = 10$ iterations of DGD with $N = 3$ agents. We observe that the spectral worst-case performance bound (in blue) largely improves on the theoretical one (in red), especially when λ approaches 1, in which case the theoretical bound grows unbounded.

The improvements of the bounds when λ is close to 1 is particularly relevant since large values for λ are frequent

[1]Note the factor 2 in the second term of the bound (18) was missing in [1] but its presence was confirmed by the authors of [1].

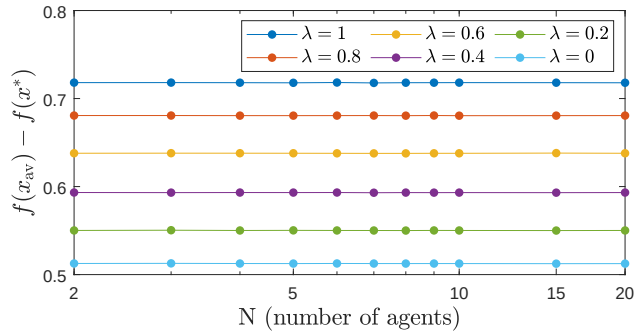


Fig. 1: Independence of N for the spectral worst-case performance of 5 iterations of DGD in the setting of Theorem 4.

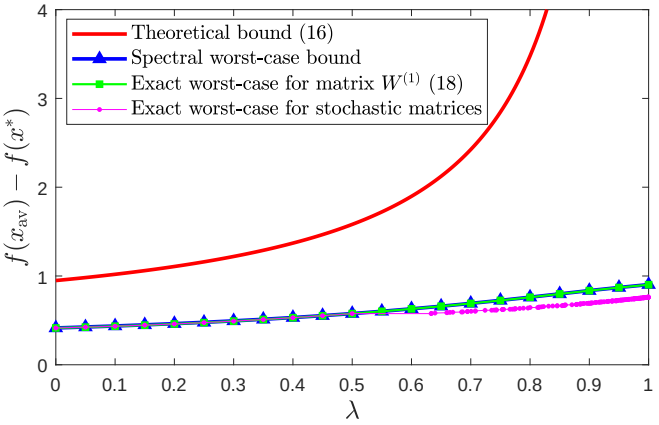


Fig. 2: Evolution with λ of the worst-case performance of $K = 10$ iterations of DGD in the setting of Theorem 4 with $N = 3$ agents. The plot shows (i) the theoretical bound from equation (18) (in red), largely above (ii) the spectral worst-case performance (in blue), (iii) the exact worst-case performance for the symmetric generalized doubly stochastic matrix $W^{(1)}$ from equation (20) (in green) and (iv) the exact worst-case performance for symmetric doubly stochastic matrices found based on an exhaustive exploration of such matrices used in the exact PEP formulation (in pink). This indicates the tightness of the spectral formulation of PEP for DGD with symmetric generalized doubly stochastic matrices, within numerical errors.

for averaging matrices of large networks of agents [30]. For example, for a 5 by 5 grid of agents with Metropolis weights [1], the range of eigenvalues of the resulting averaging matrix is $[-0.92, 0.92]$. In that case, after $K = 10$ iterations, our spectral bound guarantees that the performance measure is below 0.85, compared to 8.2 for the theoretical bound from Theorem 4. This accuracy of 0.85 would only be guaranteed using Theorem 4 with $K = 936$.

c) Worst averaging matrix and tightness analysis: When considering the spectral formulation with a symmetric spectral range $-\lambda^- = \lambda^+ = \lambda$, we observe that the worst averaging matrices are matrices of the following form

$$W^{(1)} = J - \lambda(I - J), \tag{20}$$

where $J = \frac{\mathbf{1}\mathbf{1}^\mathbf{T}}{N}$, i.e. it has all entries equal to $\frac{1}{N}$, and I is the identity matrix. Matrix $W^{(1)}$ is symmetric and generalized doubly stochastic, leading 1 to be one of its eigenvalues. All its other eigenvalues are equal to $-\lambda$. Matrix $W^{(1)}$ always produces a remainder $\|\tilde{Y} - W^{(1)}\tilde{X}\|_F$ close to zero for DGD, but it may not be the only one. The bound obtained using the exact PEP formulation with this specific matrix $W^{(1)}$ for $K = 10$ is plotted in green in Fig. 2 and *exactly* matches the spectral bound in blue, within numerical errors. This means that the spectral formulation provides a *tight performance bound* for DGD with symmetric generalized doubly stochastic matrices, even though it is a relaxation,. This observation has been confirmed for other values of K (5, 15, and 20).

d) Doubly stochastic versus generalized doubly stochastic: Since every doubly stochastic matrix is also generalized doubly stochastic, the spectral bound also provides an upper bound on the performance of DGD with symmetric doubly stochastic matrices. This bound remains tight for $\lambda \leq \frac{1}{N-1}$ because the worst-case matrix $W^{(1)}$ (20) we have obtained is non-negative and is therefore doubly stochastic. For $\lambda > \frac{1}{N-1}$, this is no longer the case and the analysis is performed by empirically looking for symmetric stochastic averaging matrices leading to the worst performance. In Fig. 2, for $N = 3$ and $\lambda > 0.5$, we have generated more than 6000 random symmetric doubly stochastic 3 by 3 matrices. We have analyzed their associated DGD performance using the exact PEP formulation and have only kept those leading to the worst performances. The resulting pink curve deviates no more than 20% below the spectral bound. In that case, the spectral bound is thus no longer tight for DGD with doubly stochastic matrices but remains very relevant. This observation has been confirmed for other values of K and N ($N = 3, 5, 7$, and $K = 10, 15$).

e) Evolution with the total number of iterations K: Fig. 3 shows the evolution of the spectral worst-case performance for DGD multiplied by $\sqrt{K}$, for different values of λ. Except when $\lambda = 1$, all lines tend to a constant value, meaning that the spectral bound behaves in $\mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$, as the theoretical bound (18), but with a much smaller multiplicative constant. When $\lambda = 1$, the line grows linearly and never reaches a constant value. In that case, the worst averaging matrices lead to counterproductive interactions, preventing DGD from working in the worst case.

f) Tuning the step-size α: The PEP methodology allows us to easily tune the parameters of a method. For example, Fig. 4 shows the evolution of the spectral worst-case performance of DGD with the constant step-size it uses, in the setting of Theorem 4 with $N = 3$, $K = 10$ and $\lambda = 0.8$. In that case, we observe that the value $\alpha = \frac{1}{\sqrt{K}}$ used in Theorem 4 for deriving the theoretical performance bound is not the best possible choice for α and should be divided by two to improve the performance guarantees by 30%. The optimal value for α, regarding our spectral bound, is the

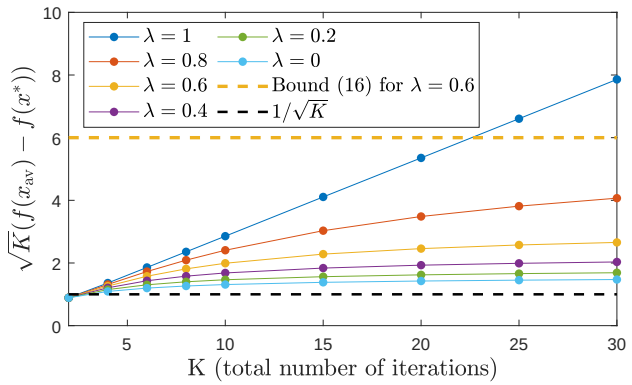


Fig. 3: Evolution with K of the *normalized* spectral worst-case performance of K iterations of DGD in the setting of Theorem 4 with $N = 3$. The shown spectral worst-cases are normalized by $\frac{1}{\sqrt{K}}$ to show that they evolve at this rate.

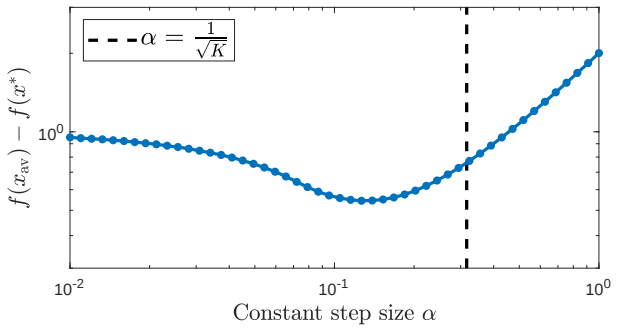


Fig. 4: Evolution with α of the spectral worst-case performance of $K = 10$ iterations of DGD in the setting of Theorem 4 with $N = 3$ agents and $\lambda = 0.8$ (except for α).

one that provides the best worst-case guarantee, whatever the averaging matrix from $\mathcal{W}(-\lambda, \lambda)$ is used.

The impact of the step-size on the other experiments and observations can be studied by setting $\alpha = \frac{h}{\sqrt{K}}$, for some $h > 0$. We focused on $h = 1$ for comparison with the theoretical bound from Theorem 4. Nevertheless, all our other observations have been confirmed[2] for $h = 0.1, 0.5, 2, 10$.

B. DIGing

The DIGing algorithm [4], described in Algorithm 1, combines DGD with a *gradient tracking* technique. Each agent i holds an estimation s_i of the average of all the local gradients and uses it in its update instead of its own local gradient. The DIGing algorithm allows using a different network matrix W^k at each iteration k.

The linear convergence of DIGing has been established in [4, Theorem 3.14] provided that the local functions f_i are L-smooth and μ-strongly convex (with $L \geq \mu > 0$); the network matrices are symmetric, doubly stochastic, and have their second largest eigenvalue λ below 1 (in absolute value);

[2]For too small step-size such as $h \leq 0.1$, the worst averaging matrix observed is no more $W^{(1)}$.

Algorithm 1 DIGing

Choose step-size $\alpha > 0$ and pick any $x_i^0 \in \mathbb{R}^d$;
Initialize $s_i^0 = \nabla f_i(x_i^0)$ for all $i = 1, \dots, N$;
for $k = 0, 1, \dots$ **do**
  for $i = 1, \dots, N$ **do**
    $x_i^{k+1} = \sum_{j=1}^N w_{ij}^k x_j^k - \alpha s_i^k$;
    $s_i^{k+1} = \sum_{j=1}^N w_{ij}^k s_j^k + \nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)$;
  end for
end for

and the step-size is within the interval

$$\alpha \in \left(0, \frac{(1-\lambda)^2}{2L\left(1 + 4\sqrt{N}\sqrt{L/\mu}\right)}\right].$$

The largest accepted step-size decreases thus as $\mathcal{O}(\frac{1}{\sqrt{N}})$ and has also a dependence in $\mathcal{O}(\frac{1}{L\sqrt{L/\mu}})$ which is less favorable than the usual $\mathcal{O}(\frac{1}{L})$ in optimization, leading thus often to very small values for accepted α. The spectral condition on the network matrices ($|\lambda| < 1$) guarantees the network connectivity at each iteration. Actually, [4] imposes a weaker spectral condition which only requires that the union of all the networks over B steps is connected. In this section, we consider the case $B = 1$. Under all these conditions, [4, Theorem 3.14] guarantees the following R-linear[3] convergence:

$$\sqrt{\sum_{i=1}^N \|x_i^K - x^*\|^2} \leq C\rho_{\rm theo}^K, \quad \text{for any } K \in \mathbb{N}, \tag{21}$$

where C is a positive constant and $\rho_{\rm theo} \in (0, 1)$ is the convergence rate depending on N, λ, L and μ (see [4, Theorem 3.14] for details about its expression).

This section analyzes the worst-case performance of DIGing via the exact and spectral formulations, in the same settings as [4, Theorem 3.14] to get a fair comparison. Therefore, we consider the set of L-smooth and μ-strongly convex functions for local functions. As performance criterion, we consider the same as in (21) but squared and scaled by N:

$$\mathcal{P}^K = \frac{1}{N}\sum_{i=1}^N \|x_i^K - x^*\|^2. \tag{22}$$

The corresponding theoretical convergence rate for this criterion is given by $\rho_{\rm theo}^2$. We also consider initial conditions

[3] We recall the definition of the R-linear convergence and its differences with the Q-linear convergence, based on the definitions provided in [4]. Suppose that a sequence $\{x^k\}$ converges to x^* in some norm $\|\cdot\|$. We say that the convergence is: (i) R-linear if there exists $\rho \in (0, 1)$ and some positive constant C such that $\|x^k - x^*\| \leq C\rho^k$ for all k; (ii) Q-linear if there exists $\rho \in (0, 1)$ such that $\frac{\|x^{k+1} - x^*\|}{\|x^k - x^*\|} \leq \rho$ for all k. Both convergences are geometric but the Q-linear convergence is stronger since it implies monotonic decrease of $\|x^k - x^*\|$, while R-linear convergence does not. By definition, Q-linear convergence implies R-linear convergence with the same rate but the inverse implication does not hold in general.

similar to those used implicitly in [4, Theorem 3.14]:

$$\frac{1}{N}\sum_{i=1}^{N}\|x_i^0 - x^*\|^2 \leq D^2, \tag{23}$$

$$\frac{1}{N}\sum_{i=1}^{N}\|s_i^0 - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j^0)\|^2 \leq E^2. \tag{24}$$

Condition (23) bounds the initial performance criterion $\mathcal{P}(x^0)$ which measures the average error of the initial iterates x_i^0. Condition (24) bounds the average error made by the agents on the initial average gradient estimates s_i^0.
For the spectral formulation, to have the same setting as [4, Theorem 3.14], we consider time-varying averaging matrices that are symmetric, generalized doubly stochastic, and with a symmetric range of eigenvalues $[-\lambda, \lambda]$, i.e. in $\mathcal{W}(-\lambda, \lambda)$. In a second time, we will also consider constant network matrices.

The problem depends on 6 parameters: L, μ, D, E, λ, α. We fix $L = 1$ and $D = 1$, as the results can then be scaled up to general values using appropriate changes of variables. The value of E is arbitrarily fixed to $E = 1$, but all the observations have been confirmed for other values of E ($E = 0.1, 10$). Different values of the step-size α will be analyzed to understand its impact on the worst-case performance of the DIGing algorithm. We show the results for representative values of μ and λ ($\mu = 0.1$ and $\lambda = 0.9$).

a) Impact of the number of agents N: As it was the case for DGD (see Section IV-A), we have observed that the spectral worst-case of DIGing is independent of the number of agents N (for $N \geq 2$). This differs from the theoretical analysis of DIGing from [4, Theorem 3.14] for which the range of accepted step-sizes, as well as the convergence rate, depend on N. It appears that the worst-case performance of DIGing does not get worst when N increases, or can at least be bounded uniformly for all values of N, for example with our spectral bound. Such uniform bound will allow us to better choose the step-size α of DIGing, identically for all N. For the subsequent analysis of the results of our spectral PEP formulation for DIGing, we fix $N = 2$ for keeping a low computational complexity. This also corresponds to the most favorable situation for the theoretical results [4, Theorem 3.14], which get worse as N increases.

b) Comparison between the spectral and theoretical bounds: We compare the spectral bounds obtained with PEP with the corresponding guarantees obtained using ρ_{theo}^2, i.e. the square of the theoretical convergence rate bound [4, Theorem 3.14]. Fig. 5 shows the evolution of both guarantees (spectral and theoretical) with the total number of iterations K of the algorithm, for different values of the step-size α and for $N = 2$, $\lambda = 0.9$ and $\mu = 0.1$. The spectral bounds are always smaller than the corresponding theoretical ones.

For the three values of α, we observe a linear decrease of the spectral worst-cases, which strongly suggests a linear convergence rate. The observed rates are listed in TABLE I below and can be compared with the theoretical convergence rate ρ_{theo}^2, which are all larger. The step-size $\alpha = 2.6 \times 10^{-4}$ is the one that optimizes the theoretical convergence guarantee ρ_{theo} from [4, Theorem 3.14] for $N = 2$. For some step-sizes, such as $\alpha = 10^{-3}$, convergence is not guaranteed by [4, Theorem 3.14], even when $N = 2$, while our observations suggest that it does occur. And this get worse when N becomes larger since it would require the theoretical step-sizes to be smaller, i.e. $\alpha_{\max} \approx \frac{4\times10^{-4}}{\sqrt{N}}$. Therefore, the spectral bound for DIGing can help to greatly improve the choice of the step-size.

The same observations can be done for other values of μ and λ. In particular, other values of μ leads to the same graphs, with a different scale for the vertical axis. We test it for $\mu = 0.01$ and $\mu = 0.001$.

step-size α	$1 -$ observed rate	$1 -$ theoretical rate
10^{-4}	2×10^{-5}	7×10^{-6}
2.6×10^{-4}	5×10^{-5}	2×10^{-5}
10^{-3}	2×10^{-4}	/

TABLE I: Theoretical [4] and observed spectral rates for $N = 2$, $\lambda = 0.9$, $\mu = 0.1$ and for different step-sizes α.

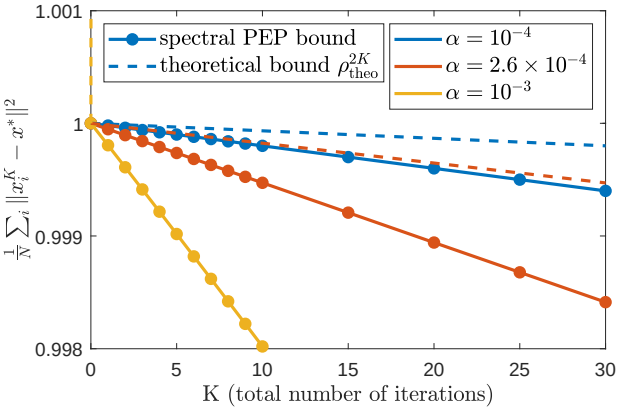


Fig. 5: Evolution with K of the spectral worst-case of K iterations of DIGing with $N = 2$, $\lambda = 0.9$, $\mu = 0.1$, and different values of step-size α. The corresponding theoretical rates from [4, Theorem 3.14] are also shown in comparison. Logarithmic y-axis.

c) Convergence rate analysis with PEP: Fig. 5 strongly suggests a linear convergence rate but the performance guarantees only hold for the values of K tested and we cannot extrapolate them with certainty, e.g. the performance could explode after a larger number of iterations. We now show how to use our PEP formulations to obtain a *guaranteed convergence rate* valid for any number of iterations. The idea is to use the same metric as an initial condition and performance measure to be able to consider the problem over only one general DIGing iteration. We also need to ensure that the DIGing update preserved the assumptions on initial conditions. The metric we use is the weighted combination of the two error measures (23) and (24) previously used separately in the initialization:

$$\mathcal{P}_\gamma^K = \frac{1}{N}\sum_{i=1}^{N}\|x_i^K - x^*\|^2 + \frac{\gamma}{N}\sum_{i=1}^{N}\|s_i^K - \frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j^K)\|^2,$$

where γ is a positive weighting coefficient.

Proposition 5 (Convergence rate of DIGing with PEP): Consider the spectral PEP formulation of DIGing with $\mathcal{P}_\gamma^1$ as performance criterion and with the following initialization: pick any $x_i^0, s_i^0 \in \mathbb{R}^d$ such that

$$\sum_{i=1}^N s_i^0 = \sum_{i=1}^N \nabla f_i(x_i^0) \qquad \text{and} \qquad \mathcal{P}_\gamma^0 = 1.$$

Let θ_γ be the optimal value of this PEP, then

$$\mathcal{P}^k \leq \mathcal{P}_\gamma^k \leq \theta_\gamma^k \mathcal{P}_\gamma^0 \qquad \text{for any } k,\ \gamma \geq 0.$$

The convergence is Q-linear[4] for $\mathcal{P}_\gamma^k$ and R-linear for $\mathcal{P}^k$, both with convergence rate θ_γ depending on coefficient γ.

Proof: One can verify that the following changes of variables, using a coefficient $M \geq 0$,

$$\tilde{x}_i = \sqrt{M} x_i, \quad \tilde{s}_i = \sqrt{M} s_i \quad \text{and} \quad \tilde{f}_i(\tilde{x}_i) = M f_i(x_i),$$

do not affect the behavior of DIGing and scale both $\mathcal{P}_\gamma^0$ and $\mathcal{P}_\gamma^1$ by a factor M:

$$\tilde{\mathcal{P}}_\gamma^0 = M\mathcal{P}_\gamma^0, \qquad \tilde{\mathcal{P}}_\gamma^1 = M\mathcal{P}_\gamma^1.$$

Since $M = \tilde{\mathcal{P}}_\gamma^0$ and θ_γ is the optimal value of $\mathcal{P}_\gamma^1$ (for $\mathcal{P}_\gamma^0 = 1$), we have that

$$\tilde{\mathcal{P}}_\gamma^1 \leq \theta_\gamma \tilde{\mathcal{P}}_\gamma^0. \tag{25}$$

Equation (25) holds for any value of $\tilde{\mathcal{P}}_\gamma^0 \geq 0$ (e.g. for $\mathcal{P}_\gamma^k$). Moreover, the iterations of DIGing are independent of k, and thus, inequality (25) is valid for any iteration k

$$\mathcal{P}_\gamma^{k+1} \leq \theta_\gamma \mathcal{P}_\gamma^k,$$

provided that iterates x_i^k, s_i^k also satisfy the initial condition

$$\sum_{i=1}^N s_i^k = \sum_{i=1}^N \nabla f_i(x_i^k), \qquad \text{for any } k. \tag{26}$$

This condition (26) holds by assumption for $k = 0$ and is preserved by a DIGing update with a stochastic matrix W (see Algorithm 1), as

$$\begin{aligned}\sum_{i=1}^N s_i^{k+1} &= \sum_{i=1}^N \sum_{j=1}^N w_{ij}^k s_j^k + \sum_{i=1}^N \nabla f_i(x_i^{k+1}) - \sum_{i=1}^N \nabla f_i(x_i^k) \\ &= \sum_{j=1}^N s_j^k + \sum_{i=1}^N \nabla f_i(x_i^{k+1}) - \sum_{i=1}^N \nabla f_i(x_i^k) \\ &= \sum_{i=1}^N \nabla f_i(x_i^{k+1}),\end{aligned}$$

Finally, by definition of $\mathcal{P}^k$ (see (22)) and $\mathcal{P}_\gamma^k$, we have well that $\mathcal{P}^k \leq \mathcal{P}_\gamma^k$ for any k, $\gamma \geq 0$. ■

Using Proposition 5, we can obtain guaranteed convergence rates for DIGing, which depend on the weighting coefficient γ, for the metric $\mathcal{P}_\gamma^K$. These convergence rates are also valid for $\mathcal{P}^K$, for all $\gamma \geq 0$, and can thus be compared with the observed rates from Fig. 5 and the theoretical convergence rates (21) ([4, Theorem 3.14]). An exploration of the different values for the weighting coefficient $\gamma \geq 0$ suggests that the best rates are obtained for $\gamma = \frac{\alpha}{L}$ but other rates are also valid. With $\gamma = \frac{\alpha}{L}$, we recover exactly the same rates as those observed in Fig. 5, but they are guaranteed with certainty for any number of iterations K. The PEP problem from Proposition 5 has a small size since it only considers one iteration, and is thus rapidly solved. The size of the problem still increases with the number of agents N but once again, we observe that the results are independent of N.

[4]See definition of Q-linear and R-linear convergence in footnote 3

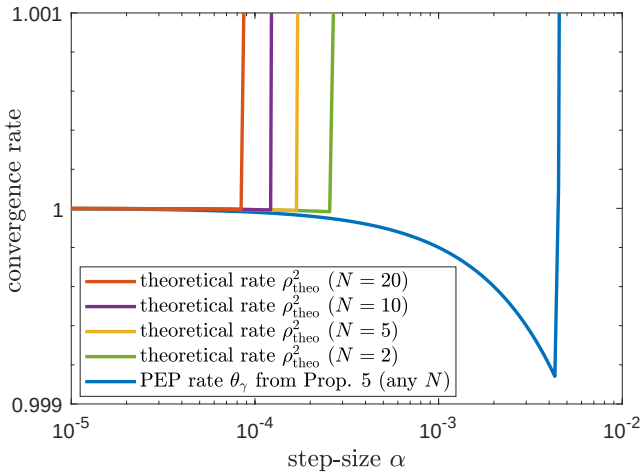


Fig. 6: Convergence rate evolution with the step-size α for DIGing with $\lambda = 0.9$ and $\mu = 0.1$. Theoretical rates from [4] are shown for different values of N. PEP rates θ_γ, obtained with Proposition 5 for $\gamma = \frac{\alpha}{L}$, are lower and allow for larger step-sizes. These rates are computed with $N = 2$ but the results are identical for any value of N.

The approach above can be applied to other algorithms provided that their updates are independent of each other. It presents some parallels with the approach used in the automatic analysis with IQC [20]. Both approaches analyze the decrease of a particular function over only one iteration. We design the decreasing criterion $\mathcal{P}_\gamma$ by hand and have optimized the value of γ to find the smallest rate. The IQC approach allows to easily optimize the rate over a wider class of Lyapunov functions and it may therefore give smaller rates. On the other hand, our PEP approach provides the worst functions and communication networks resulting from the worst-case solutions. It also allows comparing what happens over one iteration and over several, and with network matrices that are variable or constant in time.

d) Impact of the step-size α: Fig. 6 compares the spectral rates, obtained with the spectral PEP formulation from Proposition 5 with the theoretical ones from [4, Theorem 3.14] for a wide range of values for the step-size α. The spectral rates are identical for any value of N and present a first regime where they decrease as α increases until a certain threshold step-size α_t. This decrease is numerically close to $1 - 2\mu\alpha$. After α_t, we observe a sharp increase in the spectral rates. Since the theoretical results [4] depends on the value of N, Fig. 6 shows different curves, corresponding to theoretical bounds on the convergence rates for different numbers of agents N. All these curves also present two regimes, however, the decrease of the first regime is slower

and the sharp increase takes place after a much lower threshold step-size. Moreover, both the sharp increase and the threshold step-size worsen as N increases.
Therefore, the spectral rates obtained allow for significant improvements in the tuning of the step-size α by choosing a larger value ($\approx \alpha_t$), which is independent of N. This leads to better convergence guarantees and therefore to better use of DIGing. For example, in the setting of Fig. 6, when $N = 20$, the theoretical bound requires a step-size below 10^{-4}, while the optimal step-size according to our spectral rate is around 4×10^{-3}. This choice of the step-size allows improving the convergence guarantee by at least 2 orders of magnitude.
We observed the same two regimes for all the other values of μ and λ we tested. However, the smaller the value of λ is, the larger the gap between the values of the theoretical and spectral convergence rate at the threshold step-size.

e) Impact of time-varying averaging matrices: In the spectral formulation, we can choose to link different iterations together and to analyze the worst-case when we use the same constant averaging matrix W at each iteration. We can also choose to consider each consensus step independently with potentially time-varying averaging matrices. For DIGing, we have chosen the second option to allow time-varying averaging matrices and be in the same condition as [4]. However, we observe that both choices lead to the same worst-case values, even though the solution achieving these worst-cases may be different. One worst-case solution obtained with a constant matrix is therefore also a worst-case solution of the situation with time-varying matrices. We can thus analyze what is the worst constant matrix for DIGing and it will also be valid for time-varying settings.

f) Worst averaging matrix and tightness analysis: When the step-size α is optimized, i.e., it is the threshold value on Fig. 6, we observe that the worst matrix for DIGing is the same as for DGD, and is thus given by $W^{(1)}$ in (20). This matrix is only determined by the value of λ and N and the remainder it produces $\|\tilde{Y} - W^{(1)}\tilde{X}\|$ is always close to zero in that case. This matrix $W^{(1)}$ is symmetric and generalized doubly stochastic. The same worst matrix is also recovered for larger step-size α. The bounds obtained using the exact PEP formulation with this specific matrix $W^{(1)}$ *exactly* match the corresponding spectral bounds, within numerical errors. This means that the spectral formulation, even though it is a relaxation, provides again a *tight performance bound* for DIGing with symmetric generalized doubly stochastic matrices and sufficiently large step-size.

In summary, our spectral PEP formulation provides numerically tight convergence rates for DIGing that are independent of the number of agents N, and allows for better tuning of the constant step-size α, leading to more efficient use of the DIGing algorithm.

C. Accelerated Distributed Nesterov Gradient Descent

As third use case, we analyze the accelerated distributed Nesterov gradient descent (Acc-DNGD) algorithm proposed in [8]. We focus on the version designed for convex (not necessarily strongly convex) and L-smooth functions, described in Algorithm 2. It achieves one of the best proved convergence rate in such setting, $\mathcal{O}\big(\frac{1}{K^{1.4-\epsilon}}\big)$ for any $\epsilon \in (0, 1.4)$, but there remains several open questions on choice of parameters and actual performance. We show how our technique allows shedding light on these questions. We use the notations of [8], which are slightly different from the rest of this paper. Here, η_k denotes the diminishing step-size and α_k denotes a weighting factor. Each agent i keeps variables x_i, v_i, y_i, and s_i. The variables s_i are local gradient tracking variables allowing each agent to estimate the average gradient $\frac{1}{N}\sum_{i=1}^N \nabla f_i(y_i)$. The step-size η_k is diminishing as

$$\eta_k = \frac{\eta}{(k+k_0)^\beta}, \tag{27}$$

where $\eta \in \big(0, \frac{1}{L}\big)$, $\beta \in (0, 2)$ and $k_0 \geq 1$. The sequence of α_k starts with $\alpha_0 = \sqrt{\eta_0 L} \in (0, 1)$ and the next element of the sequence is each time computed as the unique solution in $(0, 1)$ of

$$\alpha_{k+1}^2 = \frac{\eta_{k+1}}{\eta_k}(1 - \alpha_{k+1})\alpha_k^2.$$

The convergence result [8, Theorem 4] guarantees that the algorithm achieves an average functional error bounded as

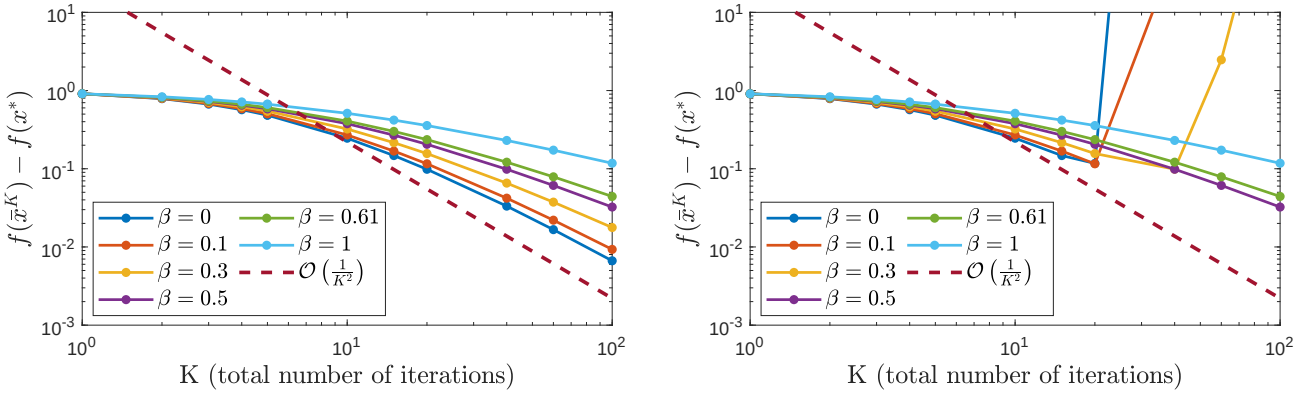
$$f(\overline{x}^k) - f(x^*) \leq \mathcal{O}(\frac{1}{k^{2-\beta}}) \qquad \text{for } \beta \in (0.6, 2).$$

Recall that $f(x) = \frac{1}{N}\sum_{i=1}^N f_i(x)$, $\overline{x}^k = \frac{1}{N}\sum_{i=1}^N x_i^k$ and x^* is a minimizer of f. This convergence guarantee for Acc-DNGD only holds under specific conditions concerning the values of η and k_0 [8, Theorem 4]. These assumptions seem strong since they impose, in particular, that η tends to 0 and k_0 tends to ∞ both when λ tends to 1 (disconnected graph) and to 0 (fully connected graph). The authors of [8] conjecture that

(i) Exact values of parameters η, k_0 do not actually matter and we can choose values that are not satisfying assumptions from [8, Theorem 4], and still obtain a rate of $\mathcal{O}(\frac{1}{k^{2-\beta}})$. For example, authors of [8] used $\eta = \frac{1}{2L}$, $k_0 = 1$ in their numerical experiments with $\beta = 0.61$.
(ii) Choosing $\beta \in [0, 0.6]$ leads to the same rate $\mathcal{O}(\frac{1}{k^{2-\beta}})$. The case $\beta = 0$ uses constant step-size η and is important because it would lead to a rate $\mathcal{O}(\frac{1}{k^2})$, i.e. the best possible rate in centralized framework for similar settings [31, Theorem 2.1.7].

Algorithm 2 Acc-DNGD

Initialize $x_i^0 = v_i^0 = y_i^0 = 0$ and $s_i^0 = \nabla f(0)$ for all i;
for $k = 0, 1, \ldots$ **do**
  for $i = 1, \ldots, N$ **do**
    $x_i^{k+1} = \sum_{j=1}^N w_{ij} y_j^k - \eta_k s_i^k$;
    $v_i^{k+1} = \sum_{j=1}^N w_{ij} v_j^k - \frac{\eta_k}{\alpha_k} s_i^k$;
    $y_i^{k+1} = \alpha_{k+1} x_i^{k+1} + (1 - \alpha_{k+1}) v_i^{k+1}$;
    $s_i^{k+1} = \sum_{j=1}^N w_{ij} s_j^k + \nabla f_i(y_i^{k+1}) - \nabla f_i(y_i^k)$;
  end for
end for



(a) $\eta = 0.05$, $\lambda = 0$

(b) $\eta = 0.05$, $\lambda = 0.75$

(c) $\eta = 0.5$, $\lambda = 0$

(d) $\eta = 0.5$, $\lambda = 0.75$

Fig. 7: Evolution with K of the worst-case average functional error $f(\bar{x}^K) - f(x^*)$ for Acc-DNGD, obtained with the spectral PEP formulation, with $N = 2$, $L = 1$ and $k_0 = 1$. Different values of β, η and λ are shown. Dashed lines show curves evolving in $\mathcal{O}(\frac{1}{K^2})$, which corresponds to the rate conjectured in [8] when $\beta = 0$. This asymptotic rate is reached for $\beta = 0$ only when η is sufficiently small.

In this section, we use our spectral PEP formulation to analyze the Acc-DNGD algorithm, which allows having a better idea about the impact of the parameters on its performance and nuance the conjectures (i) and (ii) of [8]. Fig. 7 shows the evolution of the worst-case average functional error $f(\bar{x}^K) - f(x^*)$ with the total number of iterations K. We consider a symmetric range of eigenvalues $-\lambda^- = \lambda^+ = \lambda$, and thus we obtain spectral bounds valid over the entire class of matrices $\mathcal{W}(-\lambda, \lambda)$. The results are shown for different values of β, η and λ, while L, N and k_0 are fixed. We aim at testing the conjectures (i) and (ii). To test (ii), about the validity range of β, conjectured to extend below 0.6, we choose $\beta = \{0, 0.1, 0.3, 0.5\}$. To compare with proven valid values of β and test conjecture (i), we choose $\beta = \{0.61, 1\}$. We choose $\lambda = \{0, 0.75\}$ and $\eta = \{0.05, 0.5\}$ to have two representative values for each parameter (small and large). We fix $L = 1$ because the results for other values of L can be recovered by scaling. We fix $N = 2$ because the value of N does not impact the worst-case value obtained with the spectral formulation, as for the two previous algorithms. We fix $k_0 = 1$ for simplicity and because it does not affect the long term evolution of the performance. In particular, the value of k_0 has no impact when $\beta = 0$, see (27).

We first observe that the value η influences the performance of the algorithm. Indeed, when η becomes larger (e.g., from Fig. 7a, 7b to Fig. 7c, 7d), the worst-case functional error decreases faster in a first phase but it sometimes explodes afterwards, preventing the algorithm to converge. This occurs for example when β is too small (e.g. $\beta = 0$ and $\beta = 0.1$ in Fig. 7c) or when λ is too large (e.g. in Fig. 7d and 7b where $\lambda = 0.75$). In Fig. 7d, this sharp increase even occurs in the experimental setting of [8], i.e. $\eta = \frac{1}{2L} = 0.5$ and $\beta = 0.61$ (green line). These observations contradict conjecture (i) as currently stated.

Secondly, we observe in Fig. 7 that the curves for $\beta = 0.5$ and $\beta = 0.61$ behaves similarly in all the different plot. This suggests that the worst-case values present no phase transition at $\beta = 0.6$, and therefore, supports conjecture (ii) claiming that values of β lower than 0.6 also provide rates $\mathcal{O}(\frac{1}{K^{2-\beta}})$, including the case $\beta = 0$. The curves for $\beta = 0$ appear indeed to approach a decrease in $\mathcal{O}(\frac{1}{K^2})$, as conjectured in [8], but only in Fig. 7a where the values η and λ are sufficiently small. Indeed, the rate $\mathcal{O}(\frac{1}{K^2})$ cannot be observed for larger values of η or λ where PEP problems may become unbounded (see Figures 7b, 7c and 7d). Therefore, the η parameter should be tuned according to the values of λ and the choice of β. Qualitatively, we can see that

it must decrease when λ increases or when β decreases. Further analysis should be performed to tune the value of η in general. Having sufficiently small step-sizes η seems thus to be the key for conjecture (ii) to be true. Since smaller values for β appear to require a smaller step-size, the performance would approach the convergence rate $\mathcal{O}(\frac{1}{K^{2-\beta}})$ more slowly in that case. However, even with well-tuned η, we cannot exclude that the worst-case performance of Acc-DNGD does not explode after a large number of iterations in some cases, as is the case in an early phase for some settings in Fig. 7.

In addition, it is interesting to note that the worst matrix that is recovered from the spectral PEP formulation for Acc-DNGD is again $W^{(1)}$ from (20), as it was the case for DGD and DIGing. For Acc-DNGD also, the bounds obtained using the exact PEP formulation with this specific matrix $W^{(1)}$ *exactly* match the corresponding spectral bounds, within numerical errors. This means that the spectral formulation, even though it is a relaxation, provides a *tight performance bound* for Acc-DNGD with symmetric generalized doubly stochastic matrices.

D. Code and Toolbox

PEP problems have been written and solved using the PESTO Matlab toolbox [21], with Mosek solver, within 200 seconds maximum. For example, for $N = 3$, the time needed for a regular laptop to solve the spectral formulation for DGD (from Section IV-A) is about 3, 12, 48, and 192 seconds respectively for $K = 5, 10, 15, 20$. The PESTO toolbox is available on GITHUB (`https://github.com/PerformanceEstimation/Performance-Estimation-Toolbox`) We have updated PESTO to allow easy and intuitive PEP formulation of gradient-based decentralized optimization methods, and we have added a code example for DGD, DIGing, and Acc-DNGD.

V. CONCLUSION

We have developed a methodology that automatically computes numerical worst-case performance bounds for any decentralized optimization method that combines first-order oracles with average consensus steps. This opens the way for computer-aided analysis of many other decentralized algorithms, which could lead to improvements in their performance guarantees and parameter tuning, and could allow rapid exploration of new algorithms. Moreover, the guarantees computed with our tool appear tight in many cases.

Our methodology is based on the performance estimation problem (PEP) for which we have developed two representations of average consensus steps. Our first formulation provides the exact worst-case performance of the method for a specific network matrix. The second formulation provides upper performance bounds that are valid for an entire spectral class of matrices. This spectral formulation often allows recovering the worst possible network matrix based on the PEP solutions. We demonstrate the use of our automatic performance methodology on three algorithms and we discover, among other things, that DGD worst-case performance is much better than the theoretical guarantee; DIGing has performance independent of the number of agents and accepts much larger step-size than predicted by the theory; Acc-DNGD appears to present a rate $\mathcal{O}(\frac{1}{K^{2-\beta}})$, for $\beta \in (0, 2)$, only if the step-sizes are sufficiently small.

Further developments of our methodology may include a spectral formulation that is independent of the number of agents [32] or that considers other classes of network matrices, e.g. non symmetric or B-connected networks.

ACKNOWLEDGMENT

The authors wish to thank Adrien Taylor for his helpful advice concerning the PESTO toolbox.

APPENDIX I
NOTE ON SCALING OF DGD

In the DGD analysis, we have one constant R for the bound on the subgradients $\|\nabla f_i(x_i^K)\|^2 \leq R^2$ (for all K), and another one D for the bound on the initial distance to optimum $\|x^0 - x^*\|^2 \leq D^2$. In our performance estimation problem, we consider general positive values for these parameters $D > 0$ and $R > 0$ and we parametrize the step-size by $\alpha = \frac{Dh}{R\sqrt{K}}$, for some $h > 0$.

To pass from this general problem to the specific case where $D = 1$ and $R = 1$, we consider the following changes of variables:

$$\tilde{x}_i = \frac{x_i}{D}, \quad \tilde{f}_i(\tilde{x}_i) = \frac{1}{DR} f_i(x_i) \quad \text{and} \quad \tilde{\alpha} = \frac{R\alpha}{D}.$$

These changes of variables do not modify the nature of the problem and allow expressing the worst-case guarantee obtained for $f(x_{\text{av}}) - f(x^*)$ with general values of D, R, and h, denoted $w(D, R, h)$, in terms of the worst-case guarantee obtained for $\tilde{f}(\tilde{x}_{\text{av}}) - \tilde{f}(\tilde{x}^*)$ with $D = R = 1$:

$$w(D, R, h) = DR\ \tilde{w}(1, 1, h). \tag{28}$$

The same kind of scaling can be applied to Theorem 4. The theorem is valid for general values of D and R but is specific to $\alpha = \frac{1}{\sqrt{K}}$, which is equivalent to picking $h = \frac{R}{D}$. After the scaling, we obtain the following bound, valid for $D = 1$, $R = 1$ and any value of $\alpha = \frac{h}{\sqrt{K}}$ with $h > 0$:

$$\tilde{f}(\tilde{x}_{\text{av}}) - \tilde{f}(\tilde{x}^*) \leq \frac{h^{-1} + h}{2\sqrt{K}} + \frac{2h}{\sqrt{K}(1 - \lambda)}. \tag{29}$$

This scaled theoretical bound with $h = 1$ is equivalent to the bound from Theorem 4 with $D = R = 1$, which was the focus of the numerical analysis in Section IV.
This bound (29) can be extended to any value of $D > 0$ and $R > 0$, using the relation from equation (28):

$$f(x_{\text{av}}) - f(x^*) \leq DR\left(\frac{h^{-1} + h}{2\sqrt{K}} + \frac{2h}{\sqrt{K}(1 - \lambda)}\right).$$

REFERENCES

[1] A. Nedić, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 953–976, 2018.

[2] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, pp. 48 – 61, 02 2009.

[3] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, 04 2014.

[4] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, 07 2016.

[5] Z. Li, W. Shi, and M. Yan, "A decentralized proximal-gradient method with network independent step-sizes and separated convergence rates," *IEEE Transactions on Signal Processing*, vol. PP, 04 2017.

[6] C. Xi, R. Xin, and U. A. Khan, "Add-opt: Accelerated distributed directed optimization," *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1329–1339, 2018.

[7] R. Xin and U. A. Khan, "A linear algorithm for optimization over directed graphs with geometric convergence," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 315–320, 2018.

[8] G. Qu and N. Li, "Accelerated distributed nesterov gradient descent," *IEEE Transactions on Automatic Control*, vol. 65, no. 6, pp. 2566–2581, 2020.

[9] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, 2012.

[10] K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulié, "Optimal algorithms for smooth and strongly convex distributed optimization in networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 3027–3036.

[11] K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulié, "Optimal convergence rates for convex distributed optimization in networks," *Journal of Machine Learning Research*, vol. 20, no. 159, pp. 1–31, 2019.

[12] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, 01 2011.

[13] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization," *Signal Processing, IEEE Transactions on*, vol. 62, 07 2013.

[14] Z. Song, L. Shi, S. Pu, and M. Yan, "Optimal gradient tracking for decentralized optimization," 2021.

[15] H. Hendrikx, "Accelerated methods for distributed optimization," Ph.D. dissertation, PSL, 2021, advisors : Francis, BACH and Laurent, MASSOULIE.

[16] A. B. Taylor, J. M. Hendrickx, and F. Glineur, "Smooth strongly convex interpolation and exact worst-case performance of first-order methods," *Mathematical Programming*, vol. 161, 02 2015.

[17] ——, "Exact worst-case performance of first-order methods for composite convex optimization," *SIAM Journal on Optimization*, vol. 27, 12 2015.

[18] S. Colla and J. M. Hendrickx, "Automated worst-case performance analysis of decentralized gradient descent," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 2627–2633.

[19] L. Lessard, B. Recht, and A. Packard, "Analysis and design of optimization algorithms via integral quadratic constraints," *SIAM Journal on Optimization*, vol. 26, 08 2014.

[20] A. Sundararajan, B. V. Scoy, and L. Lessard, "Analysis and design of first-order distributed optimization algorithms over time-varying graphs," *IEEE Transactions on Control of Network Systems*, vol. 7, pp. 1597–1608, 2020.

[21] A. B. Taylor, J. M. Hendrickx, and F. Glineur, "Performance estimation toolbox (PESTO): Automated worst-case analysis of first-order optimization methods," in *IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 1278–1283.

[22] Y. Drori and M. Teboulle, "Performance of first-order methods for smooth convex minimization: A novel approach," *Mathematical Programming*, vol. 145, 06 2012.

[23] A. Taylor, "Convex interpolation and performance estimation of first-order methods for convex optimization," Ph.D. dissertation, UCLouvain, 2017, advisors : François Glineur and Julien M. Hendrickx.

[24] D. Kim and J. Fessler, "Optimized first-order methods for smooth convex minimization," *Mathematical Programming*, vol. 159, 06 2014.

[25] A. B. Taylor and Y. Drori, "An optimal gradient method for smooth strongly convex minimization," *Mathematical Programming*, pp. 1–38, 2022.

[26] A. B. Taylor, J. M. Hendrickx, and F. Glineur, "Exact worst-case convergence rates of the proximal gradient method for composite convex minimization," *Journal of Optimization Theory and Applications*, vol. 178, 08 2018.

[27] B. Goujaud, C. Moucer, F. Glineur, J. Hendrickx, A. Taylor, and A. Dieuleveut, "PEPit: computer-assisted worst-case analyses of first-order optimization methods in Python," *arXiv preprint arXiv:2201.04040*, 2022.

[28] Hanley and C.-K. Li, "Generalized doubly stochastic matrices and linear preservers," *Linear and Multilinear Algebra*, vol. 53, pp. 1–11, 01 2005.

[29] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM Journal on Optimization*, vol. 26, 10 2013.

[30] P. Bhunia, S. Bag, and K. Paul, "Bounds for eigenvalues of the adjacency matrix of a graph," *Journal of Interdisciplinary Mathematics*, vol. 22, 06 2019.

[31] Y. Nesterov, *Lectures on Convex Optimization*, 2nd ed. Springer Publishing Company, Incorporated, 2018.

[32] S. Colla and J. M. Hendrickx, "Automated performance estimation for decentralized optimization via network size independent problems," in *2022 61th IEEE Conference on Decision and Control (CDC)*, 2022.