OJMO

Open Journal of Mathematical Optimization

Sebastien Colla & Julien M. Hendrickx

Exploiting Agent Symmetries for Performance Analysis of Distributed Optimization Methods

Volume 7 (2026), article no. 1 (42 pages)
https://doi.org/10.5802/ojmo.49

Article submitted on March 18, 2024, revised on October 25, 2025,
accepted on January 12, 2026.

CENTRE MERSENNE

Open Journal of Mathematical Optimization is a member of the
Centre Mersenne for Open Scientific Publishing
http://www.centre-mersenne.org/
e-ISSN: 2777-5860

Exploiting Agent Symmetries for Performance Analysis of Distributed Optimization Methods

Sebastien Colla
UCLouvain, ICTEAM institute, Louvain-la-Neuve, Belgium
Julien M. Hendrickx
UCLouvain, ICTEAM institute, Louvain-la-Neuve, Belgium

Abstract

We show that, in many settings, the worst-case performance of a distributed optimization algorithm is independent of the number of agents in the system, and can thus be computed in the fundamental case with just two agents. This result relies on a novel approach that systematically exploits symmetries in worst-case performance computation, framed as Semidefinite Programming (SDP) via the Performance Estimation Problem (PEP) framework. Harnessing agent symmetries in the PEP yields compact problems whose size is independent of the number of agents in the system. When all agents are equivalent in the problem, we establish the explicit conditions under which the resulting worst-case performance is independent of the number of agents and is therefore equivalent to the basic case with two agents. Our compact PEP formulation also allows the consideration of multiple equivalence classes of agents, and its size only depends on the number of equivalence classes. This enables practical and automated performance analysis of distributed algorithms in numerous complex and realistic settings, such as the analysis of the worst agent performance. We leverage this new tool to analyze the performance of the EXTRA algorithm in advanced settings and its scalability with the number of agents, providing a tighter analysis and deeper understanding of the algorithm performance.

Digital Object Identifier 10.5802/ojmo.49

Keywords Distributed optimization, Performance estimation problem, Worst-case analysis.

Acknowledgments S. Colla is a FRIA grantee of the Fonds de la Recherche Scientifique - FNRS. J. M. Hendrickx is supported by the Federation Wallonie-Bruxelles through the "*RevealFlight*" Concerted Research Action (ARC) and by the F.R.S.-FNRS via the research project KORNET and the Incentive Grant for Scientific Research (MIS) "*Learning from Pairwise Comparisons*".

1 Introduction

We consider the distributed optimization problem in which a set of agents $\mathcal{V} = \{1, \dots, n\}$ is connected through a communication network and works together to minimize the average of their local functions $f_i : \mathbb{R}^d \to \mathbb{R}$,

$$\min_{x \in \mathbb{R}^d} \quad f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x), \tag{1}$$

Each agent i performs local computation on its own local function f_i and exchanges local information with its neighbors to update its own guess x_i of the solution. The agents all want to come to an agreement on the minimizer x^* of the global function f. One of the first methods proposed to solve (1) is the distributed (sub)gradient descent (DGD) [21] where agents successively perform an average consensus step (2) and a local gradient step (3):

$$y_i^k = \sum_{j=1}^{n} w_{ij}^k x_j^k, \qquad \text{for } i = 1, \dots, n, \tag{2}$$

$$x_i^{k+1} = y_i^k - \alpha^k \nabla f_i(x_i^k), \qquad \text{for } i = 1, \dots, n, \tag{3}$$

for given step-sizes $\alpha^k > 0$ and matrices of weights $W^k = [w_{ij}^k] \in \mathbb{R}^{n \times n}$, typically assumed symmetric and with rows and columns summing to one. We call such matrices *averaging matrices*.

▶ **Definition 1.** *We say that a matrix $W \in \mathbb{R}^{n\times n}$ is an* averaging matrix *if*

1. $W^T = W$, *(Symmetry)*
2. $W\mathbf{1} = \mathbf{1}$ *and* $\mathbf{1}^T W = \mathbf{1}^T$, *(Averaging Consensus)*

There are many other methods combining gradient sampling and consensus steps. We call $\mathcal{A}_D$ the set of all these decentralized optimization algorithms.

▶ **Definition 2** (Class of distributed optimization methods $\mathcal{A}_D$)**.** *We define $\mathcal{A}_D$ as the set of all the decentralized optimization algorithms built based on the three following types of operations and which may involve an arbitrary number of local variables:*

i. Gradient: *Each agent samples the (sub)gradient of its local function at any of its local variables x_i*

$$g_i = \nabla f_i(x_i) \qquad \text{for } i = 1, \dots, n.$$

ii. Consensus: *All the agents perform a consensus step on any of their local variables x_i*

$$y_i = \sum_{j=1}^{n} w_{ij} x_j, \quad \text{for } i = 1, \dots, n. \tag{4}$$

with weights w_{ij} given by an averaging matrix W. Different consensus steps can possibly use the same averaging matrix.

iii. Linear Combinations: *Each agent declares that a linear combination of its local variables holds, with coefficients known in advance. Depending on the settings, the linear coefficients should be identical for all agents (coordinated) or not (uncoordinated).*

This class of methods includes many algorithms. These three operations even allow implicit (or proximal) updates, e.g. updates where the point at which the gradient is evaluated is not explicitly known:

$$x_i^{k+1} = x_i^k - \alpha^k \nabla f_i(x_i^{k+1}) \qquad \text{for all } i = 1, \dots, n.$$

Among the primal-based algorithms, $\mathcal{A}_D$ includes, for example, DGD [21], DIGing [19], EXTRA [25], NIDS [16], Acc-DNGD [22], OGT [28] and their variations [20, 36, 15, 11]. Among the dual-based algorithms, $\mathcal{A}_D$ includes, for example, the Distributed Dual Dveraging [8], MSDA [23], MSDP [24], APAPC [12], OPTRA [35], APM [14] and others [33]. The classical decentralized version of ADMM [2, 27] does not fit in $\mathcal{A}_D$ because the agents do not explicitly use averaging consensus when interacting, but the weighted decentralized version of ADMM [17] does fit in $\mathcal{A}_D$.

In optimization, the assessment of the performance of a method is generally based on worst-case guarantees. Accurate worst-case guarantees on the performance of decentralized algorithms are crucial for a comprehensive understanding of how their performance is influenced by their parameters and the network topology, which then allows to correctly tune and compare the different algorithms. However, deriving such bounds can often be a challenging task, requiring a proper combination of the impact of the optimization component and of the communication network, which may result in conservative or highly complex bounds. Moreover, theoretical analyses often require specific settings that differ from one algorithm to another, making them difficult to compare.

1.1 Contributions and paper organization

In this work, we propose a simplified and unified way of analyzing the worst-case performance of distributed optimization methods from $\mathcal{A}_D$, based on the Performance Estimation Problem (PEP) framework and the exploitation of agent symmetries in the problem. This allows us, in particular, to identify and characterize situations in which performance is independent of the number of agents n in the network, in which case the performance computation can be reduced to the basic case with two agents.

The Performance Estimation Problem (PEP) formulates the computation of a worst-case performance guarantee as an optimization problem itself, by searching for the iterates and the function leading to the largest error after a given number of iterations of the algorithm. The PEP approach has led to many results in centralized optimization, see e.g. [32, 30], and we have recently proposed a formulation tailored for decentralized optimization that searches for the worst local iterates and local functions for each agent, see [6] (and its preliminary conference version [4]). This formulation considers explicitly each agent in the PEP and so the size of the optimization

problem increases with the number of agents n, and the results are valid only for a given n. We therefore call this formulation the agent-dependent formulation. Section 2 summarizes this formulation and adds some new results and interpretations. To obtain guarantees that are valid for any symmetric and stochastic averaging matrix, with given bounds on the eigenvalues, in [6] we proposed a way of representing a consensus step (4) in PEP via necessary constraints that consensus variables x_i and y_i should satisfy. Subsection 2.2 describes these constraints and analyzes their sufficiency. We first show that no convex description can tightly describe the consensus iterates, and then we characterize the generalized steps that are tightly described by the proposed convex constraints. This helps explain why these necessary conditions enable accurate performance calculations in PEP, as observed in [6].

In [6], we have also observed that for many performance settings, the worst-case guarantees obtained with PEP are independent of the number of agents, while the PEP problems are not. This motivates us to find compact ways of formulating the agent-dependent PEP. We have performed a first attempt in [5], where we built a relaxation of the problem whose size is independent of the number of agents n. In particular, the relaxation does not exploit separability of the objective function $f(x)$ in (1). While the relaxation in [5] gives good worst-case bounds, it remains an open question whether the equivalence with the agent-dependent formulation holds and how to interpret the resulting worst-case solution for the decentralized problem. In this paper, we provide an intuitive and systematic way of exploiting the agent symmetries in PEP to make the problem compact, with a size independent of the number of agents. In Section 3, we define *equivalent* agents in a PEP for decentralized optimization, and we leverage the convexity of the PEP to show that we can restrict it to solutions symmetrized over equivalent agents, without impacting its worst-case value. Section 4 focuses on the case where all the agents are equivalent, meaning none of them play a specific role in the algorithm or the performance evaluation. In that case, we show that the agent-dependent PEP can be written compactly, i.e. with an SDP whose size is independent of n. Moreover, we show that the worst-case value of this compact formulation is also independent of n in many common performance settings of distributed optimization which are scale-invariant. This result is particularly powerful since it allows determining situations where the performance analysis of a distributed algorithm can be reduced to the fundamental case with only two agents. We further leverage agent equivalence to draw general conclusions about the symmetry of worst-case local functions and iterates in decentralized algorithms.

Then, Section 5 generalizes the results from Section 4 to situations where there are several equivalence classes of agents in the PEP, leading to a compact PEP formulation whose size only depends on the number of equivalence classes r, and not directly on the total number of agents n. Indeed, while many performance estimation problems for decentralized optimization have all agents equivalent, there are more complex ones that involve multiple equivalence classes of agents, for example, when different groups of agents use different (uncoordinated) step-sizes, function classes, initial conditions, or even different algorithms. It can also happen that the performance measure focuses on a specific group of agents, e.g. the performance of the worst agent. Efficiently and accurately assessing distributed optimization algorithm performance in such advanced settings would enhance comprehensive analysis and deepen our understanding of their behavior.

We demonstrate this in Section 6, where we analyze the performance of the EXTRA algorithm [25] in advanced settings and its evolution with the number of agents in the problem. We choose EXTRA because it is a well-known decentralized optimization algorithm, one of the first to converge with constant steps, and has served as an inspiration and building block for other algorithms. Our analysis of EXTRA first confirms our theoretical results predicting which performance settings would lead to agent-independent guarantees. It also reveals that the performance of the worst agent scales sublinearly with n and can benefit from an appropriate step-size decrease with n. Inspired by statistical approaches, we go further by analyzing the 80-th percentile of the agent performance, i.e. the error at or below which 80% of the agents fall in the worst-case scenario. This performance measure presents a better dependence on the number of agents n than the performance of the worst agent and quickly reaches a plateau when n increases, which can be validated by solving compact PEP for $n \to \infty$. To the best of our knowledge, such percentile analysis is beyond the reach of current classical analysis techniques. Finally, we analyze the performance of EXTRA under agent heterogeneity in the classes of local functions and observe that it does not depend on the total number of agents but only on the proportion of each class of functions that are present in the system.

1.2 Related work

An alternative approach for automatic computation of performance guarantees of optimization methods is proposed in [13] and relies on the formulation of optimization algorithms as dynamical systems. Integral quadratic constraints (IQC), generally used to obtain stability guarantees on complex dynamical systems, are adapted to provide sufficient conditions for the convergence of optimization methods and deduce numerical bounds on the convergence rates. This theory has been extended to decentralized optimization in [29]. The methodology analyzes the convergence rate of a single iterate, which is beneficial for the problem size that remains small, and that is also independent of the number of agents in the decentralized case. However, this does not allow dealing with non-geometric convergences, e.g. on smooth convex functions, nor to analyze cases where a property applies over several iterations, e.g. when averaging matrices are constant. These are possible with the PEP approach which computes the worst-case performance on a given number t of iterations, but solving problems whose size grows with t. Moreover, the current IQC approach only applies to settings where all the agents are equivalent, which is not the case, for example, if we want to analyze the performance of the worst agent. In this work, we propose a practical way, via the PEP approach, to compute performance where there are several equivalence classes of agents. This expands the range of situations we can automatically analyze efficiently in decentralized optimization, including more complex or realistic settings.

1.3 Notations

Let $x_i^k \in \mathbb{R}^d$ denote the k-th local variable x of agent i. The local variable of all the agents can be stacked vertically in $\mathbf{x}^k \in \mathbb{R}^{nd}$:

$$\mathbf{x}^k = \begin{bmatrix} x_1^k \\ \vdots \\ x_n^k \end{bmatrix},$$

We can also gather different iterates $k = 0, \dots, t$ in a matrix $X \in \mathbb{R}^{nd \times t+1}$:

$$X = \begin{bmatrix} \mathbf{x}^0 & \dots & \mathbf{x}^t \end{bmatrix}.$$

These vector notations apply to any variables that are used in the decentralized algorithms and assume that the local variables exist for each agent and each iteration, but our work also applies if the variables are not defined for all agents or all iterations of the algorithm. Furthermore, our approach also applies if there is no clear concept of iterations. Using these vector notations, we can write the consensus step (4) for all agents at once as

$$\mathbf{y}^k = (W^k \otimes I_d)\mathbf{x}^k, \qquad \text{for } k = 0, \dots, t$$

where $W^k \in \mathbb{R}^{n \times n}$ is the averaging matrix used in the consensus, I_d denotes the identity matrix of size d and $\otimes$ the Kronecker product. This latter notation means that we apply the same matrix W^k for each dimension of the agent variables. Moreover, if the same averaging matrix W is used for different consensus steps ($k = 0, \dots, t$), we can write all the consensus steps at once using matrix notations:

$$Y = (W \otimes I_d)X.$$

The i^{th} largest eigenvalues of a matrix W is denoted $\lambda_i(W)$. The agent average of iterate $\mathbf{x}^k \in \mathbb{R}^{nd}$ is denoted $\bar{x}^k \in \mathbb{R}^d$ and is defined as $\bar{x}^k = \frac{1}{n}\sum_{i=1}^n x_i^k$. We can also gather the agent average of different iterates $k = 0, \dots, t$ in a matrix $\overline{X} \in \mathbb{R}^{d \times t+1}$: $\overline{X} = \begin{bmatrix} \bar{x}^0 & \dots & \bar{x}^t \end{bmatrix}$. Finally, $\mathbf{1}$ is the vector full of 1.

2 Agent-dependent performance estimation problem for distributed optimization

2.1 Performance Estimation Problem (PEP) framework

To obtain a tight bound on the performance of a distributed optimization algorithm $\mathcal{A}$, the conceptual idea is to find instances of local functions and starting points for all agents, allowed by the setting considered, that give the largest error after a given number t of iterations of the algorithm. The performance estimation problem

(PEP) formulates this idea as a real optimization problem that maximizes the error measure $\mathcal{P}$ of the algorithm result, over all possible functions and initial points allowed [7]:

$$
\begin{aligned}
w(\mathcal{S}_n) \;=\; & \max_{x^*,\{x_i^k,y_i^k,f_i\}_{i\in\mathcal{V}}} \mathcal{P}(f_i, x_i^0,\dots,x_i^t,x^*) \quad \overset{\text{e.g.}}{=} \quad \frac{1}{n}\sum_{i=1}^{n}\big(f_i(\bar{x}^t) - f_i(x^*)\big) && (5)\\
\text{s.t.} \quad & x_i^k, y_i^k \quad \text{from algorithm } \mathcal{A}, \quad \text{e.g. DGD (2)–(3)} && \text{for } i\in\mathcal{V},\ k=1,\dots,t, && \text{(algorithm)}\\
& f_i \in \mathcal{F}, && \text{for } i\in\mathcal{V} && \text{(class of functions)}\\
& x_i^0 \quad \text{satisfies } \mathcal{I}, \quad \text{e.g. } \|x_i^0 - x^*\|^2 \le 1,\ i\in\mathcal{V} && && \text{(initial conditions)}\\
& \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(x^*) = 0, && && \text{(optimality condition for (1))}
\end{aligned}
$$

where $\mathcal{S}_n$ is the performance evaluation setting which specifies the number of agents n, the algorithm $\mathcal{A}$, the number of steps t, the performance criterion $\mathcal{P}$, the class of functions $\mathcal{F}$ for the local function, the initial conditions $\mathcal{I}$ and the class of averaging matrices $\mathcal{W}$ that can be used in the algorithm:

$$\mathcal{S}_n = \{n, \mathcal{A}, t, \mathcal{P}, \mathcal{F}, \mathcal{I}, \mathcal{W}\}.$$

We assume here that the local functions all belong to the same function class. This assumption is usually made in the literature but is not necessary in this PEP formulation. The problem can also have other auxiliary iterates as variables if needed for the analyzed algorithm. Here we have chosen to show auxiliary iterates y_i^k which are the results of the consensus steps (4) that may be used at each iteration of the algorithm.

The optimal value of problem (5), denoted $w(\mathcal{S}_n)$, gives by definition a tight worst-case performance bound for the given setting $\mathcal{S}_n$. Moreover, the optimal solution corresponds to an instance of local functions and initial points actually reaching this upper bound, which can provide very relevant information on the bottlenecks faced by the algorithm. Solving a PEP such as (5) is in general not easy because the problem is inherently infinite-dimensional, as it contains continuous functions f_i among its variables. Nevertheless, Taylor et al. have shown [32, 30] that PEPs can be formulated as a finite semidefinite program (SDP) and can thus be solved exactly, for a wide class of *centralized* first-order algorithms and different classes of functions. The reformulation techniques developed for classical optimization algorithms can also be applied for distributed optimization, as detailed in [6]. In what follows, we briefly explain how to reformulate the problem (5) into an SDP. One of the differences in PEP for distributed optimization is that the problem must find the worst-case for each of the n local functions f_i and the n sequences of local iterates $x_i^0 \dots x_i^t$ $(i = 1,\dots,n)$. To render problem (5) finite, for each agent i, rather than considering its local function f_i as a whole, we only consider the discrete set $\{(x_i^k, g_i^k, f_i^k)\}_{k\in I=\{0,\dots,t,*\}}$ of local iterates x_i^k together with their local gradient-vectors g_i^k and local function values f_i^k. We then impose interpolation constraints on this set to ensure its consistency with an actual function $f_i \in \mathcal{F}$, in the sense that the set is $\mathcal{F}$-*interpolable*: there exists a function $f_i \in \mathcal{F}$ such that $f_i(x_i^k) = f_i^k$ and $\nabla f_i(x_i^k) = g_i^k$. Such interpolation constraints are provided for many classes of functions in [30, Section 3], such as the class L-smooth and μ-strongly convex functions.

▶ **Proposition 3** (Interpolation constraints for $\mathcal{F}_{\mu,L}$ [30])**.** *Let I be a finite index set and $\mathcal{F}_{\mu,L}$ the set of L-smooth and μ-strongly convex functions. A set of triplets $\{(x_k, g_k, f_k)\}_{k\in I}$ is $\mathcal{F}_{\mu,L}$-interpolable if and only if the following conditions hold for every pair of indices $k \in I$ and $l \in I$:*

$$f_k - f_l - g_l^T(x_k - x_l) \ge \frac{1}{2(1-\mu/L)}\left(\frac{1}{L}\|g_k - g_l\|^2 + \mu\|x_k - x_l\|^2 - 2\frac{\mu}{L}(g_k - g_l)^T(x_k - x_l)\right). \tag{6}$$

Then, all the constraints from (5) can be expressed in terms of $\{y_i^k, x_i^k, g_i^k, f_i^k\}_{k\in I=\{0,\dots,t,*\}}$. When the averaging matrix W is given, i.e. $\mathcal{W} = \{W\}$, the algorithm constraints simply consist of linear constraints between these variables. The optimality condition for (1) can be expressed as a linear constraint on the local gradients at x^*. The performance criterion, the initial conditions, and the interpolation constraints are usually quadratic and potentially non-convex in the local iterates and local gradients but they are often linear in the scalar products of these and in the function values, see (6) for example. In such cases, letting the decision variables of the PEP be these scalar products and the function values, one can reformulate it as a semi-definite program (SDP) which can be solved efficiently. For this purpose, we define a vector of function values $\mathbf{f}$ and a

Gram matrix G that contains scalar products between all vectors, e.g. the local iterates $y_i^k, x_i^k \in \mathbb{R}^d$ and the local gradient vectors $g_i^k \in \mathbb{R}^d$.

$$\mathbf{f} = [f_i^k]_{i\in\mathcal{V},\ k\in I=\{0,\dots,t,*\}} \in \mathbb{R}^{n(t+2)}, \quad G = P^T P \in \mathbb{R}^{3n(t+2)\times 3n(t+2)}, \text{ with } P = \left[y_i^k\ x_i^k\ g_i^k\right]_{i\in\mathcal{V},k\in I}. \tag{7}$$

By definition, G is symmetric and positive semidefinite. Moreover, every matrix $G \succeq 0$ corresponds to a matrix P whose number of rows is equal to rank G. It has been shown in [30] that the reformulation of a PEP, such as (5), with G as variable is lossless provided that we use necessary and sufficient interpolation constraints for $\mathcal{F}$ and that we do not impose the dimension d of the worst-case, 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 resulting SDP-PEP thus takes the form of

$$\begin{aligned} w(\mathcal{S}_n) \quad = \quad & \max_{\mathbf{f},G} \quad \mathcal{P}(\mathbf{f},G) \\ & \text{s.t.} \quad\ G \succeq 0, \\ & \quad \mathbf{f},G \quad \text{satisfy} \quad \begin{array}{l} \text{algorithm constraints for } \mathcal{A} \\ \text{interpolation constraints for } \mathcal{F}, \\ \text{initial conditions } \mathcal{I}, \\ \text{optimality condition for (1)}, \end{array} \end{aligned} \tag{8}$$

where the objective function $\mathcal{P}$ and the constraints are linear in $\mathbf{f}$ and G. The performance criterion function $\mathcal{P}$ in (8) corresponds to the same function as the function $\mathcal{P}$ in (5), up to a change of variables. We slightly abuse notations to denote these two functions with the same symbol $\mathcal{P}$, for readibility.

Such an SDP formulation (8) 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 [30] for more details about the SDP formulation of a PEP, including ways of reducing the size of matrix G. However, the dimension of G always depends on the number of iterations t, and on the number of agents n.

From a solution G, $\mathbf{f}$ of the SDP formulation, we can construct a solution for the variables $\{y_i^k, x_i^k, g_i^k, f_i^k\}_{i\in\mathcal{V},k\in I}$, e.g. using the Cholesky decomposition of G. For each agent $i \in \mathcal{V}$, its resulting set of points satisfies the interpolation constraints, so we can construct its corresponding worst-case local function from $\mathcal{F}$ interpolating these points. Proposition 5 below states sufficient conditions under which a PEP on a distributed optimization method can be formulated as an SDP. These conditions are satisfied in many PEP settings, allowing to formulate and solve the SDP PEP formulation for any distributed first-order methods from $\mathcal{A}_D$, with many classes of local functions, initial conditions, and performance criteria, see [30, 6, 3]. The proposition uses the following definition.

▶ **Definition 4** (Gram-representable [30]). *Consider a Gram matrix G and a vector $\mathbf{f}$, as defined in (7). We say that an expression, such as 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 or expressions involving (part of) G and $\mathbf{f}$.*

▶ **Proposition 5** ([30, Proposition 2.6]). *Let $w(\mathcal{S}_n = \{n, \mathcal{A}, t, \mathcal{P}, \mathcal{F}, \mathcal{I}, \mathcal{W}\})$ be the worst-case performance of the execution of t iterations of distributed algorithm $\mathcal{A}$ with $n \geq 2$ agents, with respect to the performance criterion $\mathcal{P}$, valid for any starting point satisfying the set of initial conditions $\mathcal{I}$, any local functions in a given set of functions $\mathcal{F}$ and any averaging matrix in the set of matrices $\mathcal{W}$. If $\mathcal{A}, \mathcal{P}, \mathcal{I}$ and interpolation constraints for $\mathcal{F}$ and for $\mathcal{W}$ are linearly (or LMI) Gram representable, then the computation of $w(\mathcal{S}_n)$ can be formulated as an SDP, with $G \succeq 0$ and $\mathbf{f}$ as variables, see (8).*

▶ Remark. In [30], Definition 4 and Proposition 5 were only formulated for linearly Gram-representable constraints, but their extensions to LMI Gram-representable constraints are direct and were already introduced in [6] in which LMI constraints are used to represent consensus steps in PEP.

As briefly mentioned, when the set of possible averaging matrices $\mathcal{W}$ contains only one given matrix W, i.e. we know the averaging matrix used, the PEP framework presented here is complete and provides worst-case bounds specific to this value of W. However, the literature on distributed optimization generally provides bounds on the performance of an algorithm that are valid over a larger set of averaging matrices $\mathcal{W}$, typically based on its spectral properties, see [18] for a survey. Such bounds allow better characterization of the general behavior of the algorithm and can be applied in a wider range of settings. Obtaining such bounds with PEP requires having

a tractable representation of the set of all pairs of variables $\{(\mathbf{x}^k, \mathbf{y}^k)\}$ that can be involved in consensus steps with the same matrix from the given matrix class. Our previous work [6] proposes and uses necessary constraints for describing the set of symmetric and doubly-stochastic matrices with a given range on the non-principal eigenvalues. The next section describes these constraints and analyzes their sufficiency.

2.2 Representation of consensus steps in PEP

In PEP for a decentralized optimization algorithm $\mathcal{A} \in \mathcal{A}_D$, with $n \geq 2$ agents, we would like to find Gram-representable constraints to represent all the pairs of points $\{\mathbf{x}^k, \mathbf{y}^k\}$ such that

$$\mathbf{y}^k = (W \otimes I_d)\mathbf{x}^k, \quad \text{for } k = 1, \dots, t, \qquad \text{with } \; W \in \mathcal{W}_{(\lambda^-,\lambda^+)}, \tag{9}$$

where $\mathcal{W}_{(\lambda^-,\lambda^+)}$ is the following set of symmetric averaging matrices, with fixed bounds $\lambda^-, \lambda^+ \in (-1, 1)$ on the eigenvalues (except for $\lambda_1 = 1$):

$$\mathcal{W}_{(\lambda^-,\lambda^+)} = \left\{ W \in \mathbb{R}^{n\times n} \; : \; \begin{array}{c} W^T = W \\ \lambda_1(W) = 1, \; v_1(W) = \mathbf{1}/\sqrt{n} \\ \lambda^- \leq \lambda_n(W) \leq \cdots \leq \lambda_2(W) \leq \lambda^+ \end{array} \right\}. \tag{10}$$

Such a set is often used to derive theoretical bounds on distributed algorithms [18]. You may notice that it is not restricted to non-negative matrices, even if the literature often assumes non-negativity through doubly-stochasticity. We motivate this choice by two arguments. Firstly, non-negativity is not needed for the convergence of pure consensus steps [34]. Secondly, among results using doubly-stochasticity, we found no results exploiting the non-negativity of the matrix, see e.g. [18, 25, 16]. Such results are thus about *generalized doubly-stochastic* matrices [10], which refers to matrices whose rows and columns sum to one, as the ones contained in $\mathcal{W}_{(\lambda^-,\lambda^+)}$.

A set of pairs of points $\{(\mathbf{x}^k, \mathbf{y}^k)\}$ satisfying the consensus steps constraints (9) is called $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable.

▶ **Definition 6** ($\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolability)**.** *Let I be a finite set of indices. A set of pairs $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ is $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable if,*

$$\exists \, W \in \mathcal{W}_{(\lambda^-,\lambda^+)} \; : \; \mathbf{y}^k = (W \otimes I_d)\mathbf{x}^k \quad \textit{for all } k \in I.$$

In this definition, the pairs of points $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ are all linked by the same averaging matrix $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$. Thus, settings or algorithms using different averaging matrices work with different sets of pairs of points.

This section first shows why there is no Gram-representable constraints to characterize $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolability of a set of pairs of points $\{\mathbf{x}^k, \mathbf{y}^k\}$ involved in consensus steps of the form of (9), preventing its tight representation in an SDP PEP formulation. Then we present a relaxed Gram-representable description of $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable pairs of points and characterize the points allowed in this relaxed description. Finally, we also explain how it can be used to build the *spectral agent-dependent PEP formulation*, which provides worst-case guarantees for a decentralized algorithm, that are valid for any averaging matrix from $\mathcal{W}_{(\lambda^-,\lambda^+)}$.

2.2.1 The intractable quest for tightness

It would be ideal to find necessary and sufficient conditions for a set of pairs $\{(\mathbf{x}^k, \mathbf{y}^k)\}$ to be $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable, see Definition 6. Moreover, the conditions should be Gram-representable, i.e., linear in the scalar products of $\mathbf{x}_i$ and $\mathbf{y}_i$, to be able to use them in an SDP PEP formulation (see Proposition 5). In this subsection, we will show, by a non-convexity argument, that there are no tight conditions for $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolability that are Gram-representable.

▶ **Theorem 7.** *There is no Gram-representable necessary and sufficient conditions for $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolability of a set of pairs of variables $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$.*

The proof of this theorem relies on a lemma showing the non-convexity of the set of Gram matrices we are considering. To define this set properly, we consider matrices $P_X, P_Y \in \mathbb{R}^{d\times nt}$ aggregating consensus iterates of n agents over t consensus steps.

▶ **Definition 8** ($\mathcal{G}_{(\lambda^-,\lambda^+)}$). *We define $\mathcal{G}_{(\lambda^-,\lambda^+)}$ as the set of symmetric and positive semidefinite Gram matrices of the form*

$$G_c = \begin{bmatrix} P_X^T P_X & P_X^T P_Y \\ P_Y^T P_X & P_Y^T P_Y \end{bmatrix}, \quad \text{where} \quad \begin{aligned} P_X &= [\overbrace{x_1^0 \dots x_n^0}^{\mathbf{x}^0} \dots x_1^t \dots x_n^t] \in \mathbb{R}^{d\times nt}, \\ P_Y &= [\underbrace{y_1^0 \dots y_n^0}_{\mathbf{y}^0} \dots y_1^t \dots y_n^t] \in \mathbb{R}^{d\times nt}, \end{aligned}$$

such that the set of pairs $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k=1,\dots,t}$ is $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable, in the sense of Definition 6.

In practice, in the PEPs, the Gram matrices can involve other variables than those of the consensus (e.g., gradients). We can therefore consider G_c as a submatrix of the full Gram matrix, which contains only the scalar products related to consensus steps.

▶ **Lemma 9.** *Let $\lambda^- < \lambda^+ \in (-1, 1)$. The set $\mathcal{G}_{(\lambda^-,\lambda^+)}$ is non-convex.*

Proof. We build two Gram matrices $G_1, G_2 \in \mathcal{G}_{(\lambda^-,\lambda^+)}$ and show that their combination $G_3 = \frac{1}{2}(G_1 + G_2)$ is not in $\mathcal{G}_{(\lambda^-,\lambda^+)}$. We build a counter-example for one consensus step, i.e., for $t = 1$, and then we explain why it can also be used for $t > 1$. In the set $\mathcal{G}_{(\lambda^-,\lambda^+)}$, there is no constraint on the rank d of the matrices, so we choose to build G_1 and G_2 with rank $d = 1$, as follows

$$\begin{aligned} &\text{let} \qquad \mathbf{x} = \begin{bmatrix} n & 0 & \dots & 0 \end{bmatrix}^T \in \mathbb{R}^n, \\ &\qquad [W_1]_{ij} = \begin{cases} \frac{1+(n-1)\lambda^-}{n} & \text{for } i = j \\ \frac{1-\lambda^-}{n} & \text{for } i \neq j \end{cases}, \qquad [W_2]_{ij} = \begin{cases} \frac{1+(n-1)\lambda^+}{n} & \text{for } i = j \\ \frac{1-\lambda^+}{n} & \text{for } i \neq j \end{cases}, \\ &\text{and} \qquad \mathbf{y}_1 = W_1\mathbf{x} = \begin{bmatrix} 1+(n-1)\lambda^- \\ 1-\lambda^- \\ \vdots \\ 1-\lambda^- \end{bmatrix}, \qquad \mathbf{y}_2 = W_2\mathbf{x} = \begin{bmatrix} 1+(n-1)\lambda^+ \\ 1-\lambda^+ \\ \vdots \\ 1-\lambda^+ \end{bmatrix}, \end{aligned} \tag{11}$$

we build G_1 and G_2 as

$$G_1 = \begin{bmatrix} \mathbf{x}\mathbf{x}^T & \mathbf{x}\mathbf{y}_1^T \\ \mathbf{y}_1\mathbf{x}^T & \mathbf{y}_1\mathbf{y}_1^T \end{bmatrix}, \qquad G_2 = \begin{bmatrix} \mathbf{x}\mathbf{x}^T & \mathbf{x}\mathbf{y}_2^T \\ \mathbf{y}_2\mathbf{x}^T & \mathbf{y}_2\mathbf{y}_2^T \end{bmatrix}.$$

The eigenvalues of W_1 are $\{\lambda^-, \dots, \lambda^-, 1\}$, and those of W_2 are $\{\lambda^+, \dots, \lambda^+, 1\}$, so that $W_1, W_2 \in \mathcal{W}_{(\lambda^-,\lambda^+)}$. Therefore, by construction, both pairs $(\mathbf{x}, \mathbf{y}_1)$ and $(\mathbf{x}, \mathbf{y}_2)$ are $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable, and so G_1 and G_2 are in $\mathcal{G}_{(\lambda^-,\lambda^+)}$, and can be combined as $G_3 = \frac{1}{2}(G_1 + G_2)$,

$$G_3 = \begin{bmatrix} \mathbf{x}\mathbf{x}^T & \frac{1}{2}\mathbf{x}(\mathbf{y}_1 + \mathbf{y}_2)^T \\ \frac{1}{2}(\mathbf{y}_1 + \mathbf{y}_2)\mathbf{x}^T & \frac{1}{2}(\mathbf{y}_1\mathbf{y}_1^T + \mathbf{y}_2\mathbf{y}_2^T) \end{bmatrix} = \begin{bmatrix} P_{X_3}^T P_{X_3} & P_{X_3}^T P_{Y_3} \\ P_{Y_3}^T P_{X_3} & P_{Y_3}^T P_{Y_3} \end{bmatrix}. \tag{12}$$

We show that, for any dimension $d_3 > 0$, there is no $P_{X_3}, P_{Y_3} \in \mathbb{R}^{d_3\times n}$ satisfying (12) and such that $G_3 \in \mathcal{G}_{(\lambda^-,\lambda^+)}$. Let us proceed by contradiction. Suppose there is some P_{X_3}, P_{Y_3} satisfying (12) and

$$P_{Y_3}^T = W_3 P_{X_3}^T, \quad \text{for } W_3 \in \mathcal{W}_{(\lambda^-,\lambda^+)}. \tag{13}$$

Then, the block $P_{X_3}^T P_{X_3} = \mathbf{x}\mathbf{x}^T$ has rank one, because $\mathbf{x} \in \mathbb{R}^n$, and thus the block $P_{Y_3}^T P_{Y_3}$, which can be expressed as $P_{Y_3}^T P_{Y_3} = W_3 P_{X_3}^T P_{X_3} W_3$ using (13), also has rank one[1]. However, the block is also defined as $P_{Y_3}^T P_{Y_3} = \frac{1}{2}(\mathbf{y}_1\mathbf{y}_1^T + \mathbf{y}_2\mathbf{y}_2^T)$ from (12), which has rank 2 for any $\lambda^- \neq \lambda^+ \in (-1, 1)$, because vectors $\mathbf{y}_1$ and $\mathbf{y}_2$ (11) are linearly independent in that case. By this contradiction argument, we have proved that $G_3 = \frac{1}{2}(G_1 + G_2)$ is not in $\mathcal{G}_{(\lambda^-,\lambda^+)}$, meaning that this set is not convex for $t = 1$.

When we consider multiple consensus steps using the same matrix, i.e., when $t > 1$, the counter-example developed above for $t = 1$ can still be chosen for one of the steps, and the Gram matrices G_1, G_2, and G_3 would then be submatrices of the Gram matrices in $\mathcal{G}_{(\lambda^-,\lambda^+)}$, showing the non-convexity of $\mathcal{G}_{(\lambda^-,\lambda^+)}$ for $t > 1$ as well. ◀

Proof of Theorem 7. By definition, Gram-representable constraints define a convex subset of symmetric and semidefinite Gram-matrices. However, by Lemma 9, we know that set $\mathcal{G}_{(\lambda^-,\lambda^+)}$ of Gram-matrices such that the pairs of points $\{\mathbf{x}^k, \mathbf{y}^k\}$ are $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable is non-convex. Therefore, we know that there exists no Gram-representable constraints that tightly describe all the points $\{\mathbf{x}^k, \mathbf{y}^k\}$ that are $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable. ◀

[1] This is due to $\text{rank}(AB) \leq \min(\text{rank}(A), \text{rank}(B))$.

2.2.2 A relaxed formulation

In previous work [6], necessary constraints have been used to represent the effect of consensus steps in PEPs. These constraints describe a convex set of Gram matrices that contains the non-convex set $\mathcal{G}_{(\lambda^-,\lambda^+)}$, and can therefore be exploited in PEP to derive upper bounds on the worst-case performance of distributed algorithms. However, the precise meaning of these constraints was not known. In this subsection, we show that these constraints correspond to tight interpolation constraints for a larger class of steps, containing the consensus step as a particular case.

Generalized consensus steps

The consensus steps (9) can be seen as a linear operation on vectors $\mathbf{x}^k \in \mathbb{R}^{nd}$, with a matrix $M \in \mathbb{R}^{nd\times nd}$ that has a specific Kronecker structure:

$$\mathbf{y}^k = M\mathbf{x}^k, \quad \text{for } k = 1,\dots,t \qquad \text{with } \; M = W \otimes I_d \;\text{ and }\; W \in \mathcal{W}_{(\lambda^-,\lambda^+)}.$$

By definition, matrix M has the same eigenvalues as $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$ (10), with multiplicity d times larger. Therefore, it has d eigenvalues equal to 1, and their associated eigenvectors form a basis of the consensus subspace $\mathcal{C}$.

▶ **Definition 10** (Consensus subspace). *The consensus subspace of $\mathbb{R}^{nd}$ is denoted $\mathcal{C}$ and is defined as*

$$\mathcal{C} = \left\{\mathbf{x} \in \mathbb{R}^{nd} \;:\; x_1 = \cdots = x_n \in \mathbb{R}^d\right\}.$$

The dimension of $\mathcal{C}$ is $|\mathcal{C}| = d$. The orthogonal complement of $\mathcal{C}$ is denoted $\mathcal{C}^\perp$ and has dimension $|\mathcal{C}^\perp| = (n-1)d$:

$$\mathcal{C}^\perp = \left\{\mathbf{x} \in \mathbb{R}^{nd} \;:\; \mathbf{x}^T\mathbf{y} = 0, \text{ for all } \mathbf{y} \in \mathcal{C}\right\}.$$

We consider generalized consensus steps where the Kronecker structure of M is relaxed, so that the vectors $\mathbf{x}^k$ are multiplied by a full matrix with similar properties:

$$\mathbf{y}^k = M\mathbf{x}^k, \quad \text{for } k = 1,\dots,t, \qquad \text{with } \; M \in \mathcal{M}_{(\lambda^-,\lambda^+)}, \tag{14}$$

where $\mathcal{M}_{(\lambda^-,\lambda^+)}$ is a set of symmetric matrices that have the same properties as $W \otimes I_d$ with $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$, but without constraint on their structure. In particular, matrices in $\mathcal{M}_{(\lambda^-,\lambda^+)}$ are still symmetric, generalized doubly stochastic and with the same range of eigenvalues. The set $\mathcal{M}_{(\lambda^-,\lambda^+)}$ is formally defined as follows, for a given range of eigenvalue $\lambda^-, \lambda^+ \in (-1,1)$, $\lambda^- \le \lambda^+$:

$$\mathcal{M}_{(\lambda^-,\lambda^+)} = \left\{ M \in \mathbb{R}^{nd\times nd} \;:\; \begin{array}{l} M^T = M \\ \lambda_1(M) = \cdots = \lambda_{|\mathcal{C}|}(M) = 1, \\ \text{with eigenvectors forming a basis of } \mathcal{C} \\ \lambda^- \le \lambda_{nd}(M) \le \cdots \le \lambda_{|\mathcal{C}|+1}(M) \le \lambda^+ \end{array} \right\},$$

We say that a matrix $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$ preserves the consensus subspace $\mathcal{C}$ as we have $M\mathbf{v} = \mathbf{v}$ for any $\mathbf{v} \in \mathcal{C}$. Relaxation of the Kronecker structure in a consensus step allows different weights to be applied to each component of the agent vectors, and different components to be mixed.

Interpolation constraints for $\mathcal{M}_{(\lambda^-,\lambda^+)}$

We provide tight interpolation constraints on a set of pairs of points $\{\mathbf{x}^k, \mathbf{y}^k\}$ for such generalized consensus steps (14), hence providing necessary constraints for describing classical consensus steps (9). Interpolation constraints are necessary and sufficient conditions for a set of pairs of points $\{\mathbf{x}^k, \mathbf{y}^k\}$ to be interpolable by a matrix $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$.

▶ **Definition 11** ($\mathcal{M}_{(\lambda^-,\lambda^+)}$-interpolability). *Let I be a finite set of indices. A set of pairs of points $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ is $\mathcal{M}_{(\lambda^-,\lambda^+)}$-interpolable if,*

$$\exists\, M \in \mathcal{M}_{(\lambda^-,\lambda^+)} \;:\; \mathbf{y}^k = M\mathbf{x}^k \quad \text{for all } k \in I.$$

This definition is related to the interpolation of a symmetric linear operator, developed in [1]. In this work, we focus on linear operators preserving a given subspace, namely the consensus subspace $\mathcal{C}$. Based on results from [6] and [1], we give necessary and sufficient conditions to have $\mathcal{M}_{(\lambda^-,\lambda^+)}$-interpolability of a set of pairs of points. Before stating these interpolation conditions, we first introduce a few notions. We aggregate all the vectors $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ in matrices $X, Y \in \mathbb{R}^{nd\times t}$:

$$X = [\mathbf{x}^1 \dots \mathbf{x}^t] \quad \text{and} \quad Y = [\mathbf{y}^1 \dots \mathbf{y}^t]$$

We also consider the decomposition of matrices X and Y in two orthogonal terms:

$$X = X_{/\!/} + X_\perp \text{ and } Y = Y_{/\!/} + Y_\perp,$$

where $X_{/\!/}, Y_{/\!/} \in \mathbb{R}^{nd\times t}$ and $Y_\perp, X_\perp \in \mathbb{R}^{nd\times t}$ are obtained by projecting each column of X and Y respectively on the consensus subspace $\mathcal{C}$ and on its orthogonal subspace $\mathcal{C}^\perp$. This decomposition allows decoupling the effect of the generalized consensus steps (14) as follows:

$$Y = MX = M(X_{/\!/} + X_\perp) = X_{/\!/} + MX_\perp \quad \iff \quad Y_{/\!/} = X_{/\!/} \text{ and } Y_\perp = MX_\perp,$$

where $MX_{/\!/} = X_{/\!/}$ because $X_{/\!/} \in \mathcal{C}$ and $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$.

▶ **Theorem 12** ($\mathcal{M}_{(\lambda^-,\lambda^+)}$ interpolation constraints)**.** *Let I be a set of t indices and $\lambda^-, \lambda^+ \in (-1, 1)$, $\lambda^- \le \lambda^+$. A set of pairs of points $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ is $\mathcal{M}_{(\lambda^-,\lambda^+)}$-interpolable if, and only if,*

$$X_{/\!/} = Y_{/\!/}, \tag{15}$$

$$(Y_\perp - \lambda^- X_\perp)^T (Y_\perp - \lambda^+ X_\perp) \preceq 0, \tag{16}$$

$$X_\perp^T Y_\perp = Y_\perp^T X_\perp, \tag{17}$$

where $X_{/\!/}, Y_{/\!/} \in \mathbb{R}^{nd\times t}$ contains columns of X and Y projected onto the subspace $\mathcal{C}$, and $X_\perp, Y_\perp \in \mathbb{R}^{nd\times t}$ the columns projected on its orthogonal complement $\mathcal{C}^\perp$.

▶ Remark. Theorem 12 is stated for the case where symmetric linear operators from $\mathcal{M}_{(\lambda^-,\lambda^+)}$ preserve the consensus subspace $\mathcal{C}$ (Definition 10), but it actually holds for any general preserved subspace $\mathcal{B} \subseteq \mathbb{R}^{nd}$. Indeed, the proof below remains valid if we replace the consensus subspace $\mathcal{C}$ by any subspace $\mathcal{B}$ (and adapt the dimensions where needed).

Proof of Theorem 12. We need to prove that conditions (15), (16) and (17) are necessary and sufficient for the existence of a matrix $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$ such that $Y = MX$. The proof relies on the use of [1, Theorem 3.3] which proved the necessity and sufficiency of interpolation constraints similar to (16) and (17), for general linear operators, without any preserved subspace. In our case, we know that the matrix M has no impact on the consensus subspace $\mathcal{C}$ and only acts on the part in $\mathcal{C}^\perp$. We thus need to use [1, Theorem 3.3] on this part only. Decoupling both parts can be done using an appropriate change of variables:

$$\widetilde{X} = \begin{bmatrix} \widetilde{X}_{/\!/} \\ \widetilde{X}_\perp \end{bmatrix} = \begin{bmatrix} Q_{/\!/}^T \\ Q_\perp^T \end{bmatrix} X = Q^T X, \tag{18}$$

where $Q = [Q_{/\!/}\, Q_\perp] \in \mathbb{R}^{nd\times nd}$ is an orthogonal matrix of change of variables and is split into two sets of columns: $Q_{/\!/} \in \mathbb{R}^{nd\times|\mathcal{C}|}$ form a basis of the consensus subspace $\mathcal{C}$ and $Q_\perp \in \mathbb{R}^{nd\times|\mathcal{C}^\perp|}$ a basis of its orthogonal complement $\mathcal{C}^\perp$. The new variable $\widetilde{X}$ has thus two sets of components, $\widetilde{X}_{/\!/} \in \mathbb{R}^{|\mathcal{C}|\times t}$ describing the coordinates along the subspace $\mathcal{C}$, and $\widetilde{X}_\perp \in \mathbb{R}^{|\mathcal{C}^\perp|\times t}$ describing the coordinates along the orthogonal complement $\mathcal{C}^\perp$. By definition, $X_{/\!/}$ and $Y_{/\!/}$ have components only along $\mathcal{C}$ and $X_\perp$ and $Y_\perp$ only along $\mathcal{C}^\perp$:

$$X_{/\!/} = Q \begin{bmatrix} \widetilde{X}_{/\!/} \\ 0 \end{bmatrix}, \qquad Y_{/\!/} = Q \begin{bmatrix} \widetilde{Y}_{/\!/} \\ 0 \end{bmatrix}, \qquad \text{and} \qquad X_\perp = Q \begin{bmatrix} 0 \\ \widetilde{X}_\perp \end{bmatrix}, \qquad Y_\perp = Q \begin{bmatrix} 0 \\ \widetilde{Y}_\perp \end{bmatrix}. \tag{19}$$

By applying this change of variable to condition (15), we obtain $\widetilde{X}_{/\!/} = \widetilde{Y}_{/\!/}$. The two conditions are well equivalent because the change of variable is reversible. By applying this change of variable (19) to conditions (16) and (17), since Q is orthogonal, i.e., $Q^T Q = I$, we obtain

$$(\widetilde{Y}_\perp - \lambda^- \widetilde{X}_\perp)^T (\widetilde{Y}_\perp - \lambda^+ \widetilde{X}_\perp) \preceq 0,$$
$$\widetilde{X}_\perp^T \widetilde{Y}_\perp = \widetilde{Y}_\perp^T \widetilde{X}_\perp.$$

As the change of variable is reversible, the two sets of constraints are equivalent. By [1, Theorem 3.3], these two constraints are necessary and sufficient for the existence of a symmetric matrix $\widetilde{M}_\perp \in \mathbb{R}^{|\mathcal{C}^\perp|\times|\mathcal{C}^\perp|}$ with eigenvalues between λ^- and λ^+ such that $\widetilde{Y}_\perp = \widetilde{M}_\perp \widetilde{X}_\perp$. Therefore, we have proved that conditions (15), (16) and (17) are equivalent to

$$\begin{bmatrix} \widetilde{Y}_{/\!/} \\ \widetilde{Y}_\perp \end{bmatrix} = \begin{bmatrix} I_d & 0 \\ 0 & \widetilde{M}_\perp \end{bmatrix} \begin{bmatrix} \widetilde{X}_{/\!/} \\ \widetilde{X}_\perp \end{bmatrix} \text{ with } \widetilde{M}_\perp = \widetilde{M}_\perp^T,\ \lambda(\widetilde{M}_\perp) \in [\lambda^-, \lambda^+]. \tag{20}$$

We now need to come back to the initial variables X and Y and characterize the matrix M in terms of $\widetilde{M}_\perp$. We can use the (reversible) change of variable (18) to express equation (20) in terms of Y and X

$$Y = \underbrace{Q \begin{bmatrix} I_d & 0 \\ 0 & \widetilde{M}_\perp \end{bmatrix} Q^T}_{M} X \qquad \text{with } \lambda(\widetilde{M}_\perp) \in [\lambda^-, \lambda^+].$$

By splitting Q as $Q = [Q_{/\!/}\ Q_\perp]$, matrix M can be written as

$$M = Q_{/\!/} Q_{/\!/}^T + Q_\perp \widetilde{M}_\perp Q_\perp^T \qquad \text{with } \lambda(\widetilde{M}_\perp) \in [\lambda^-, \lambda^+].$$

Since $\widetilde{M}_\perp$ is symmetric, this expression shows that M is also symmetric. We now show that the symmetric matrix M has $|\mathcal{C}|$ eigenvalues equal to 1, with associated eigenvectors $Q_{/\!/}$. Let $v_{/\!/}$ denote any column of $Q_{/\!/}$, i.e., any basis vector of the consensus subspace $\mathcal{C}$, we have well that

$$M v_{/\!/} = Q_{/\!/} Q_{/\!/}^T v_{/\!/} + Q_\perp \widetilde{M}_\perp Q_\perp^T v_{/\!/} = v_{/\!/}.$$

Indeed, $Q_\perp^T v_{/\!/} = 0$ and $Q_{/\!/} Q_{/\!/}^T v_{/\!/} = v_{/\!/}$, since $v_{/\!/}$ is a column of $Q_{/\!/}$ which satisfies $Q_\perp^T Q_{/\!/} = 0$ and $Q_{/\!/}^T Q_{/\!/} = I$.
The other eigenvalues of M are equal to the ones of $\widetilde{M}_\perp$. Indeed, Let $(v_\perp, \lambda_\perp)$ be a pair of eigenvector and eigenvalue of matrix $\widetilde{M}_\perp \in \mathbb{R}^{\mathcal{C}^\perp \times \mathcal{C}^\perp}$, then $(Q_\perp v_\perp, \lambda_\perp)$ is a pair of eigenvector and eigenvalue of matrix $M \in \mathbb{R}^{nd\times nd}$. This can be verified easily as

$$M(Q_\perp v_\perp) = Q_{/\!/} Q_{/\!/}^T (Q_\perp v_\perp) + Q_\perp \widetilde{M}_\perp Q_\perp^T (Q_\perp v_\perp) = \lambda_\perp Q_\perp v_\perp,$$

since $Q_{/\!/}^T Q_\perp = 0$ and $Q_\perp^T Q_\perp = I$. ◀

In Theorem 12, we have proved that conditions (15), (16) and (17) are necessary and sufficient for the existence of a matrix $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$ such that $Y = MX$. We now give some interpretation of the constraints, explain why they are only necessary for $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolability, and describe the impacts in the PEPs for decentralized optimization.

Interpretation of the constraints

By Definition 10 of the consensus subspace, the orthogonal decomposition terms can be expressed using the agent averages:

$$X_{/\!/} = \overline{X} \otimes \mathbf{1}_n \qquad X_\perp = X - X_{/\!/}, \tag{21}$$

$$Y_{/\!/} = \overline{Y} \otimes \mathbf{1}_n \qquad Y_\perp = X - Y_{/\!/}, \tag{22}$$

where $\overline{X}$, $\overline{Y} \in \mathbb{R}^{d\times t}$ are agent average of X and Y, i.e., $\overline{X}_{\cdot,k} = \frac{1}{n}\sum_{i=1}^n x_i^k$. Therefore, constraint (15) is equivalent to $\overline{X} = \overline{Y}$. This constraint is related to the 1 eigenvalues of the matrix and their corresponding eigenvectors, which form a basis for the consensus space $\mathcal{C}$. The constraint imposes that each vector $\mathbf{x}^k$ have the same agents average as $\mathbf{y}^k$, for each generalized consensus step (14) ($k = 1, \dots, t$).

Interpretations of the linear matrix inequality (LMI) constraints (16) and (17) are detailed in [6]. In summary, (16) requires the disagreement between the agents, measured by $y_\perp^T y_\perp$ for y and $x_\perp^T x_\perp$ for x to be reduced by a factor $\lambda^2 \in [0, 1)$ after each consensus step. Constraints (16) and (17) also allow linking different consensus steps to each other, via the impact of off-diagonal terms which exploit the fact that these steps use the same averaging matrix.

Necessary interpolation conditions for $\mathcal{W}_{(\lambda^-,\lambda^+)}$

In Theorem 12, we have proved that conditions (15), (16) and (17) are necessary and sufficient for the existence of a matrix $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$ such that $Y = MX$. By definition of $\mathcal{M}_{(\lambda^-,\lambda^+)}$ and $\mathcal{W}_{(\lambda^-,\lambda^+)}$, these conditions are necessary for the existence of an averaging matrix $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$ such that $Y = (W \otimes I_d)X$, as shown in the following corollary.

▶ **Corollary 13.** *Let $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ be a set of t pairs of points and $\lambda^-, \lambda^+ \in (-1, 1)$, $\lambda^- \leq \lambda^+$. If $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ is $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable (Definition 6), then we have*

$$\overline{X} = \overline{Y}, \tag{23}$$

$$(Y_\perp - \lambda^- X_\perp)^T (Y_\perp - \lambda^+ X_\perp) \preceq 0, \tag{24}$$

$$X_\perp^T Y_\perp = Y_\perp^T X_\perp, \tag{25}$$

with $\overline{X}, \overline{Y} \in \mathbb{R}^{d\times t}$ and $X_\perp, Y_\perp \in \mathbb{R}^{nd\times t}$ defined in (21)–(22).

Proof. From a matrix $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$, we can always build a matrix $M = (W \otimes I_d) \in \mathcal{M}_{(\lambda^-,\lambda^+)}$, meaning that any set of pairs of points that is $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable is also $\mathcal{M}_{(\lambda^-,\lambda^+)}$-interpolable. Therefore, the points $\{(\mathbf{x}^k, \mathbf{y}^k)\}_{k\in I}$ are $\mathcal{M}_{(\lambda^-,\lambda^+)}$-interpolable and satisfy constraints (23), (24) and (25) by Theorem 12. ◀

The reverse statement does not hold since constraints (23), (24) and (25) guarantee the interpolation by a matrix $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$, that may not have the Kronecker structure required to extract an averaging matrix $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$. The full mask of $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$ allows different weights to be applied to each component of the agent and can also mix different components together, which is not the case for consensus steps (9) that impose a sparse mask of the form $(W \otimes I_d)$.

Relaxed Performance Estimations Problems

One can show that the interpolation constraints stated in Corollary 13 are LMI Gram-representable. The detailed proof is given in [6, Theorem 1] but the main idea is to express the constraints with the scalar products of the local variables x_i^k, y_i^k. Therefore, by Proposition 5, these constraints can be used in SDP PEP formulations. As explained in [6], using necessary constraints (23), (24) and (25) to represent consensus steps of the form of (9) in PEP allows obtaining worst-case performance guarantees for a decentralized optimization algorithm $\mathcal{A} \in \mathcal{A}_D$, that are valid for any averaging matrix $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$. This formulation is therefore called the *spectral agent-dependent PEP formulation* and its results the *spectral worst-case*. This is an agent-dependent PEP formulation because the PEP depends on each agent individually.

When different averaging matrices are used for different sets of consensus steps in the algorithm, these constraints from Corollary 13 can be applied independently to each set of consensus steps.

The bounds obtained with this formulation have a priori no tightness guarantees because it considers larger sets of possible consensus iterates $\{\mathbf{x}^k, \mathbf{y}^k\}$, i.e., those that are $\mathcal{M}_{(\lambda^-,\lambda^+)}$-interpolable. However, this relaxation of the Kronecker structure in consensus steps does not appear to impact the worst-case results, as already observed numerically in [6]. Moreover, one can always check the tightness of the PEP solution a posteriori, by checking if the provided solution is $\mathcal{W}_{(\lambda^-,\lambda^+)}$-interpolable. If successful, this check also recovers an instance of the worst matrix $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$, see [6] for details.

Impact of the relaxation

While the literature for distributed optimization usually assumes consensus steps of the form $\mathbf{y} = (W \otimes I_d)\mathbf{x}$ with $W \in \mathcal{W}_{(\lambda^-,\lambda^+)}$, the existing theoretical performance guarantees also appear to use non-tight description of these consensus steps. Indeed, we did not find any theoretical guarantee that exploits the Kronecker structure $(W \otimes I_d)$ of the consensus steps. In particular, many results in decentralized optimization remain valid for generalized consensus steps $\mathbf{y} = M\mathbf{x}$ with $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$ because the proofs generally rely on the convergence analysis of the consensus step, which is not impacted by the structure of $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$, as shown in Proposition 14. This proposition extends classical results of consensus theory, to the use of matrices $M \in \mathbb{R}^{nd\times nd}$ that do not have the Kronecker structure of $(W \otimes I_d)$. When $d = 1$, we have $M = W \in \mathbb{R}^{n\times n}$ and no extension is needed.

▶ **Proposition 14.** *Let $M \in \mathcal{M}_{(\lambda^-,\lambda^+)} \subset \mathbb{R}^{nd\times nd}$ with $\lambda^-, \lambda^+ \in (-1,1)$ and $\mathbf{x}^0 \in \mathbb{R}^{nd}$ a vector stacking the variables $x_i^0 \in \mathbb{R}^d$ of each agent. If the vector $\mathbf{x}^0$ is updated with a series of general matrix steps $\mathbf{x}^{k+1} = M\mathbf{x}^k$, then the sequence of vectors $\mathbf{x}^k$ converges to the agent average consensus vector, i.e., a vector stacking n times the agent average $\bar{x}^0 = \frac{1}{n}\sum_{i=1}^n x_i^0$:*

$$\lim_{k\to\infty} \mathbf{x}^k = \mathbf{1}_n \otimes \bar{x}^0.$$

Proof. Since $M \in \mathcal{M}_{(\lambda^-,\lambda^+)}$, the matrix admits the following spectral decomposition:

$$M = V\Sigma V^T, \qquad \text{with} \qquad V_{1,\dots,d} = (\mathbf{1}_n \otimes I_d)/\sqrt{n}, \tag{26}$$

where V is the orthogonal matrix of eigenvectors of M and Σ the diagonal matrix of eigenvalues of M. We define a new variable $\mathbf{z}^k = V^T\mathbf{x}^k$, for which the matrix step $\mathbf{x}^{k+1} = M\mathbf{x}^k$ becomes

$$\mathbf{z}^{k+1} = \Sigma\mathbf{z}^k.$$

The d first eigenvalues in Σ are associated with eigenvectors $V_{1,\dots,d}$ and are equal to 1; all the others are smaller than 1 in absolute value. Therefore,

$$\lim_{k\to\infty} \mathbf{z}^k = \lim_{k\to\infty} \Sigma^k\mathbf{z}^0 = \begin{bmatrix} z_1^0 & \dots & z_d^0 & 0 & \dots & 0 \end{bmatrix}^T = \widetilde{\mathbf{z}}^0,$$

where vector $\widetilde{\mathbf{z}}^0$ contains the d first entries of $\mathbf{z}^0$ and $(n-1)d$ zeros. Thus, the sequence $\mathbf{x}^k$ converges to $V\widetilde{\mathbf{z}}^0$ which is the combination of the columns $V_{1,\dots,d}$ with coefficients given by the d first entries of $\mathbf{z}^0$, i.e.,

$$\lim_{k\to\infty} \mathbf{x}^k = \lim_{k\to\infty} V\mathbf{z}^k = V\widetilde{\mathbf{z}}^0 = V_{1,\dots,d}\ \mathbf{z}^0_{1,\dots,d}.$$

Finally, by definition of variable $\mathbf{z}$, we have $\mathbf{z}^0 = V^T\mathbf{x}^0$, meaning that the d first entries of $\mathbf{z}^0$ are given by $\mathbf{z}^0_{1,\dots,d} = V^T_{1,\dots,d}\mathbf{x}^0$, and therefore:

$$\lim_{k\to\infty} \mathbf{x}^k = V_{1,\dots,d}V^T_{1,\dots,d}\mathbf{x}^0 = \frac{1}{n}(\mathbf{1}_n\mathbf{1}_n^T \otimes I_d)\mathbf{x}^0 = \mathbf{1}_n \otimes \bar{x}^0,$$

where the second equality holds by definition of $V_{1,\dots,d}$ (26). ◀

2.3 Section summary: agent-dependent PEP

In summary, in this section, we have seen that the worst-case performance of t iterations of a decentralized optimization algorithm from $\mathcal{A}_D$ can be computed using an SDP reformulation of the Performance Estimation Problem (PEP) with a size growing with t and the number of agents n (see Proposition 5). When the averaging matrix W is given, the resulting PEP worst-case bound is tight. When computing a worst-case valid for all averaging matrices from $\mathcal{W}_{(\lambda^-,\lambda^+)}$ (symmetric, generalized doubly-stochastic and with a range on non-principal eigenvalues), the resulting PEP is a relaxation, which may provide non-tight performance bounds. We have proved that some form of relaxation is inevitable if we want to obtain a convex formulation (see Theorem 7). The constraints we provide to represent the consensus steps in PEP allow for more general matrix steps that can mix components of vectors to which they apply (see Theorem 12). While this is not natural, it does not alter the convergence of pure consensus, and we do not expect it to be an important source of conservatism in performance limits. (see Proposition 14). Moreover, the tightness of the PEP solutions can be verified a posteriori.

In the rest of the paper, we will exploit symmetries of agents in this agent-dependent PEP formulation to obtain equivalent PEP formulations whose size is independent of the number of agents n. We also characterize conditions under which the resulting worst-case is also independent of n.

3 Equivalence classes of agents in PEP

To exploit agent symmetries in the agent-dependent PEP, let us define an equivalence relation between agents, along with the corresponding equivalence classes. These will be used to build our new compact PEP formulation whose size depends only on the number of equivalence classes.

Let $\mathbf{f}$ and $G = P^TP$ be a solution of an agent-dependent PEP, described in Section 2. We can order their elements to regroup them depending on the agent to which they are associated:

$$\mathbf{f} = \begin{bmatrix} f_1^T & \dots & f_n^T \end{bmatrix}, \quad \text{where } f_i = \begin{bmatrix} f_i^k \end{bmatrix}_{k\in I} \in \mathbb{R}^q \text{ is a vector with the } q \text{ function values of agent } i, \tag{27}$$

Similarly, we write $P \in \mathbb{R}^{d \times np}$ as follows

$$P = \begin{bmatrix} P_1 & \dots & P_n \end{bmatrix} \quad \text{where } P_i \in \mathbb{R}^{d \times p} \text{ contains the } p \text{ vector variables related to agent } i, \tag{28}$$

e.g. $P_i = \begin{bmatrix} y_i^k & x_i^k & g_i^k \end{bmatrix}_{k \in I}$, and the Gram matrix $G \in \mathbb{R}^{np \times np}$ is thus defined as

$$G = P^T P = \begin{bmatrix} P_1^T P_1 & P_1^T P_2 & \dots \\ P_2^T P_1 & \ddots & \\ \vdots & & P_n^T P_n \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & \dots \\ G_{21} & \ddots & \\ \vdots & & G_{nn} \end{bmatrix}. \tag{29}$$

Therefore, diagonal blocks G_{ii} are symmetric and off-diagonal blocks are such that $G_{ij} = G_{ji}^T$. We assume that each block P_i (and f_i) contains the same type and number of variables for each agent i. The variables common to all agents, such as x^*, are copied into each agent block P_i. This also covers the case where agents are heterogeneous and hold different types or numbers of variables since we can always add variables as columns of P_i (and f_i), even if agent i does not use them. Moreover, each agent block P_i (and f_i) has the same column order, enabling us to use the same coefficient vectors for column selection in each agent block.

▶ Notation (Coefficient vector $e_x \in \mathbb{R}^p$). A coefficient vector for a variable x is denoted $e_x \in \mathbb{R}^p$, and contains linear coefficients selecting the correct combination of columns in $P_i \in \mathbb{R}^{d \times p}$ to obtain vector $x_i \in \mathbb{R}^d$, i.e. $P_i e_x = x_i$, for any $i = 1, \dots, n$. This notation allows, for example, to write $x_i^T x_i$ as $x_i^T x_i = e_x^T P_i^T P_i e_x = e_x^T G_{ii} e_x$, for any agent i. Similarly, let $e_{f(x)}$ be a vector of coefficients selecting the correct element in the vector f_i such that $e_{f(x)}^T f_i = f_i(x_i)$. These coefficient vectors will be used to explicit our new PEP formulations in Sections 4 and 5, as well as in Appendix A.

Using this agent block notation for $\mathbf{f}$ and G, we can define permutations of agents in a PEP solution, leading to the definition of an equivalence relation between agents in performance estimation problems (Definition 15 below).

▶ Notation (Permuted solutions). For any feasible solution $(\mathbf{f}, G)$ of an agent-dependent PEP formulation (8), written in the form of (27) and (29), the solution obtained after permutation of agents $i, j \in \mathcal{V}$ in $\mathbf{f}$ and G is denoted $(\mathbf{f}_{\Pi_{ij}}, G_{\Pi_{ij}})$.

▶ Example (Permuted solutions). Let $(\mathbf{f}, G)$ be a feasible PEP solution written as

$$\mathbf{f} = \begin{bmatrix} f_1^T & f_2^T & \dots & f_n^T \end{bmatrix} \qquad G = P^T P \qquad \text{with} \qquad P = \begin{bmatrix} P_1 & P_2 & \dots & P_n \end{bmatrix}.$$

Agents 1 and 2 can be swapped to obtain a permuted solution $(\mathbf{f}_{\Pi_{12}}, G_{\Pi_{12}})$:

$$\mathbf{f}_{\Pi_{12}} = \begin{bmatrix} f_2^T & f_1^T & \dots & f_n^T \end{bmatrix} \qquad G_{\Pi_{12}} = P_{\Pi_{12}}^T P_{\Pi_{12}} \qquad \text{with} \qquad P_{\Pi_{12}} = \begin{bmatrix} P_2 & P_1 & \dots & P_n \end{bmatrix}.$$

▶ **Definition 15** (Equivalent agents). *Let $i, j \in \mathcal{V}$ be two distinct agents. We define an equivalence relation between the two agents $i \sim j$ if, for any feasible solution $(\mathbf{f}, G)$ (27)–(29) of an agent-dependent PEP formulation (8), the permuted solution $(\mathbf{f}_{\Pi_{ij}}, G_{\Pi_{ij}})$, with agent i and j swapped, is feasible for the PEP and provides the same performance value,*

$$\mathcal{P}(\mathbf{f}, G) = \mathcal{P}(\mathbf{f}_{\Pi_{ij}}, G_{\Pi_{ij}}).$$

Intuitively, this means that equivalent agents have exactly the same role in the algorithm and the performance setting. By definition, this relation is reflexive, symmetric, and transitive, and thus corresponds well to an equivalence relation, which induces a partition into equivalence classes.

▶ **Definition 16** (Equivalence classes of agents). *Let $\mathcal{V}$ be a set of n agents. The equivalence relation from Definition 15 implies a partition $\mathcal{T}$ of set $\mathcal{V}$ into r equivalence classes (or subsets) of agents: $\mathcal{T} = \{\mathcal{V}_1, \dots, \mathcal{V}_r\}$. Each class only contains agents that are equivalent to each other. The size of each class $\mathcal{V}_u$ is denoted n_u $(u = 1, \dots, r)$. We also denote the equivalence class of a given agent i by $\mathcal{V}_{u_i}$.*

Proposition 17 proves the existence of PEP solutions for which equivalent agents have equal blocks in the solution. This will be the building block of our compact PEP formulations in Sections 4 and 5.

▶ **Proposition 17** (Existence of symmetric solutions in PEP)**.** *Let* $\mathbf{f} \in \mathbb{R}^{nq}$ *and* $G \in \mathbb{R}^{np\times np}$ *be any feasible solution of an agent-dependent (and LMI Gram-representable) PEP formulation for distributed optimization* (8)*; and* $\mathcal{T} = \{\mathcal{V}_1, \dots, \mathcal{V}_r\}$ *be the partition of* $\mathcal{V}$ *induced by the equivalence relation from Definition 15. There is a symmetrized agent-class solution* $(\mathbf{f}^s_{\mathcal{T}}, G^s_{\mathcal{T}})$*, with equal blocks for equivalent agents, providing another valid solution for the PEP,*

$$\mathbf{f}^s_{\mathcal{T}} = \left[(f^s_1)^T \dots (f^s_n)^T\right] \text{ with } \quad f^s_i = f^{u_i}_A := \frac{1}{n_{u_i}} \sum_{k\in\mathcal{V}_{u_i}} f_k, \qquad \text{for all } i \in \mathcal{V}, \tag{30}$$

$$G^s_{\mathcal{T}} = \begin{bmatrix} G^s_{11} & G^s_{12} & \dots \\ G^s_{21} & \ddots & \\ \vdots & & G^s_{nn} \end{bmatrix} \text{ with } G^s_{ij} = \begin{cases} G^{u_i}_A := \frac{1}{n_{u_i}} \sum_{k\in\mathcal{V}_{u_i}} G_{kk} & \text{for } i = j, \\ G^{u_i}_B := \frac{1}{n_{u_i}(n_{u_i}-1)} \sum_{k\in\mathcal{V}_{u_i}} \sum_{\substack{l\in\mathcal{V}_{u_i} \\ l\neq k}} G_{kl} & \text{for } j \neq i, j \in \mathcal{V}_{u_i}, \\ G^{u_i u_j}_E := \frac{1}{n_{u_i}} \sum_{k\in\mathcal{V}_{u_i}} G_{kj}, & \text{for } j \neq i, j \in \mathcal{V}\setminus\mathcal{V}_{u_i}, \end{cases} \tag{31}$$

Moreover, this symmetrized solution $(\mathbf{f}^s_{\mathcal{T}}, G^s_{\mathcal{T}})$ *provides the same performance value as* $(\mathbf{f}, G)$

$$\mathcal{P}(\mathbf{f}^s_{\mathcal{T}}, G^s_{\mathcal{T}}) = \mathcal{P}(\mathbf{f}, G).$$

Proof. By definition, any permutation of equivalent agents $\Pi \in \mathbb{R}^{n\times n}$ applied on a given PEP solution $\mathbf{f}$, G provides another valid PEP solution, with the same objective value:

$$\mathbf{f}_\Pi = \mathbf{f}(\Pi \otimes I_q), \qquad P_\Pi = P(\Pi \otimes I_p), \qquad G_\Pi = P_\Pi^T P_\Pi = (\Pi \otimes I_p)^T G (\Pi \otimes I_p).$$

A permutation of agents corresponds to a permutation of (sets of) columns and then the permutation matrix multiplies $\mathbf{f}$ and P on the right. The Kronecker products $\otimes$ are used to permute all the variables related to the same agent at the same time.

Since the PEP problem is LMI Gram-representable, its objective is linear in $\mathbf{f}$ and G and the combination of PEP solutions with the same objective value gives another PEP solution with the same objective value. We can thus construct a symmetrized PEP solution with the same objective value as $(\mathbf{f}, G)$ by averaging all the possible permuted solutions $(\mathbf{f}_\Pi, G_\Pi)$. Each permutation exchanges only agents from the same equivalence class, and therefore, the average applies to each class separately:

$$f^s_i = \frac{(n_{u_i}-1)!}{n_{u_i}!} \sum_{k\in\mathcal{V}_{u_i}} f_k = \frac{1}{n_{u_i}} \sum_{k\in\mathcal{V}_{u_i}} f_k \ := f^{u_i}_A, \qquad \text{for all } i \in \mathcal{V},$$

$$G^s_{ii} = \frac{(n_{u_i}-1)!}{n_{u_i}!} \sum_{k\in\mathcal{V}_{u_i}} G_{kk} = \frac{1}{n_{u_i}} \sum_{k\in\mathcal{V}_{u_i}} G_{kk} \ := G^{u_i}_A \qquad \text{for all } i \in \mathcal{V},$$

$$G^s_{ij} = \frac{(n_{u_i}-2)!}{n_{u_i}!} \sum_{k\in\mathcal{V}_{u_i}} \sum_{\substack{l\in\mathcal{V}_{u_i} \\ l\neq k}} G_{kl} = \frac{1}{n_{u_i}(n_{u_i}-1)} \sum_{k\in\mathcal{V}_{u_i}} \sum_{\substack{l\in\mathcal{V}_{u_i} \\ l\neq k}} G_{kl} \ := G^{u_i}_B \qquad \text{for all } i \in \mathcal{V} \text{ and } j \neq i,\ j \in \mathcal{V}_{u_i},$$

$$G^s_{ij} = \frac{(n_{u_i}-1)!}{n_{u_i}!} \sum_{k\in\mathcal{V}_{u_i}} G_{ik} = \frac{1}{n_{u_i}} \sum_{k\in\mathcal{V}_{u_i}} G_{kj} \ := G^{u_i u_j}_E \qquad \text{for all } i \in \mathcal{V} \text{ and } j \neq i,\ j \in \mathcal{V}\setminus\mathcal{V}_{u_i},$$

◀

▶ **Definition 18** (Blocks f^u_A, G^u_A, G^u_B and G^{uv}_E)**.** *The blocks repeating in* $\mathbf{f}^s$ (30) *are denoted by* f^u_A*, for each equivalence class* $u = 1, \dots, r$*. For the symmetrized Gram matrix* G^s (31)*, we denote the three types of blocks composing it by* G^u_A*,* G^u_B*, and* G^{uv}_E*, for all equivalence classes* $u, v = 1, \dots, r$*.*

- *Blocks* $G^u_A \in \mathbb{R}^{p\times p}$ *(for* $u = 1, \dots, r$*) correspond to the blocks of the Gram matrix containing the scalar products between the variables of one agent from* $\mathcal{V}_u$*. These blocks lie on the diagonal of* G^s (31) *and are symmetric by definition.*
- *Blocks* $G^u_B \in \mathbb{R}^{p\times p}$ *(for* $u = 1, \dots, r$*) correspond to the blocks of the Gram matrix containing the scalar products between variables of two different agents from the same class* $\mathcal{V}_u$*. These blocks are symmetric because they can be written as a sum of symmetric matrices* $G^u_B = \frac{1}{n_u(n_u-1)} \sum_{k,l\in\mathcal{V}_u, k>l} (G_{kl} + G^T_{lk})$*. Moreover, they are only defined for classes with at least 2 agents (*$n_u \geq 2$*).*
- *Blocks* $G^{uv}_E \in \mathbb{R}^{p\times p}$ *(for* $u \neq v = 1, \dots, r$*) correspond to the blocks of the Gram matrix containing the scalar products between variables of two agents from different classes* $\mathcal{V}_u$ *and* $\mathcal{V}_v$*. These blocks are non-symmetric and are only defined when there are at least 2 different classes of agents (*$r \geq 2$*).*

Therefore, we can solve the agent-dependent PEP problem, described in Section 2, restricted to symmetric solutions of the form of Proposition 17, without impacting the resulting worst-case value. The symmetry in the solutions allows working with a limited number of variables f_A^u, G_A^u, G_B^u, and G_E^{uv} in PEP, depending only on the (smaller) number of equivalent classes of agents r. This requires reformulating all the elements of the agent-dependent PEP (objective and constraints) in terms of these variables. Section 4 shows how to perform such a reformulation when all the agents are equivalent in the PEP, which occurs in many usual settings. This leads to a PEP formulation with only one equivalence class and hence whose size is independent of the number of agents. Section 5 generalizes reformulation techniques from Section 4 to a general number of equivalence classes.

4 Agent-independent PEP formulation when all agents are equivalent

This section focuses on cases where all agents are equivalent.

▶ **Assumption 19** (Equivalent Agents)**.** *All the agents are equivalent in the Performance Estimation Problem applied to a distributed algorithm, in the sense of Definition 15. Therefore, there is only one equivalence class which is* $\mathcal{V}$*.*

In other words, this assumption means that all the agents can be permuted in a PEP solution without impacting its worst-case value and thus no agent plays a specific role in the algorithm or its performance evaluation. This assumption is satisfied in many usual performance evaluation settings $\mathcal{S}_n$, for which all agents play an identical role in the algorithm and its performance evaluation. Under Assumption 19, the symmetrized solution from Proposition 17 only has a few different blocks that repeat.

▶ **Corollary 20** (Fully symmetric PEP solutions)**.** *When all the agents* ($n \geq 2$) *are equivalent, see Assumption 19, any solution* $(\mathbf{f}, G)$ *of an agent-dependent (and LMI Gram-representable) PEP formulation for distributed optimization* (8) *can be fully symmetrized, without impacting its worst-case value:*

$$\mathbf{f}^s = \begin{bmatrix} f_A^T \dots f_A^T \end{bmatrix} \qquad \text{with} \quad f_A = \frac{1}{n}\sum_{i=1}^{n} f_i \ \in \mathbb{R}^q, \tag{32}$$

$$G^s = \begin{bmatrix} G_A & G_B & \dots \\ G_B & \ddots & \\ \vdots & & G_A \end{bmatrix} \quad \text{with} \quad G_A = \frac{1}{n}\sum_{i=1}^{n} G_{ii} \ \in \mathbb{R}^{p\times p}, \quad \text{and } G_B = \frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j\neq i}^{n} G_{ij} \ \in \mathbb{R}^{p\times p}. \tag{33}$$

Proof. The results follow from applying Proposition 17 with $\mathcal{V}$ as the only equivalence class. Since there is only one equivalence class of agents, the Gram matrix only has two types of blocks, as explained in Definition 18. ◀

In this section, we consider the agent-dependent PEP formulation from Section 2, with a given class of averaging matrix, and we constrain its solution to be fully symmetric, as expressed in (32) and (33). First, we explain how and when this leads to a compact PEP formulation with size independent of the number of agents. Secondly, we describe the implications of a fully symmetrized PEP solution on the worst-case local iterates and functions.

4.1 The PEP formulation restricted to fully symmetric solutions

When all agents are equivalent (Assumption 19), Corollary 20 shows that restricting PEP to fully symmetric solutions does not impact the value of the worst-case. We will see in Theorem 23 that it allows writing the PEP in a compact form since the performance measure and the constraints can be expressed only in terms of the smaller blocks f_A, G_A, and G_B that are repeating in $\mathbf{f}^s$ and G^s. Theorem 23 states sufficient conditions for which a PEP for distributed optimization can be made compact (i.e. with size independent of n) and for which it provides a worst-case value independent of the number of agents n.

Before stating the theorem, we introduce two types of expressions or constraints that we can allow in PEP to obtain agent-independent worst-case values. Let us consider a simple example to motivate the first type of expressions.

▶ Example. If we consider a PEP for distributed optimization with the initial conditions $f_i(x_i^0) - f_i(x^*) \leq 1$ for all $i \in \mathcal{V}$ and the performance measure $\mathcal{P} = \sum_{i=1}^n (f_i(x_i^0) - f_i(x^*))$, then the worst-case value would scale with the number of agents n in the problem: if we consider more agents, the maximum value of $\mathcal{P}$ will increase. To obtain a PEP that provides worst-case values independent of n, the performance measure needs to be an expression that is invariant to the number of agents, $\frac{1}{n}\sum_{i=1}^n (f_i(x_i^0) - f_i(x^*))$ in our example. We propose a notion called scale-invariant, defined below.

▶ **Definition 21** (Scale-invariant expression and constraint)**.** *In a distributed system with n agents, we call a Gram-representable expression h scale-invariant if it can be written as a linear combination $h = \sum_j c_j h_j$, where coefficients c_j are independent of n and terms h_j are any of these three forms, for any variables x_i and y_j assigned to agents i and j, including the case $x_i = y_j$:*

a. $\frac{1}{n}\sum_{i=1}^n f_i(x_i)$, *the average of function values,*

b. $\frac{1}{n}\sum_{i=1}^n x_i^T y_i$, *the average of scalar products between two variables of the same agent,*

c. $\frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n x_i^T y_j$, *the average over the n^2 pairs of agents of the scalar products between two variables, each assigned to any agent.*

A Gram representable constraint $h \leq D$, for $D \in \mathbb{R}$, is scale-invariant if the expression h is scale-invariant.

▶ Remark. The name *scale-invariant* comes from the fact that, if we duplicate each agent of the system, with its local variables and its local function, then the value of a scale-invariant expression is unchanged. For example, the following expressions can be shown, possibly after algebraic manipulations, to be scale-invariant:

$$\frac{1}{n}\sum_{i=1}^n (f_i(\bar{x}) - f_i(x^*)), \qquad \frac{1}{n}\sum_{i=1}^n \|x_i - x^*\|^2, \qquad \frac{1}{n}\sum_{i=1}^n \|x_i - \bar{x}\|^2, \qquad \left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(x_i)\right\|^2.$$

The variables common to all agents, such as $\bar{x}$ or x^*, can indeed be used in scale-invariant expressions because they are assumed to be copied in each agent set of variables:

$$\bar{x}_i = \bar{x} = \frac{1}{n}\sum_{j=1}^n x_j \quad \text{for all } i \in \mathcal{V} \qquad \text{and} \qquad x_i^* = x^* \quad \text{for all } i \in \mathcal{V}.$$

These constraints can then be written with scale-invariant expressions. See Appendix A for details.

Let us consider another simple example to motivate the second type of expressions we define.

▶ Example. If we consider a PEP for distributed optimization with the initial conditions $\|x_i^0 - x^*\|^2 \leq n$ for all $i \in \mathcal{V}$ and the performance measure $\mathcal{P} = \frac{1}{n}\sum_{i=1}^n \|x_i^0 - x^*\|^2$, then the worst-case value would scale with the number of agents n in the problem: if we consider more agents, the maximum value of $\mathcal{P}$ will increase. To obtain a PEP that provides worst-case values independent of the number of agents n, the initial constraints involving only one agent at a time need to be independent of n, $\|x_i^0 - x^*\|^2 \leq 1$ in this example. We propose the notion of single-agent constraint, defined below.

▶ **Definition 22** (Single-agent expression and constraint)**.** *In a distributed system with n agents, we call an expression h single-agent if it can be written as a linear combination $h = \sum_k c_k h_k(i)$, where coefficients c_k are independent of n and all terms $h_k(i)$ only involve the local variables and the local function of a single agent i. A constraint $h \leq D$, for $D \in \mathbb{R}$, is single-agent if the expression h is single-agent.*

▶ Remark. When all the agents are equivalent in the PEP, they all share the same single-agent constraints. Here are examples of single-agent constraints

$$\|x_i - x^*\|^2 \leq 1, \quad \text{for all } i \in \mathcal{V}, \qquad \|\nabla f_i(x_i)\|^2 \leq 1, \quad \text{for all } i \in \mathcal{V}.$$

We can now state Theorem 23 which characterizes the settings where the worst-case value $w(\mathcal{S}_n)$ can be computed (i) in a compact manner for all $n \geq 2$ and (ii) independently of the value of n. We remind that $w(\mathcal{S}_n = \{n, \mathcal{A}, t, \mathcal{P}, \mathcal{F}, \mathcal{I}, \mathcal{W}\})$ denotes the worst-case performance of the execution of t iterations of distributed algorithm $\mathcal{A}$ with n agents, with respect to the performance criterion $\mathcal{P}$, valid for any starting point satisfying the set of initial conditions $\mathcal{I}$, any local functions in the set of functions $\mathcal{F}$ and any averaging matrix in the set of matrices $\mathcal{W}$.

▶ **Theorem 23** (Agents-independent worst-case performance). *Let $n \geq 2$ and $\mathcal{W} = \mathcal{W}_{(\lambda^-,\lambda^+)}$ with $\lambda^- \leq \lambda^+ \in (-1,1)$. We assume that the algorithm $\mathcal{A} \in \mathcal{A}_D$, the performance measure $\mathcal{P}$, the initial conditions $\mathcal{I}$, and the interpolation constraints for $\mathcal{F}$ are linearly (or LMI) Gram-representable. If Assumption 19 holds, meaning that all the agents are equivalent in the PEP, then*

1. *The computation of $w(\mathcal{S}_n)$ can be formulated as a compact SDP PEP, with $f_A \in \mathbb{R}^q$, $G_A \in \mathbb{R}^{p\times p}$ and $G_B \in \mathbb{R}^{p\times p}$ as variables and whose size is independent of the number of agents n.*
2. *If, in addition, the performance criterion $\mathcal{P}$ is scale-invariant and the set of initial conditions $\mathcal{I}$ only contains single-agent constraints, applied to every agent $i \in \mathcal{V}$, and scale-invariant constraints, then the resulting SDP PEP problem and its worst-case value $w(\mathcal{S}_n)$ are fully independent of the number of agents n.*

▶ **Corollary 24.** *Under the conditions stated in part 2 of Theorem 23, a general worst-case guarantee, valid for any $n \geq 2$ (including $n \to \infty$), can be obtained using $n = 2$:*

$$w(\mathcal{S}_n) = w(\mathcal{S}_2) \qquad \text{for all } n \geq 2.$$

▶ Remark. The conditions of part 2 of Theorem 23, which guarantee agent-independent worst-case performance, are satisfied for many usual settings, as illustrated in Appendix A.

To prove Theorem 23, we rely on Lemma 25 which gives a way of imposing the SDP condition $G^s \succeq 0$ solely based on its block matrices G_A and G_B.

▶ **Lemma 25.** *Let $G^s \in \mathbb{R}^{np\times np}$ be a fully symmetrized Gram matrix, as defined in (33), for $n \geq 2$. The SDP constraint $G^s \succeq 0$ can be expressed with $G_A \in \mathbb{R}^{p\times p}$ and $G_B \in \mathbb{R}^{p\times p}$:*

$$G^s \succeq 0 \qquad \Longleftrightarrow \qquad G_A + (n-1)G_B \succeq 0 \quad \text{and} \quad G_A - G_B \succeq 0. \tag{34}$$

Proof. We will use the fact that

$$G^s \succeq 0 \qquad \Longleftrightarrow \qquad z^T G^s z \geq 0 \quad \text{for all } z \in \mathbb{R}^{np}.$$

Let us separate the vector $z \in \mathbb{R}^{np}$ into n sub-vectors z_i $(i = 1,\dots,n)$ that can each be written as the sum of an average part $\bar{z} \in \mathbb{R}^p$ and a centered one $z_i^\perp$:

$$z_i = \bar{z} + z_i^\perp, \qquad \text{where } \bar{z} = \frac{1}{n}\sum_{i=1}^n z_i \text{ and } \sum_{i=1}^n z_i^\perp = 0.$$

Using this decomposition of z, together with the definition of G^s (33), we can express $z^T G^s z$ as follows

$$\begin{bmatrix} \bar{z}+z_1^\perp \\ \vdots \\ \bar{z}+z_n^\perp \end{bmatrix}^T \begin{bmatrix} G_A & G_B & \dots \\ G_B & \ddots & \\ \vdots & & G_A \end{bmatrix} \begin{bmatrix} \bar{z}+z_1^\perp \\ \vdots \\ \bar{z}+z_n^\perp \end{bmatrix} = \sum_{i=1}^n\sum_{j=1}^n (\bar{z}+z_i^\perp)^T G_B (\bar{z}+z_j^\perp) + \sum_{i=1}^n (\bar{z}+z_i^\perp)^T (G_A - G_B)(\bar{z}+z_i^\perp),$$

$$\text{Since } \sum_{i=1}^n z_i^\perp = 0, \qquad = n^2 \bar{z}^T (G_B)\bar{z} + n\bar{z}^T(G_A - G_B)\bar{z} + \sum_{i=1}^n (z_i^\perp)^T (G_A - G_B)(z_i^\perp),$$

$$= \bar{z}^T (nG_A + n(n-1)G_B)\bar{z} + \sum_{i=1}^n (z_i^\perp)^T (G_A - G_B)(z_i^\perp).$$

Therefore, $z^T G^s z \geq 0$ for all $z \in \mathbb{R}^{np}$ when

$$\bar{z}^T (nG_A + n(n-1)G_B)\bar{z} + \sum_{i=1}^n (z_i^\perp)^T (G_A - G_B)(z_i^\perp) \geq 0, \quad \text{for all } \bar{z}, z_i^\perp \in \mathbb{R}^p \text{ such that } \sum_{i=1}^n z_i^\perp = 0. \tag{35}$$

Conditions $G_A + (n-1)G_B \succeq 0$ and $G_A - G_B \succeq 0$ are sufficient to ensure that (35) is satisfied and hence that $G^s \succeq 0$. To show the necessity of these conditions, let us consider different cases for $\bar{z}$ and $z_i^\perp$:

- When $z_i^\perp = 0$ for all i, then (35) simply imposes that $\bar{z}^T(nG_A + n(n-1)G_B)\bar{z} \geq 0$ for all $\bar{z} \in \mathbb{R}^p$, which is equivalent to $G_A + (n-1)G_B \succeq 0$.
- When $\bar{z} = 0$, $z_1^\perp = -z_2^\perp = s$, for an arbitrary $s \in \mathbb{R}^p$, and $z_i^\perp = 0$ for $i = 3,\dots,n$, then (35) imposes that $s^T(G_A - G_B)s \geq 0$ for all $s \in \mathbb{R}^p$, which is equivalent to $G_A - G_B \succeq 0$. ◀

Proof of Theorem 23 (part 1). As detailed in Section 2.2, the set of averaging matrices $\mathcal{W}_{(\lambda^-,\lambda^+)}$ has Gram-representable necessary interpolation constraints, see Corollary 13. Since $\mathcal{A}, \mathcal{P}, \mathcal{I}$ and interpolation constraints for $\mathcal{F}$ are also Gram-representable, Proposition 5 guarantees that the computation of $w(n, \mathcal{A}, t, \mathcal{P}, \mathcal{I}, \mathcal{F}, \mathcal{W}_{(\lambda^-,\lambda^+)})$ can be formulated as an agent-dependent SDP PEP problem (8). Moreover, since the agents are equivalent, we can restrict to fully symmetric solutions without impacting the worst-case value (see Corollary 20). Hence, all the PEP components can be written in terms of $\mathbf{f}^s$ and G^s, which are composed of blocks f_A, G_A, and G_B. Lemma 25 shows how the SDP constraint $G^s \succeq 0$, can be expressed in terms of G_A and G_B. The other elements of the PEP can directly be expressed in terms of f_A, G_A, and G_B, using the definition of $\mathbf{f}^s$ and G^s (32) (33), as detailed in Appendix A. Since all the PEP elements can be expressed in terms of blocks $f_A \in \mathbb{R}^q$, $G_A \in \mathbb{R}^{p\times p}$ and $G_B \in \mathbb{R}^{p\times p}$, the problem can be made smaller by considering them as the variables of the SDP PEP, instead of the full matrix $G^s \in \mathbb{R}^{np\times np}$ and vector $\mathbf{f}^s \in \mathbb{R}^{nq}$. The size of the problem is thus independent of the number of agents n, but n could still appear as a coefficient in the constraints or the objective of the problem. ◀

Lemma 25 makes the value of n appear as a coefficient, but we can define a change of variables allowing to remove the dependency on n in the new constraints (34). Let us defined the new matrix variables $G_C \in \mathbb{R}^{p\times p}$ as follows:

$$G_C = \frac{1}{n}(G_A + (n-1)G_B). \tag{36}$$

Moreover, we can also express G_C directly based on the blocks of the initial Gram matrix G (29):

$$G_C = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} G_{ij}. \tag{37}$$

While G_A contains the average scalar products between local variables of the same agent, and G_B the average scalar products between local variables of different agents, this new block G_C contains the average scalar products between local variables of any two agents. We call G_C the collective block matrix. By applying this change of variables (36) in Lemma 25, we obtain conditions that are independent of n:

▶ **Lemma 26** (Constraint $G^s \succeq 0$)**.** *Let $G^s \in \mathbb{R}^{np\times np}$ be a symmetric Gram matrix, as defined in* (33)*. The SDP constraint $G^s \succeq 0$ can be expressed with $G_A \in \mathbb{R}^{p\times p}$ and $G_C \in \mathbb{R}^{p\times p}$:*

$$G^s \succeq 0 \qquad \Longleftrightarrow \qquad G_C \succeq 0 \quad \textit{and} \quad G_A - G_C \succeq 0.$$

Proof. The result can be obtained by applying change of variables (36) in the constraints stated in (34). ◀

The new variable G_C allows writing many constraints without any dependence on n. Let us state a lemma that will be used to prove the second part of Theorem 23.

▶ **Lemma 27.** *In an SDP PEP for distributed optimization* (8)*, with equivalent agents (Assumption 19), any scale-invariant Gram-representable expression can be written independently of n, using variables f_A, G_A, and G_C.*

Proof. By Corollary 20, since all the agents are equivalent, we can restrict the PEP to a fully symmetric solution $(\mathbf{f}^s, G^s)$ from (32)–(33). Since the expression is scale-invariant and Gram-representable (Definition 21), it can only combine three types of terms, with coefficients independent of n,

a. The average of function values

$$\frac{1}{n}\sum_{i=1}^{n} f_i(x_i) = \frac{1}{n}\sum_{i=1}^{n} e_{f(x)}^T f_i = e_{f(x)}^T f_A, \qquad \text{for any } j = 1, \dots, n, \tag{38}$$

where $e_{f(x)}$ is the vector of coefficients such that $e_{f(x)}^T f_i = f_i(x_i)$. The second equality holds because $f_i = f_A$ for all i, by definition of the agent-symmetric $\mathbf{f}^s$ (32).

b. The average of scalar products between two variables related to the same agent i:

$$\frac{1}{n}\sum_{i=1}^{n} x_i^T y_i = \frac{1}{n}\sum_{i=1}^{n} e_x^T G_{ii} e_y = e_x^T(G_A)e_y, \tag{39}$$

where e_x (resp. e_y) is the vector of coefficients such that $P_i e_x = x_i$ (resp. $P_i e_y = y_i$), with P_i defined in (28). The second equality holds because $G_{ii} = G_A$, by definition of the agent-symmetric G^s (33).

c. The average over the n^2 pairs of agents of the scalar products between two variables, each related to any agent:

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^T y_j = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} e_x^T G_{ij} e_y = e_x^T (G_C) e_y, \tag{40}$$

where the last equality follows from the definition of G_C (37).

We have shown that the three types of terms a, b, c can be written independently of n with f_A, G_A, and G_C, and so can any Gram-representable and scale-invariant expression. ◀

We are now ready to prove the second part of Theorem 23, about the worst-case value being independent of the number of agents.

Proof of Theorem 23 (part 2). Under the conditions stated in the theorem (part 2), we will show that the SDP-PEP problem can be written, independently of n, with f_A, G_A, and G_C as variables. When all the agents are equivalent in the PEP, we can restrict to fully symmetric solutions (Corollary 20). In that case, Lemma 26 shows that SDP constraints $G^s \succeq 0$ can be expressed with G_A and G_C, independently of n.

Then, we prove that, in the symmetrized PEP, any single-agent constraint (Definition 22) can be written with f_A and G_A, independently of n. Since the PEP components are all Gram-representable, any of its single-agent constraints for agent i linearly combines local function values $f_i(x_i)$ and scalar products of local variables $x_i^T y_i$, with coefficients that are independent of n. Here, x_i and y_i denote any local vector held by agent i. By definition of the symmetric $\mathbf{f}^s$ (32), a local function value term can be expressed with f_A as

$$f_i(x_i) = e_{f(x)}^T f_i = e_{f(x)}^T f_A, \qquad \text{for any } i = 1, \dots, n. \tag{41}$$

By definition of the symmetric Gram G^s (33), a local scalar product term can be expressed with G_A as

$$x_i^T y_i = e_x^T G_{ii} e_y = e_x^T G_A e_y \qquad \text{for any } i = 1, \dots, n, \tag{42}$$

We have shown that the two types of terms (41) and (42) can be written independently of n with f_A, G_A, and so can any Gram-representable single-agent constraint. The resulting constraint, expressed with f_A and G_A, is valid for any agent $i \in \mathcal{V}$. This is consistent with the equivalent agents assumption, which requires that each single-agent constraint is applied similarly to every agent $i \in \mathcal{V}$. The reformulation of single-agent constraints, independently of n, applies to the interpolation constraints for $\mathcal{F}$, the algorithm constraints for $\mathcal{A} \in \mathcal{A}_D$ (except consensus), and part of the initial constraints $\mathcal{I}$.

We now show that the other PEP components, see e.g. (5), can also be expressed independently of n. When agents are equivalent, by Lemma 27, any Gram-representable and scale-invariant expression can be expressed with f_A, G_A, and G_C, independently of n. As explained in the proof of part 1 of the theorem, all the PEP components are Gram-representable. Therefore, we will simply prove that all the PEP components that cannot be expressed as single-agent constraints, treated above, can be written with scale-invariant expressions. The performance measure $\mathcal{P}$ and the rest of the initial conditions $\mathcal{I}$ are scale-invariant, by assumption of the theorem statement. The optimality condition for (1), can be written in a Gram-representable manner as

$$\left\| \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x^*) \right\|^2 = 0,$$

which is well scale-invariant thanks to the $1/n$ factor. Finally, the interpolation constraints for consensus steps with averaging matrices from $\mathcal{W}_{(\lambda^-,\lambda^+)}$ are given by (23), (24) and (25), from Corollary 13. In these constraints, notations hide sums over the agents and can be written with Gram-representable and scale-invariant expressions as

$$\overline{X} = \overline{Y} \quad \iff \quad \left(\frac{1}{n}\sum_{i=1}^{n}(X_i - Y_i)\right)^T \left(\frac{1}{n}\sum_{j=1}^{n}(X_j - Y_j)\right) = 0,$$

$$(Y_\perp - \lambda^- X_\perp)^T (Y_\perp - \lambda^+ X_\perp) \preceq 0 \quad \iff \quad \frac{1}{n}\sum_{i=1}^{n} \big((Y_i - \overline{Y}) - \lambda^-(X_i - \overline{X})\big)^T \big((Y_i - \overline{Y}) - \lambda^+(X_i - \overline{X})\big) \preceq 0,$$

$$X_\perp^T Y_\perp - Y_\perp^T X_\perp = 0 \quad \iff \quad \frac{1}{n}\sum_{i=1}^{n}(X_i - \overline{X})^T (Y_i - \overline{Y}) - \frac{1}{n}\sum_{i=1}^{n}(Y_i - \overline{Y})^T (X_i - \overline{X}) = 0.$$

where matrices $X_i, Y_i \in \mathbb{R}^{d\times t}$ contain the different consensus iterates $x_i^k, y_i^k \in \mathbb{R}^d$ $(k = 1, \dots, t)$ of agent i as columns. The two last constraints have been multiplied by $1/n$ to obtain scale-invariant expressions. This does not alter these constraints that have left-hand sides equal to zero.

In the end, all the PEP components can well be expressed independently of n, with $f_A \in \mathbb{R}^q$, $G_A \in \mathbb{R}^{p\times p}$ and $G_C \in \mathbb{R}^{p\times p}$. We refer to Appendix A for explicit expressions of different possible PEP constraints and objectives. Therefore, the SDP PEP can be made smaller and fully independent of the number of agents n, by considering these blocks as variables of the problem, instead of the full matrix $G^s \in \mathbb{R}^{np\times np}$ and vector $\mathbf{f}^s \in \mathbb{R}^{nq}$. ◀

4.2 Impact of agent symmetry on the worst-case solution

By definition of the fully symmetric PEP solution (32)–(33), in which the Gram matrix has equal diagonal blocks G_A and equal off-diagonal blocks G_B, we can prove symmetry results for the worst-case iterates and functions of the agents. These symmetries open up perspectives with implications for understanding, analyzing and designing distributed algorithms yet to be discovered.

▶ **Proposition 28.** *If all the agents are equivalent in the agent-dependent PEP (Assumption 19), then there is a worst-case instance such that,*

- *the worst-case sequences of iterates of the agents $x_i^0, \dots, x_i^t \in \mathbb{R}^d$ $(i = 1, \dots, n)$ are unitary transformations, i.e. rotations or reflections, of each other:*

  $$x_i^k = R_i x_1^k \qquad \text{with } R_i \in \mathbb{R}^{d\times d} \text{ such that } R_i^T R_i = I, \qquad \text{for all } i \in \mathcal{V} \text{ and all } k.$$

- *the worst-case local functions have equal values at their own iterates $f_i(x_i^k) = f_j(x_j^k)$ (for all $i, j \in \mathcal{V}$ and all k), and can be chosen identical up to a unitary transformation of variables:*

  $$f_i(x) = f_1(R_i^T x) \qquad \text{for all } i \in \mathcal{V}.$$

▶ Remark. Proposition 28 is stated for the worst-case iterates, but is also valid for any linear combinations of columns of P_i (28). Indeed, as detailed in the proof below, we have $P_i = R_i P_1$, with $R_i \in \mathbb{R}^{d\times d}$ such that $R_i^T R_i = I$, for all $i = 1, \dots, n$. Moreover, the result is also valid for more than one agent at a time and can be written for any groups of agents, e.g. for groups of two agents, we have:

$$[P_i \ P_j] = \widetilde{R}_{ij}[P_1 \ P_2] \qquad \text{with } \widetilde{R}_{ij} \in \mathbb{R}^{d\times d} \text{ such that } \widetilde{R}_{ij}^T \widetilde{R}_{ij} = I, \qquad \text{for all } i, j \in \mathcal{V} \text{ and all } k.$$

This holds by definition of the fully symmetrized Gram matrix G^s (33) which has equal diagonal blocks: $G_{ii} = G_A$ for all $i \in \mathcal{V}$, but also equal off-diagonal blocks: $G_{ij} = G_B$ for $i \neq j \in \mathcal{V}$.

Proof. By Corollary 20, when the agents are all equivalent, there is a worst-case solution of the agent-dependent PEP, which is fully symmetric: $(\mathbf{f}, G) = (\mathbf{f}^s, G^s)$. When the solution is agent-symmetric, by definition of G^s (33), we have

$$G_{ii} = P_i^T P_i = P_j^T P_j = G_{jj} \quad \text{for all } i, j \in \mathcal{V},$$

which imposes that matrices of agent variables P_i are all unitary transformations of each other. Without loss of generality, we can express all the P_i depending on P_1:

$$P_i = R_i P_1 \quad \text{with } R_i^T R_i = I, \qquad \text{for all } i \in \mathcal{V}.$$

Matrices R_i are not especially unique. This unitary transformation relation is valid for any column or combination of columns of P_i and P_1, so we have it for iterates and gradient vectors of the agents:

$$x_i^k = R_i x_1^k \qquad\qquad g_i^k = R_i g_1^k \quad \text{with } R_i^T R_i = I, \qquad \text{for all } i \in \mathcal{V} \text{ and all } k. \tag{43}$$

Concerning the local functions, by definition of $\mathbf{f}^s$ (32), we have, for all $i, j \in \mathcal{V}$,

$$f_i = f_j \quad \text{where } f_i \in \mathbb{R}^q \text{ is a vector containing the function values related to agent } i. \tag{44}$$

In particular, we have well that $f_i(x_i^k) = f_j(x_j^k)$, for all $i, j \in \mathcal{V}$ and all k. Let f_1 be a function interpolating the set of triplets $\{x_1^k, g_1^k, f_1^k\}_{k=0,\dots,t}$. We can choose f_i, the local function of agent i, as follows:

$$f_i(x) = f_1(R_i^T x),$$

Indeed, f_i interpolates well the set of triplets $\{x_i^k, g_i^k, f_i^k\}_{k=0,\dots,t}$ since (43) and (44) guarantee that $R_i^T x_i^k = x_1^k$, $R_i^T g_i^k = g_1^k$ and $f_i^k = f_1^k$, which are well interpolated by f_1 by definition. ◀

5 Compact PEP with multiple equivalence classes of agents

While all agents are equivalent (Assumption 19) for many common performance evaluation settings, there are advanced settings for which there are several equivalence classes of agents, see Definition 16. For example, when different groups of agents use different (uncoordinated) step-sizes, function classes, initial conditions, or even different algorithms. It can also happen that the performance measure focuses on a specific group of agents, e.g. the performance of the worst agent. The ability to efficiently and accurately evaluate the performance of distributed optimization algorithms in such advanced settings would enable a more comprehensive analysis and deeper understanding of the algorithms performance.

In this section, we exploit symmetries in different equivalence classes of agents in PEP, revealed in Proposition 17, to write agent-dependent PEPs in a compact form whose size only depends on the number of equivalence classes r, and not on the total number of agents n. By Proposition 17, we can solve the agent-dependent PEP problem, described in Section 2, restricted to symmetric agent-class solutions $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$, without impacting the worst-case value. These symmetric solutions $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$, defined in (30) and (31), depend on the partition $\mathcal{T}$ of the set of agents $\mathcal{V}$ into equivalence classes: $\mathcal{T} = \{\mathcal{V}_1, \dots, \mathcal{V}_r\}$, see Definition 16. A symmetrized solution $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ has equal blocks for equivalent agents. In this section, we order the agents by equivalence classes in the symmetrized solution $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ (30)–(31) so that we can partition it into different blocks, associated with the different agent classes. For example, if we have two equivalence classes ($r = 2$): one with two agents and one with one agent, the symmetrized solution can be written as follows:

$$\mathbf{f}_{\mathcal{T}}^s = \begin{bmatrix} f_A^1 & f_A^1 \mid f_A^2 \end{bmatrix}^T \quad \text{and} \quad G_{\mathcal{T}}^s = \left[\begin{array}{cc|c} G_A^1 & G_B^1 & G_E^{12} \\ G_B^1 & G_A^1 & G_E^{12} \\ \hline G_E^{21} & G_E^{21} & G_A^2 \end{array}\right],$$

where f_A^1, f_A^2, G_A^1, G_B^1, G_A^2, G_E^{12} and G_E^{21} are blocks defined in Definiton 18. We define new notations to aggregate the blocks of each equivalence class:

$$\mathbf{f}_{\mathcal{T}}^s = \begin{bmatrix} \mathbf{f}_{\mathcal{V}_1} & \mathbf{f}_{\mathcal{V}_2} \end{bmatrix}^T \quad \text{and} \quad G_{\mathcal{T}}^s = \begin{bmatrix} G_{\mathcal{V}_1}^s & G_{\mathcal{V}_1\mathcal{V}_2}^s \\ G_{\mathcal{V}_2\mathcal{V}_1}^s & G_{\mathcal{V}_2}^s \end{bmatrix},$$

In general, when ordering the agents by equivalence classes, the symmetrized solution $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ (30)–(31) can be written as

$$\mathbf{f}_{\mathcal{T}}^s = \begin{bmatrix} \mathbf{f}_{\mathcal{V}_1} & \dots & \mathbf{f}_{\mathcal{V}_r} \end{bmatrix}^T \qquad \text{where } \mathbf{f}_{\mathcal{V}_u} = \mathbf{1}_{n_u} \otimes f_A^u \tag{45}$$

$$G_{\mathcal{T}}^s = \begin{bmatrix} G_{\mathcal{V}_1}^s & G_{\mathcal{V}_1\mathcal{V}_2}^s & \dots & G_{\mathcal{V}_1\mathcal{V}_r}^s \\ G_{\mathcal{V}_2\mathcal{V}_1}^s & G_{\mathcal{V}_2}^s & & \vdots \\ \vdots & & \ddots & \\ G_{\mathcal{V}_r\mathcal{V}_1}^s & \dots & & G_{\mathcal{V}_r}^s \end{bmatrix} \quad \text{where } G_{\mathcal{V}_u}^s = \begin{cases} G_A^u & \text{if } n_u = 1, \\ \mathbf{1}_{n_u}\mathbf{1}_{n_u}^T \otimes G_B^u + (I_{n_u} \otimes (G_A^u - G_B^u)) & \text{if } n_u \geq 2, \end{cases} \tag{46}$$

$$\text{and } G_{\mathcal{V}_u\mathcal{V}_v}^s = \mathbf{1}_{n_u}\mathbf{1}_{n_v}^T \otimes G_E^{uv},$$

and where $f_A^u \in \mathbb{R}^q$, $G_A^u, G_B^u, G_E^{uv} \in \mathbb{R}^{p\times p}$ for $u, v = 1, \dots, r$ are the blocks repeating in $\mathbf{f}_{\mathcal{T}}^s$ and $G_{\mathcal{T}}^s$, see Definition 18.

▶ Remark (Limit cases for r).

- $r = n$: When no agents are equivalent, the partition $\mathcal{T}$ contains $r = n$ equivalence classes of size 1 and then the symmetrized agent-class solution $\mathbf{f}_{\mathcal{T}}^s$, $G_{\mathcal{T}}^s$, defined in Proposition 17 is identical to the agent-dependent PEP solution $\mathbf{f}$, G:

  $$\mathbf{f}_{\mathcal{T}}^s = F, \qquad \text{and} \qquad G_{\mathcal{T}}^s = G.$$

  Therefore, the resulting PEP formulation is equivalent to the agent-dependent PEP from Section 2 and is not compact as its size scales with $r = n$.
- $r = 1$: When all the agents are equivalent, the partition $\mathcal{T}$ contains $r = 1$ equivalence class of size n and then the symmetrized agent-class solution $\mathbf{f}_{\mathcal{T}}^s$, $G_{\mathcal{T}}^s$ defined in Proposition 17 is identical to the fully symmetrized solution $\mathbf{f}^s$, G^s defined in Corollary 20

  $$\mathbf{f}_{\mathcal{T}}^s = \mathbf{f}^s, \qquad \text{and} \qquad G_{\mathcal{T}}^s = G^s.$$

  Therefore, the resulting compact PEP formulation is identical to the one presented in Section 4.

5.1 The PEP formulation restricted to symmetric agent-class solutions

Based on the results from Proposition 17, the definitions of the symmetric PEP solutions $\mathbf{f}^s_{\mathcal{T}}$ (45) and $G^s_{\mathcal{T}}$ (46), we now show when and how an agent-dependent PEP can be expressed in a compact form, using only the blocks f^u_A, G^u_A, G^u_B, and G^{uv}_E $(u, v = 1, \dots, r)$ as variables. This will enable efficient computing of $w(\mathcal{S}_n)$ for all $n \geq 2$, even when all the agents are not equivalent. We remind that $w(\mathcal{S}_n = \{n, \mathcal{A}, t, \mathcal{P}, \mathcal{F}_u, \mathcal{I}, \mathcal{W}\})$ denotes the worst-case performance of the execution of t iterations of distributed algorithm $\mathcal{A}$ with n agents, with respect to the performance criterion $\mathcal{P}$, valid for any starting point satisfying the set of initial conditions $\mathcal{I}$, any local functions being each in a given set of function $f_i \in \mathcal{F}_{u_i}$ and any averaging matrix in the set of matrices $\mathcal{W}$.

▶ **Theorem 29** (Agent-class worst-case performance). *Let $\mathcal{V}$ be a set of $n \geq 2$ agents, $\mathcal{T}$ be a partition of $\mathcal{V}$ into $r \leq n$ equivalence classes of agents, as defined in Definition 15, and $\mathcal{W} = \mathcal{W}_{(\lambda^-, \lambda^+)}$ with $\lambda^- \leq \lambda^+ \in (-1, 1)$.*

If the algorithm $\mathcal{A} \in \mathcal{A}_D$, the performance measure $\mathcal{P}$, the initial conditions $\mathcal{I}$, and the interpolation constraints for each $\mathcal{F}_u$ $(u = 1, \dots, r)$ are linearly (or LMI) Gram representable, then the computation of $w(\mathcal{S}_n)$ can be formulated as a compact SDP PEP, with f^u_A, G^u_A, G^u_B, and G^{uv}_E $(u, v = 1 \dots, r)$ as variables and, hence, whose size only depends on the number of agent equivalence classes r but not directly on the total number of agents n.

▶ Remark. The blocks G^u_B contain scalar products between variables of different, but equivalent, agents. Therefore, G^u_B does not exist for equivalence classes with only one agent $n_u = 1$. The blocks G^{uv}_E contain scalar products between variables of different agents that are not equivalent. Therefore, these blocks do not exist when $r = 1$. The other blocks f^u_A and G^u_A, related to only one agent, always exist.

To prove this theorem, we rely on the following lemma which reformulates the SDP condition $G^s \succeq 0$ with the blocks G^u_A, G^u_B, and G^{uv}_E, and generalizes Lemma 25.

▶ **Lemma 30.** *Let $G^s_{\mathcal{T}} \in \mathbb{R}^{np \times np}$ be a symmetrized agent-class Gram matrix, as defined in* (46)*. The SDP constraint $G^s_{\mathcal{T}} \succeq 0$ can be expressed with $G^u_A \in \mathbb{R}^{p \times p}$, $G^u_B \in \mathbb{R}^{p \times p}$ and $G^{uv}_E \in \mathbb{R}^{p \times p}$, $u, v = 1 \dots, r$:*

$$G^s_{\mathcal{T}} \succeq 0 \quad \iff \quad \begin{cases} G^u_A \succeq 0 & \text{for all } u \text{ with } n_u = 1, \\ G^u_A - G^u_B \succeq 0 & \text{for all } u \text{ with } n_u \geq 2, \end{cases} \quad \text{and} \quad H_{\mathcal{T}} \succeq 0, \tag{47}$$

where $H_{\mathcal{T}} \in \mathbb{R}^{rp \times rp}$ is a symmetric matrix composed of r^2 blocks of dimension $p \times p$:

$$H^{uv}_{\mathcal{T}} = \begin{cases} n_u G^u_A + n_u(n_u - 1) G^u_B, & \text{when } u = v, \\ n_u n_v G^{uv}_E, & \text{when } u \neq v \end{cases}. \qquad u, v = 1, \dots, r. \tag{48}$$

Proof. We have that

$$G^s_{\mathcal{T}} \succeq 0 \quad \iff z^T G^s_{\mathcal{T}} z \geq 0 \quad \text{for all vector } z \in \mathbb{R}^{np}.$$

A vector $z \in \mathbb{R}^{np}$ can be divided into r sub-vectors of length $n_u p$ to match with the blocks of matrix $G^s_{\mathcal{T}}$ (see Proposition 17). We also decompose each sub-vector z_u in two terms: the average of z_u over the agents from the class $\mathcal{V}_u$, and the remaining part of $z^\perp_u$:

$$z_u = \mathbf{1}_{n_u} \otimes \bar{z}_u + z^\perp_u \quad \text{with} \quad \bar{z}_u = \frac{1}{n_u}(\mathbf{1}_{n_u} \otimes I_p)^T z_u, \qquad \bar{z}_u \in \mathbb{R}^p,\ z^\perp_u \in \mathbb{R}^{n_u p} \text{ and } z_u \in \mathbb{R}^{n_u p}.$$

By definition, we have that $\frac{1}{n_u}(\mathbf{1}_{n_u} \otimes I_p)^T z^\perp_u = 0$ for all $j = 1, \dots, r$. We now compute $z^T G^s_{\mathcal{T}} z$ (for any vector $z \in \mathbb{R}^{np}$) by summing over each block:

$$z^T G^s_{\mathcal{T}} z = \sum_{u=1}^{r} z_u^T G^s_{\mathcal{V}_u} z_u + \sum_{u=1}^{r} \sum_{v \neq u} z_u^T G^s_{\mathcal{V}_u \mathcal{V}_v} z_v, \tag{49}$$

where $G^s_{\mathcal{V}_u}$ and $G^s_{\mathcal{V}_u \mathcal{V}_v}$ are defined in (46). In the definition, we have two cases for $G^s_{\mathcal{V}_u}$, depending on the value of n_u. In the rest of the proof, we assume $n_u \geq 2$ and only work with $G^s_{\mathcal{V}_u} = \mathbf{1}_{n_u}\mathbf{1}^T_{n_u} \otimes G^u_B + (I_{n_u} \otimes (G^u_A - G^u_B))$. The special case for $n_u = 1$, $G^s_{\mathcal{V}_u} = G^u_A$, can be adapted to this case $n_u \geq 2$, by defining $G^u_B = 0$ when $n_u = 1$.

Here is the development for each term of (49):

$$\begin{aligned} z_u^T G^s_{\mathcal{V}_u} z_u &= \left(\mathbf{1}_{n_u} \otimes \bar{z}_u + z_u^\perp\right)^T \left(\mathbf{1}_{n_u}\mathbf{1}_{n_u}^T \otimes G_B^u + (I_{n_u} \otimes (G_A^u - G_B^u))\right)\left(\mathbf{1}_{n_u} \otimes \bar{z}_u + z_u^\perp\right) \\ &= \bar{z}_u^T n_u^2 G_B^u \bar{z}_u + \bar{z}_u^T n_u (G_A^u - G_B^u)\bar{z}_u + (z_u^\perp)^T (I_{n_u} \otimes (G_A^u - G_B^u)) z_u^\perp, \\ &= \bar{z}_u^T (n_u G_A^u + n_u(n_u-1) G_B^u)\bar{z}_u + (z_u^\perp)^T (I_{n_u} \otimes (G_A^u - G_B^u)) z_u^\perp, \end{aligned}$$

$$\begin{aligned} z_u^T G^s_{\mathcal{V}_u\mathcal{V}_v} z_v &= \left(\mathbf{1}_{n_u} \otimes \bar{z}_u + z_u^\perp\right)^T \left(\mathbf{1}_{n_u}\mathbf{1}_{n_v}^T \otimes G_E^{uv}\right)\left(\mathbf{1}_{n_v} \otimes \bar{z}_v + z_v^\perp\right) \\ &= \bar{z}_u n_u n_v G_E^{uv} \bar{z}_v. \end{aligned}$$

Using these expressions, we can write (49) as a sum of quadratic forms

$$z^T G^s_{\mathcal{T}} z = \bar{z}^T H_{\mathcal{T}} \bar{z} + \sum_{u=1}^{r} (z_u^\perp)^T (I_{n_u} \otimes (G_A^u - G_B^u)) z_u^\perp,$$

where $\bar{z} \in \mathbb{R}^{rp}$ is a vector stacking the average vectors $\bar{z}_u$ of each equivalence class ($u = 1, \ldots, r$), and $H_{\mathcal{T}} \in \mathbb{R}^{rp \times rp}$ is the matrix defined in (48). Therefore, the constraint $G^s_{\mathcal{T}} \succeq 0$ can be expressed as

$$\bar{z}^T H_{\mathcal{T}} \bar{z} + \sum_{u=1}^{r} (z_u^\perp)^T (I_{n_u} \otimes (G_A^u - G_B^u)) z_u^\perp \geq 0, \quad \text{for all } \bar{z} \in \mathbb{R}^{rp} \text{ and all } z_u^\perp \in \mathbb{R}^{n_u p} \text{ s.t. } (\mathbf{1}_{n_u} \otimes I_p)^T z_u^\perp = 0 \tag{50}$$

Conditions $H_{\mathcal{T}} \succeq 0$ and $G_A^u - G_B^u \succeq 0$ (for all $u = 1, \ldots, r$) are sufficient to ensure that (50) is satisfied and hence that $G^s \succeq 0$. To show the necessity of these conditions, let us consider different cases for $\bar{z}$ and $z_u^\perp$:

- When $z_u^\perp = 0$ for all u, then (50) simply imposes that $\bar{z}^T H_{\mathcal{T}} \bar{z} \geq 0$ for all $\bar{z} \in \mathbb{R}^{rp}$, i.e. $H_{\mathcal{T}} \succeq 0$.
- When $\bar{z} = 0$, $z_u^\perp = 0$, for all u, except for an arbitrary $u = v$, then (50) imposes

  $$\sum_{i=1}^{n_v} (z_v^\perp(i))^T (G_A^v - G_B^v) z_v^\perp(i) \geq 0, \quad \text{for all } z_v^\perp \in \mathbb{R}^p \text{ such that } (\mathbf{1}_{n_v} \otimes I_p)^T z_v^\perp = 0,$$

  where $z_v^\perp(i) \in \mathbb{R}^p$, for $i = 1, \ldots, n_v$, denotes the i^{th} part of vector $z_v^\perp \in \mathbb{R}^{n_v p}$. Since $n_v \geq 2$, we can choose $z_v^\perp(1) = -z_v^\perp(2) = s$, for an arbitrary $s \in \mathbb{R}^p$, and $z_v^\perp(j) = 0$ for $j \geq 3$, so that the above equation becomes $s^T(G_A^v - G_B^v)s \geq 0$ for all $s \in \mathbb{R}^p$, which is equivalent to $G_A^v - G_B^v \succeq 0$. This is valid for an arbitrary $v = 1, \ldots, r$.

Therefore, by setting $G_B^u = 0$ when $n_u = 1$, we have well proved the equivalence (47) for the SDP condition $G^s_{\mathcal{T}} \succeq 0$. ◀

Proof of Theorem 29. As detailed in Section 2.2, the set of averaging matrices $\mathcal{W}_{(\lambda^-,\lambda^+)}$ has Gram-representable necessary interpolation constraints, see Corollary 13. Since $\mathcal{A}, \mathcal{P}, \mathcal{I}$ and interpolation constraints for each $\mathcal{F}_u$ are also Gram-representable, Proposition 5 guarantees that the computation of $w(n, \mathcal{A}, t, \mathcal{P}, \mathcal{I}, \mathcal{F}_u, \mathcal{W}_{(\lambda^-,\lambda^+)})$ can be formulated as an agent-dependent SDP PEP problem (8). Moreover, by Proposition 17, we can restrict the PEP to symmetric agent-class solutions of the form of $\mathbf{f}^s_{\mathcal{T}}$ (30) and $G^s_{\mathcal{T}}$ (31). The symmetric solutions $\mathbf{f}^s_{\mathcal{T}}$ and $G^s_{\mathcal{T}}$ are composed of blocks f_A^u, G_A^u, G_B^u, and G_E^{uv} $(u, v = 1 \ldots, r)$. Therefore, all the PEP elements can be expressed in terms of these smaller blocks. Lemma 30 shows how the SDP constraint $G^s_{\mathcal{T}} \succeq 0$, can be expressed in terms of G_A^u, G_B^u, and G_E^{uv}. The other elements of the PEP can directly be expressed in terms of f_A^u, G_A^u, G_B^u, and G_E^{uv} $(u, v = 1 \ldots, r)$, using the definition of $\mathbf{f}^s_{\mathcal{T}}$ (30) and $G^s_{\mathcal{T}}$ (33), as detailed in Appendix A. Since all the PEP elements can be expressed in terms of blocks $f_A^u \in \mathbb{R}^q$, $G_A^u \in \mathbb{R}^{p \times p}$, $G_B^u \in \mathbb{R}^{p \times p}$ (when $n_u \geq 2$) and $G_E^{uv} \in \mathbb{R}^{p \times p}$ $(u, v = 1 \ldots, r)$, the problem can be made smaller by considering them as the variables of the SDP PEP, instead of the full matrix $G^s_{\mathcal{T}} \in \mathbb{R}^{np \times np}$ and vector $\mathbf{f}^s_{\mathcal{T}} \in \mathbb{R}^{nq}$. The size of the problem depends on the number r of equivalence classes of agents, and not directly on the total number of agents. ◀

Theorem 29 shows that the worst-case performance of a distributed optimization method can be computed with an SDP PEP whose size only depends on the number r of equivalence classes of agents, but not directly on the total number of agents n. Therefore, if the number of equivalence classes r is independent of n, so is the size of the problem. However, the problem can depend on n and n_u $(u = 1, \ldots, r)$ as parameters. Appendix A provides details to explicitly construct this compact PEP formulation.

Infinitely many agents

If the number r of equivalence classes of agents is independent of n, so is the size of the compact SDP PEP, and we can solve it efficiently for any value of n. In particular, we can take the limit when n tend to ∞ in the problem, to compute the worst-case performance of an algorithm on an infinitely large network of agents

$$\lim_{n\to\infty} w\big(n, \mathcal{A}, t, \mathcal{P}, \mathcal{I}, \mathcal{F}_u, \mathcal{W}_{(\lambda^-,\lambda^+)}\big),$$

and to determine if this performance is bounded or not. In all the cases we treated (see Section 6), we observed that if the worst-case value for $n \to \infty$ is bounded, then it depends on the size proportions of each equivalence class $\rho_1, \dots, \rho_r$:

$$\rho_u = \lim_{n\to\infty} \frac{n_u}{n}.$$

6 Case study: the EXTRA algorithm

In this section, we leverage our new PEP formulation, based on equivalence classes of agents, to evaluate the performance of the EXTRA algorithm [25] in advanced settings, and the performance evolution with system size n. This provides totally new insights into the worst-case performance of the algorithm. The EXTRA method, described in Algorithm 1, is a well-known distributed optimization method that was introduced in [25] and further developed in [26, 11, 15].

Algorithm 1 EXTRA

Choose step-size $\alpha > 0$, matrices W, $\widetilde{W} \in \mathbb{R}^{n\times n}$ and pick any $x_i^0 \in \mathbb{R}^d$ $(\forall\, i)$;
$x_i^1 = \sum_j w_{ij} x_j^0 - \alpha \nabla f_i(x_i^0)$, $\forall\, i$;
for $k = 0, 1, \dots$ **do**
  $x_i^{k+2} = x_i^{k+1} + \sum_j \left(w_{ij} x_j^{k+1} - \widetilde{w}_{ij} x_j^k\right) - \alpha\big(\nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\big)$, $\forall\, i$;
end for

Authors in [25] recommend to choose $\widetilde{W} = \frac{W+I}{2}$. This allows computing only one new consensus step at each iteration. In that case, the sharpest convergence results for EXTRA are given in [15], and summarized below for the strongly-convex case.

▶ **Proposition 31** (Theoretical performance guarantee for EXTRA [15]). *We consider the decentralized optimization problem* (1) *with optimal solution* $\mathbf{x}^* \in \mathbb{R}^d$. *Under the following assumptions,*

1. *All local functions* f_i $(i \in \mathcal{V})$ *are* L*-smooth and* μ*-strongly convex, with* $0 \le \mu \le L$;
2. *The averaging matrix* $W \in \mathcal{W}_{(-\lambda,\lambda)}$, *with* $\lambda \in [0, 1)$, *and* $\widetilde{W} = \frac{W+I}{2}$;
3. *The starting points satisfy*

$$\|x_i^0 - x^*\|^2 \le R_1 \qquad \text{and} \qquad \|\nabla f_i(x^*)\|^2 \le R_2 \qquad \text{for all } i = 1, \dots, n;$$

the EXTRA algorithm with $\alpha = \frac{1}{4L}$ *guarantees that*

$$E_f(t) \coloneqq f(\bar{x}^t) - f(x^*) \le (1-\tau)^t \frac{LR_1 + R_2/L}{1-\lambda}, \qquad \textit{(functions error)} \tag{51}$$

$$E_x(t) \coloneqq \frac{1}{n}\sum_{i=1}^n \|x_i^t - x^*\|^2 \le (1-\tau)^t \frac{R_1 + R_2/L^2}{1-\lambda}, \qquad \textit{(iterates error)} \tag{52}$$

where $\bar{x}^t = \frac{1}{n}\sum_{i=1}^n x_i^t$ *and* $\tau = \frac{1}{39\left(\frac{L}{\mu} + \frac{1}{1-\lambda}\right)}$.

Proof. Bounds (51) and (52) directly follows from [15, Corollary 2]. ◀

For comparison purposes, our PEP-based analysis of EXTRA will use the same assumptions as in Proposition 31, with $L = 1$, $\mu = 0.1$, $\lambda = 0.5$, $R_1 = 1$ and $R_2 = 1$, unless specified otherwise. Our PEP framework with equivalence classes of agents also enables the analysis of a variety of settings other than those of Proposition 31, many of which have not yet been studied in the literature. In this section, we explore, in particular, the performance of the worst agent, the k-th percentile performance, and the performance under agent heterogeneity in the classes of local functions. Other possible settings include agent heterogeneity in the initial conditions, the network topology, the algorithm parameters or algorithm execution. For example, it could be used to analyze the performance of an algorithm when there is a subset of malicious agents.

6.1 Performance of the worst agent

Classical analysis of distributed optimization algorithms often focuses on performance metrics averaged over all the agents, namely

$$E_f(t) = f(\bar{x}^t) - f(x^*) \qquad \text{or} \qquad E_x(t) = \frac{1}{n}\sum_{i=1}^{n} \|x_i^t - x^*\|^2, \tag{53}$$

where $\bar{x}^t = \frac{1}{n}\sum_{i=1}^n x_i^t$. Theoretical guarantees on these criteria for EXTRA are given in (51) and (52) from Proposition 31. As we have seen in Section 4, these error criteria lead to performance bounds that are independent of the number of agents n and are therefore easier to treat analytically. However, there exist other relevant performance measures that are more difficult to analyze and for which there are few or no results. This is the case of the performance of the worst agent in the network,

$$E_{f,\text{worst}}(t) = \max_{i\in\mathcal{V}} f(x_i^t) - f(x^*) \qquad \text{or} \qquad E_{x,\text{worst}}(t) = \max_{i\in\mathcal{V}} \|x_i^t - x^*\|^2, \tag{54}$$

where $\mathcal{V} = \{1, \dots, n\}$ is the set of agents in the network. To analyze these performance measures using PEP, we need to fix (arbitrarily) the agent that will be the worst one and use it in the PEP objective:

$$f(x_1^t) - f(x^*) \qquad \text{or} \qquad \|x_1^t - x^*\|^2,$$

The maximization of such objectives ensures that agent 1 will be the one with the largest error (on f or x). Indeed, if another agent j has a larger error, then the permutation of variables of agents 1 and j in the PEP solution would lead to another feasible PEP solution with a larger objective. Agent 1 will thus receive a specific role in the PEP solution and is not equivalent to the $n-1$ others. We can formulate a compact PEP using two equivalence classes of agents

$$\mathcal{V}_1 = \{1\} \qquad \text{and} \qquad \mathcal{V}_2 = \{2, \dots, n\},$$

and applying the theory developed in Section 5.

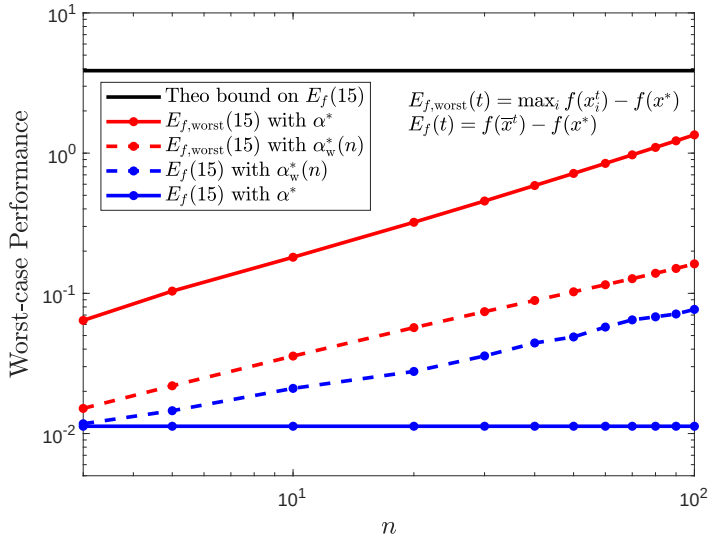

Figure 1 Comparison of the average functional error E_f (53) and the worst functional error $E_{f,\text{worst}}$ (54) for $t = 15$ iterations of the EXTRA algorithm and their evolution with the number of agents n in the system. The plot shows (i) the constant theoretical bound on E_f (51), in black, (ii) the PEP bounds on the function error of the worst agent $E_{f,\text{worst}}$, scaling sublinearly with n, in red and (iii) the PEP bounds for the function error of the average iterate E_f, in blue. The constant step-size α^* has been optimized for this performance criterion and leads to a bound constant with n (plain blue line). Step-sizes can also be optimized with respect to $E_{f,\text{worst}}(15)$, leading to smaller step-sizes $\alpha^*_w(n)$ decreasing as $\frac{1}{\sqrt{n}}$. They improve the guarantee for $E_{f,\text{worst}}$ and its scaling with n but they deteriorate the corresponding guarantee for E_f (see dashed lines). In this plot, the range of eigenvalues for the averaging matrix is $[-0.5, 0.5]$ and the local functions are 1-smooth and 0.1-strongly convex.

The worst agent functions error

Figure 1 shows the evolution of E_f and $E_{f,\text{worst}}$ with the size of the network, for 15 iterations of the EXTRA algorithm. These results have been validated with the agent-dependent PEP formulation (from Section 2) for small values of n, for verification purposes. However, as n becomes larger, this agent-dependent formulation becomes computationally intractable, reaching an SDP size of 325×325 for $n = 10$ (and $t = 15$). This limitation precisely motivated the development of the compact symmetrized PEP formulations from Sections 4 and 5.

Firstly, Figure 1 shows that the PEP bound for the average performance E_f is well constant over n, as predicted by Theorem 23, and outperforms the theoretical bound from [15, Corollary 2], which requires 2750 iterations to guarantee the same error as the PEP bounds after 15 iterations. The theoretical bound is valid for a step-size $\alpha = \frac{1}{4L}$, while the PEP bound is shown for an optimized step-size $\alpha^* = \frac{0.78}{L}$, which minimizes the bound. Since the bound is constant over n, the value of the optimized step-size can be used for any system size, however, it may depend on other settings, such as the range of eigenvalues for W: $[\lambda^-, \lambda^+]$.

There is currently no theoretical guarantee for the worst agent functions error $E_{f,\text{worst}}$ of EXTRA in the literature, but we can analyze it with our compact PEP formulation. Figure 1 shows that the PEP bound for the worst agent $E_{f,\text{worst}}$ (in red) is scaling sublinearly with the number of agents n. Using the constant step-size $\alpha^* = \frac{0.78}{L}$ (plain red line), we obtain a logarithmic slope of 0.92. The step-size can also be tuned with respect to $E_{f,\text{worst}}$, and would then have a different value for each value of n, denoted $\alpha^*_{\text{w}}(n)$. Using $\alpha^*_{\text{w}}(n)$ (dashed red line), we obtain a logarithmic slope of 0.67, which means the scaling with n is less important when using suitable step-sizes. In our case, the step-sizes $\alpha^*_{\text{w}}(n)$ are diminishing in $\frac{1}{\sqrt{n}}$. However, these smaller step-sizes deteriorate the average performance E_f (dashed blue line).

The worst agent iterates error

There is currently no theoretical guarantee for the worst agent iterates error $E_{x,\text{worst}}$ of EXTRA in the literature, but one could derive[2] a conservative bound on $E_{x,\text{worst}}$ from (52), scaling linearly with the number of agents n. Using our compact PEP formulation, we can analyze such performance metric accurately and show that the performance of the worst agent $E_{x,\text{worst}}$ for EXTRA actually scales sublinearly with n, as shown in Figure 2. Figure 2 shows, in particular, PEP-based bounds for E_x and $E_{x,\text{worst}}$. The observations are similar to those of Figure 1. The PEP bound for the average performance E_x is constant over n, as predicted by Theorem 23 and the corresponding optimized step-size is $\alpha^* = \frac{0.78}{L}$. Moreover, the PEP bound for the worst agent $E_{x,\text{worst}}$ is scaling sublinearly with the number of agents n. Using the constant step-size $\alpha^* = \frac{0.78}{L}$ (plain red line), we obtain a logarithmic slope of 0.82. As previously, the step-size can also be tuned with respect to $E_{x,\text{worst}}$, and would then have a different value for each value of n, denoted $\alpha^*_{\text{w}}(n)$. Using the optimal diminishing step-sizes $\alpha^*_{\text{w}}(n)$ (dashed red line), we obtain a logarithmic slope of 0.60, which means the scaling with n is less important when using suitable step-sizes. These smaller step-sizes only slightly deteriorate the average performance E_x (dashed blue line).

6.2 The 80-th percentile of the agent performance

In some cases, the performance of the worst agent may not be the most appropriate performance measure. When the network is very large, the performance of the worst agent can be terribly bad, while the average performance is good. Using our PEP approach, we can consider other performance measures that were so far completely out of reach of theoretical analysis. Inspired by statistical approaches, we propose here to analyze the k-th percentile in the distribution of the individual performance of each agent. In this experiment, we have chosen the 80-th percentile, that is, the performance of the worst agent, after excluding the 20% worst agents. In other words, it measures the error at or below which 80% of the agents fall in the worst-case scenario. The performance of the worst agent, analyzed in the previous subsection, can be seen as the 100-th percentile of the agent performance and can be analyzed via PEP using two equivalence classes of agents. When considering smaller percentiles, such as the 80-th, we can formulate a compact PEP using three equivalence classes of agents:

$$\mathcal{V}_1 = \{1, \dots, 0.2n\}, \qquad \mathcal{V}_2 = \{0.2n + 1\}, \qquad \mathcal{V}_3 = \{0.2n + 2, \dots, n\},$$

where we assume n can be divided by 5. The class $\mathcal{V}_1$ contains the 20% worst agents to exclude, the class $\mathcal{V}_2$ contains the agent for which the individual performance will provide the 80-th percentile, and $\mathcal{V}_3$ contains all

[2] using the fact that $\max_{i\in\mathcal{V}} \|x_i^t - x^*\|^2 \leq \sum_{i=1}^n \|x_i^t - x^*\|^2$

the other agents. The PEP should therefore maximize the performance of agent $i \in \mathcal{V}_2$ and impose that the performance of each agent $j \in \mathcal{V}_j$ is larger:

$$\begin{aligned} E_{x,80}(t) = \max_{\{f_A^u, G_A^u, G_B^u G_E^{uv}\}_{u,v=1,2,3}} & \|x_i^t - x^*\|^2 \quad \text{with } i \in \mathcal{V}_2 \\ \text{s.t.} \quad & \|x_j^t - x^*\|^2 \geq \|x_i^t - x^*\|^2 \quad \text{for all } j \in \mathcal{V}_1, \\ & \text{All the other PEP constraints.} \end{aligned} \tag{55}$$

In problem (55), the squared norms $\|x_i^t - x^*\|^2$ can be written as $(e_{x^t} - e_{x^*})^T G_A^u (e_{x^t} - e_{x^*})$ with u chosen according to the class $\mathcal{V}_u$ containing agent i. Details about the explicit construction of compact PEP formulations are given in Appendix A.

Figure 2 shows the evolution of $E_{x,80}$ with the size of the network, for 15 iterations of the EXTRA algorithm (green line). We observe that the scaling of $E_{x,80}$ with the number of agents n is very limited and quickly reaches a plateau, making this performance almost as good as the average performance E_x. Moreover, the optimal step-size for this bound, denoted α_{80}^*, is independent of n and, in this case, it stays close to α^*, optimized for E_x. The value of the plateau for $E_{x,80}$ has been confirmed by solving the compact PEP with n tending to ∞, as described at the end of Section 5.

Figure 3 shows the evolution with k of the k-th percentile of the agent performance for EXTRA, when $n \to \infty$. We see that the limit of the k-th percentile performance is finite for all values of k $\in (0, 100)$ and has a vertical asymptote in k = 100. The 100-th percentile computes the performance of the worst agent and grows well to infinity with n (see Figure 2). According to Figure 3, the value of the k-th percentile performance first increases slowly with k, and then it starts to blow up after some threshold on k, depending on t, to match the vertical asymptote in k = 100. This means that it seems useful to exclude a small part of the worst agents in the performance metric, e.g. 10–20% in the settings we are considering, but not more. Moreover, the k-th percentile of EXTRA for $n \to \infty$ seems to converge geometrically. Indeed, in Figure 3, when considering 5 more iterations in total, the k-th percentile improves by a factor between 2.2 and 3.2, depending on k.

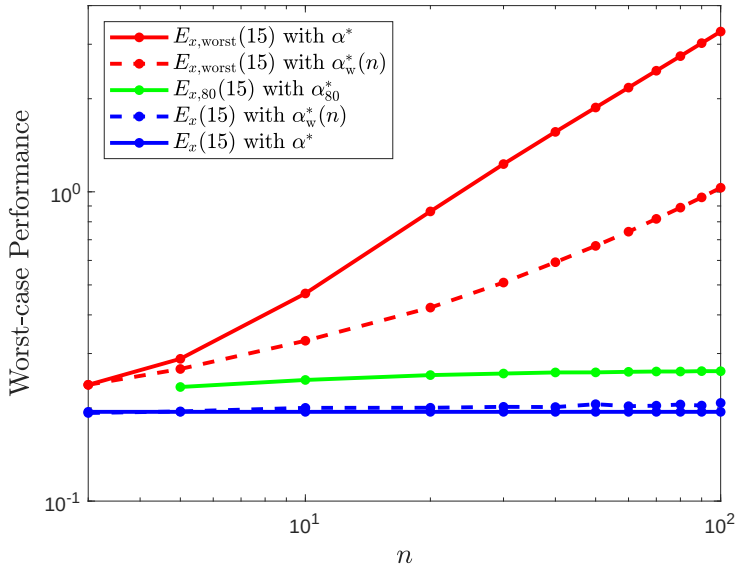

Figure 2 Comparison of the average iterates error E_x (53), the worst iterates error $E_{x,\text{worst}}$ (54) and the 80-th percentile iterates error $E_{x,80}$ (55) for $t = 15$ iterations of the EXTRA algorithm and their evolution with the number of agents n in the system. The plot shows (i) the PEP bounds on the iterates error of the worst agent $E_{x,\text{worst}}$, scaling sublinearly with n, in red, (ii) the PEP bound on the 80-th percentile iterates error $E_{x,80}$, in green, and (iii) the PEP bounds for the average iterates error E_x, in blue. The constant step-size α^* has been hand-tuned for this performance criterion and leads to a bound constant with n (plain blue line). Step-sizes can also be hand-tuned with respect to $E_{f\text{worst}}(15)$, leading to smaller step-sizes $\alpha_w^*(n)$. They improve the guarantee for $E_{f,\text{worst}}$ and its scaling with n without deteriorating too much the corresponding guarantee for E_f (see dashed lines). In this plot, the range of eigenvalues for the averaging matrix is $[-0.5, 0.5]$ and the local functions are 1-smooth and 0.1-strongly convex.

▶ Remark. Figures 2 and 3 show the performance of EXTRA for rather well-connected averaging matrices ($\lambda = 0.5$). Similar results and observations can be obtained for larger ranges of eigenvalues, i.e. $\lambda \in (0.5, 1)$, but this would require more iterations to get small errors.

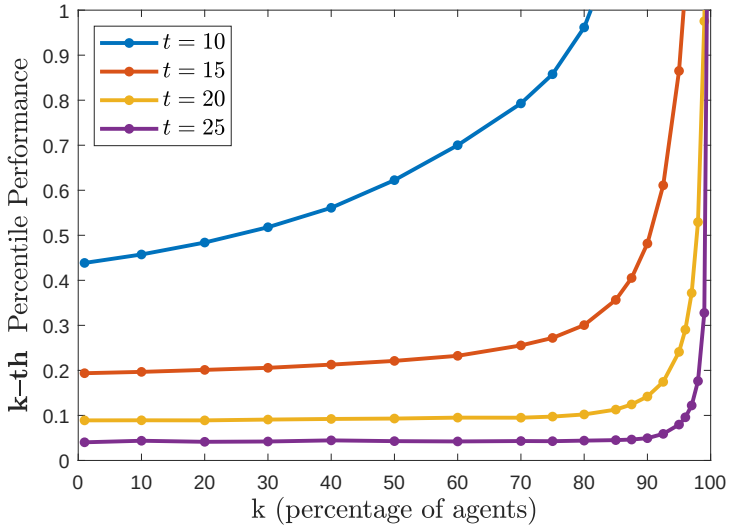

Figure 3 Evolution with k of the k-th percentile of the agent performance for t iterations of EXTRA, when the number of agents tends to infinity ($n \to \infty$). Different total numbers of iterations t are compared. In this plot, the step-size is $\alpha = 0.78$, the range of eigenvalues for the averaging matrix is $[-0.5, 0.5]$ and the local functions are 1-smooth and 0.1-strongly convex.

6.3 Performance under local functions heterogeneity

In many applications, the agents hold local functions sharing general properties, e.g. convexity or smoothness, but it is unlikely that all the local functions have exactly the same characterization for these properties. For example, the local functions can be μ_i-strongly convex and L_i-smooth, but with different parameters μ_i and L_i. In general, the performance is then computed by considering that all the local functions have the worst parameters:

$$\mu = \min_i \mu_i \qquad \text{and} \qquad L = \max_i L_i.$$

In this experiment, we analyze the gain in the performance guarantee when, instead of considering $f_i \in \mathcal{F}_{\mu,L}$ for all i, we consider two equivalence classes of agents $\mathcal{V}_1$ and $\mathcal{V}_2$ such that

$$f_i \in \mathcal{F}_{\mu_1,L} \quad \text{for all } i \in \mathcal{V}_1 \quad \text{and} \quad f_i \in \mathcal{F}_{\mu_2,L} \quad \text{for all } i \in \mathcal{V}_2.$$

These two classes of agents manipulate functions with two different condition numbers $\kappa_1 = \frac{L}{\mu_1}$ and $\kappa_2 = \frac{L}{\mu_2}$. Figure 4 shows the situation where $L = 1$, $\mu_1 = 0.01$ and $\mu_2 = 0.1$, for different sizes of classes $\mathcal{V}_1$ and $\mathcal{V}_2$. We observe that the guarantees for EXTRA only depend on the relative composition of each class $\mathcal{V}_1$ and $\mathcal{V}_2$ but not on the total number of agents n. This observation has been confirmed by solving the compact PEPs for n tending to ∞, as described at the end of Section 5. In Figure 4, when 20% of the agents pass from the class $\mathcal{V}_1$ to the class $\mathcal{V}_2$, the worst-case guarantee is improved by a constant factor 0.9. This improving factor can be linked to the worst-case guarantees obtained with a uniform value for κ in the local functions ($\kappa = 100$ or $\kappa = 10$ for all the agents) and motivates the following conjecture:

▶ **Conjecture 32.** *Let $E_x^{\kappa_1,\kappa_2}(t,\theta)$ be the performance guarantee after t iterations of EXTRA with θn agents with local functions in $\mathcal{F}_{\mu_1,L}$ and $(1-\theta)n$ agents with local functions in $\mathcal{F}_{\mu_2,L}$, for $\theta \in [0,1]$. Let $E_x^{\kappa_1}(t)$ and $E_x^{\kappa_2}(t)$ be the worst-case errors of t iterations of the EXTRA algorithm with uniform conditioning for all the local functions, respectively with $\kappa_1 = \frac{L}{\mu_1}$ and $\kappa_2 = \frac{L}{\mu_2}$. By definition, we have*

$$E_x^{\kappa_1}(t) = E_x^{\kappa_1,\kappa_2}(t,1) \quad \text{and } E_x^{\kappa_2}(t) = E_x^{\kappa_1,\kappa_2}(t,0)$$

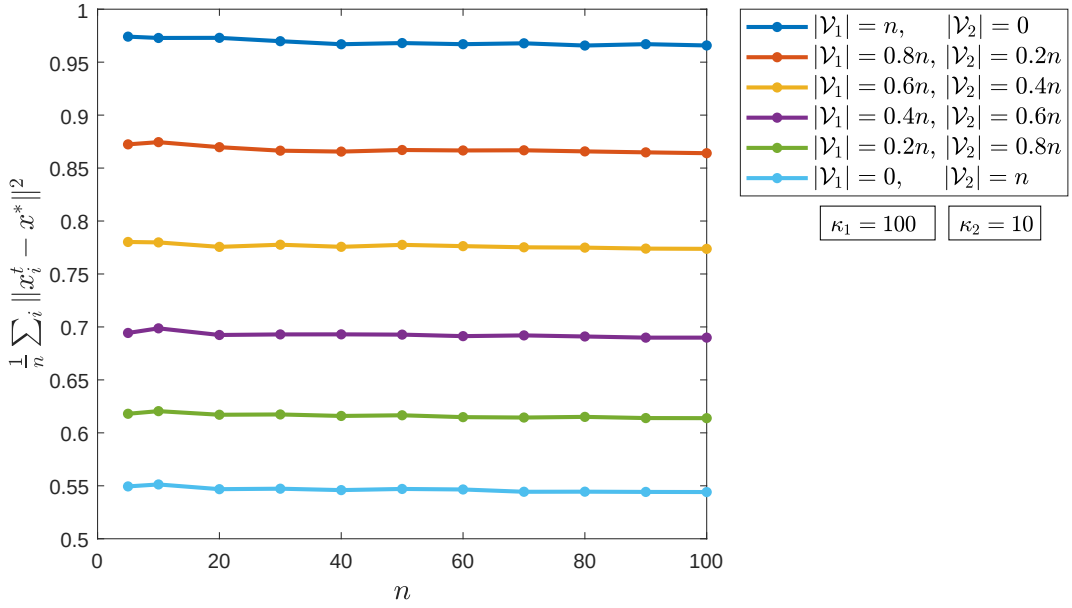

Figure 4 Evolution with the number of agents n of the worst-case average distance to optimum E_x for 15 iterations of EXTRA with two equivalence classes of agents $\mathcal{V}_1$ and $\mathcal{V}_2$ using each a different condition number for their function class. The agents in $\mathcal{V}_1$ hold 1-smooth and 0.01-strongly convex local functions ($\kappa_1 = 100$) and the agents in $\mathcal{V}_2$ hold 1-smooth and 0.1-strongly convex local functions ($\kappa_2 = 10$). The guarantees are independent of the total number of agents n and only depend on the proportion of agents in each class. In this experiment, the range of eigenvalues for the averaging matrix is $[-0.5, 0.5]$.

We conjecture that the performance $E_x^{\kappa_1,\kappa_2}(t,\theta)$ *can be expressed in function of* $E_x^{\kappa_1}(t)$ *and* $E_x^{\kappa_2}(t)$ *as*

$$E_x^{\kappa_1,\kappa_2}(t,\theta) = \left(E_x^{\kappa_1}(t)\right)^{\theta}\left(E_x^{\kappa_2}(t)\right)^{1-\theta} \qquad \text{for all } t > 0,\ \kappa_1, \kappa_2 > 0 \text{ and } \theta \in [0,1]\ .$$

In addition to the values of κ_1, κ_2 and θ shown in Figure 4, this conjecture has been verified for different other values of κ_1, κ_2 and θ. It can also be extended to any number of agent equivalence classes, associated with different values of κ. For example, we have verified it with 3 classes of agents. This results in much tighter bounds in cases where only a few agents in the system hold ill-conditioned local functions. As future direction, this also allows exploring situations where the agents hold totally different types of local functions, e.g. strongly-convex ($\mu > 0$) and weakly-convex ($\mu < 0$) functions, or smooth and non-smooth functions. This conjecture would have been difficult to guess without the PEP tool and the compact formulation developed in this paper, which shows its usefulness.

6.4 On the numerical resolution of the compact SDP PEP formulation

In this case study, we used Matlab to model and solve the SDP PEP problems. In particular, we used the YALMIP toolbox for modeling and the Mosek solver for solving. Our code is available on GitHub (`https://github.com/sebcolla/Performance-Estimation-Problems-for-distributed-optimization`). The Mosek solver, like most of the other existing ones, relies on the interior-point method and the existence of a Slater point in the feasible set of the problem. Some equality constraints in our problems may prevent the existence of a Slater point, hence leading the solver to find solutions of low numerical quality. Indeed, the compact PEP formulation, described in Sections 4 and 5, and detailed in Appendix A, contains equality constraints that impose rank deflection of the matrix variables of the resulting SDP, see for example, the optimality constraints, or the average preserving condition for consensus steps (Propositions 37 and 36 in Appendix A). This may be problematic for the numerical conditioning of the problem, and thus the numerical quality of its solution. We have verified the accuracy of the worst-case value given by the compact PEP formulation for $n \leq 5$, and always observed a relative error of less than 1%. However, the worst-case solutions (i.e. functions and iterates) can be very high-dimensional.

In the agent-dependent PEP formulations, all the problematic equality constraints can easily be removed, by exploiting them to reduce the size of the SDP. This allows the problem and its solution to be well-conditioned. However, such dimensionality reductions are more involved in our new compact PEP SDP formulations. We hope that existing toolboxes and solvers will be extended to automatically reduce the SDP size by exploiting equality constraints. Therefore, for now, if one wishes to evaluate the performance of a distributed optimization algorithm in settings for which we know the result will be independent of n by Theorem 23, we recommend solving the agent-dependent formulation with $n = 2$ because the current solvers will perform better on this version of the problem. Moreover, formulating such an agent-dependent PEP is more intuitive and is made user-friendly by the toolboxes helping in building and solving PEP, i.e. PESTO in Matlab [31] and PEPit in Python [9]. By Corollary 24, the obtained results for $n = 2$ agents will be valid for any $n \geq 2$, including for $n \to \infty$.

7 Conclusion

In this paper, we have harnessed agent symmetries to develop compact SDP PEP formulations for computing the worst-case performance of decentralized optimization algorithms. When all the agents are equivalent, we have characterized the situations where the performance is totally independent of the number of agents, which allows analyzing and computing the performance in the fundamental case with only two agents. Such situations include many common settings for the performance evaluation of distributed optimization algorithms. We have also shown that in the worst-case scenario, when all agents are equivalent, their local sequences of iterates are rotations of each other and their local functions are identical up to a change of variables. Moreover, this new compact PEP formulation also allows analyzing advanced performance settings with several equivalence classes of agents, enabling tighter analysis and deeper understanding of the algorithm performance, as we demonstrated with EXTRA. We believe that the different contributions of this paper can significantly help the research in the field of distributed optimization by simplifying the analysis and understanding of the performance of distributed algorithms.

A Appendix: Explicit expressions of the compact PEP components

A.1 When all agents are equivalent

Theorem 23 characterizes the settings in which we can formulate the performance computation of a distributed optimization algorithm as an SDP PEP which is totally independent of the number of agents n. In particular, we need all the agents to be equivalent (Assumption 19), so we can work with a fully symmetric solution $(\mathbf{f}^s, G^s)$ (Corollary 20), composed of small repeating blocks f_A, G_A, and G_B. The resulting agent-dependent PEP, expressed with blocks f_A, G_A, and G_B has a size independent of the number of agents n, but the value of n can still appear in the PEP and impact its solution, as any other parameter of the problem. Using appropriate changes of variables, we can obtain a PEP formulation that is fully independent of n, under the condition stated in part 2 of Theorem 23. In this appendix, we show how we can explicitly build this agent-independent PEP formulation for many usual settings. For that purpose, we pass through the main components of a PEP, that we have introduced in Section 2, and detail how they can be written with the blocks f_A, G_A, and G_B and which changes of variable get rid of n.

Firstly, the SDP condition $G^s \succeq 0$ can be expressed with G_A, and G_B using Lemma 25. Then, Lemma 26 makes it independent of n, by introducing a new block $G_C = \frac{1}{n}(G_A + (n-1)G_B)$. Similarly to this result, the agent-independent formulation of other PEP components relies on the blocks G_A (33) and G_C (36) of the Gram matrices. We link these two matrices by defining *the difference block* G_D:

$$G_D = G_A - G_C. \tag{56}$$

We summarize below how these three blocks, related to the symmetrized Gram G^s by definition, can be expressed based on the blocks of any Gram matrix solution G (29):

$$G_A = \frac{1}{n}\sum_{i=1}^{n} G_{ii}, \qquad G_C = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} G_{ij} \quad \text{and} \quad G_D = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(G_{ii} - G_{ij}). \tag{57}$$

Using these three blocks, we can easily write many expressions in PEP, independently of n. However, only two of them should be variables of the resulting compact SDP PEP, as the third one could always be obtained using (56). In this appendix, we call agent-independently Gram-representable the expressions that can be written f_A, G_A, G_C, and G_D, without any dependence on the number of agents n.

▶ **Definition 33** (Agent-independently Gram-representable)**.** *Let f_A be the block of the symmetrized function values vector, as defined in* (32)*, G_A, G_C, and G_D the symmetric blocks defined in* (57)*. We say that an expression, such as a constraint or an objective, is agent-independently Gram-representable if it can be expressed using a finite set of linear or LMI constraints or expressions involving only f_A, G_A, G_C, and G_D, without any dependence on the number of agents n.*

The formulation of single-agent constraints (Definition 22) into agent-independently Gram-representable constraints will rely on relations (41) and (42), involving f_A and G_A. The formulation of scale-invariant expressions (Definition 21) into agent-independently Gram-representable expressions will rely on relations (38), (39), (40), involving respectively f_A, G_A, and G_C. We also define a useful relation involving G_D, which encapsulates a combination of relations (39) and (40). This new relation allows expressing the average over the n agents of the scalar products of two centered variables related to the same agents:

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^T(y_i - \bar{y}) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i^T y_i - x_i^T y_j) = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} e_x^T(G_{ii} - G_{ij})e_y = e_x^T(G_D)e_y, \tag{58}$$

where x_i and y_j are any variables of agent i and j and $\bar{x} = \frac{1}{n}\sum_{j=1}^{n} x_i$ and $\bar{y} = \frac{1}{n}\sum_{j=1}^{n} y_i$ are their agent averages. The last equality in (58) follows from the definition of G_D (57). Moreover, $e_x, e_y \in \mathbb{R}^p$ denote coefficient vectors for variables x and y. As a reminder, a coefficient vector e_x contains linear coefficients selecting the correct combination of columns in $P_i \in \mathbb{R}^{d\times p}$ (28) to obtain vector $x_i \in \mathbb{R}^d$, i.e. $P_i e_x = x_i$, for any $i = 1, \dots, n$. This notation allows, for example, to write $x_i^T x_i$ as $x_i^T x_i = e_x^T P_i^T P_i e_x = e_x^T G_{ii} e_x$.

In what follows, we will exploit all these relations, to write all the components of a PEP, independently of n.

The following proposition treats the interpolation constraints of a PEP.

▶ **Proposition 34** (Function interpolation constraints)**.** *Let $\mathcal{F}$ be a set of functions for which there exist linearly Gram-representable interpolation constraints. In a PEP for a distributed optimization method, with fully symmetrized solutions $(\mathbf{f}^s, G^s)$ (32)–(33), the constraints $f_i \in \mathcal{F}$, for all $i \in \mathcal{V}$ can be expressed using a set of interpolation constraints that are agent-independently Gram-representable and that only involves f_A and G_A.*

Proof. Each constraint $f_i \in \mathcal{F}$ can be written with a set of Gram-representable interpolation constraints involving function values of agent i and scalar products between quantities (i.e. points and gradients) related to agent i. These are thus single-agent constraints that are all identical when we restrict to agent-symmetric PEP solution $(\mathbf{f}^s, G^s)$ (32)–(33). Therefore, they can all be expressed with f_A and G_A using (41) and (42). ◀

▶ Remark. For example, when $\mathcal{F}$ is the set of convex functions, we have,

$$f_i \in \mathcal{F} \quad \iff \quad f_i^k \geq f_i^l + (g_i^l)^T(x_i^k - x_i^l) \quad \forall\, k,l \quad \iff \quad e_{f^k}^T f_A \geq e_{f^l}^T f_A + e_{g^l}^T (G_A)(e_{x^k} - e_{x^l}),$$

where $f_i^k = f_i(x_i^k)$, $g_i^l = \nabla f_i(x_i^l)$ and e_{f^k}, e_{f^l}, e_{g^l}, e_{x^k} and e_{x^l} are appropriate vectors of coefficients. This is obtained by applying relation (41) for $f_i(x_i^k)$ and $f_i(x_i^l)$ and relation (42) for the scalar product between g_i^l and $(x_i^k - x_i^l)$.

Algorithm description

The class of methods $\mathcal{A}_D$ that we consider is defined in Definition 2 and may combine gradient evaluation, consensus steps, and linear combinations of variables. In the latter, each agent defines the same linear combination of its local variables. It should be the same combination for all the agents, otherwise, they are not equivalent, and Assumption 19 does not hold. All the variables in the combinations are related to the same agent and can therefore be squared and formulated in PEP using G_A, as in (39). Alternatively, we can reduce the number of variables in the blocks of the Gram matrix by using the linear combination to define a vector of coefficients in terms of the others, as shown in Proposition 35:

▶ **Proposition 35** (Linear combinations)**.** *Let $P_i = [y_i^1, \dots, y_i^p]$ be the p variables of agent i. We consider any linear combination of the variables, coordinated over all the agents $i \in \mathcal{V}$*

$$z_i = \sum_{k=1}^{p} \beta_k y_i^k \qquad \text{for all } i, \tag{59}$$

where β_k are given coefficients and $y_i^k \in \mathbb{R}^d$ are any local variables from agent i. Such a combination step can be formulated in the PEP by defining the vector of coefficient e_z based on the vectors $e_{y^1}, \dots, e_{y^p}$.

$$e_z = \sum_{k=1}^{p} \beta_k e_{y^k}$$

This avoids adding z_i as a new column of P_i.

Proof. By definition, we have $z_i = P_i e_z$ and $y_i^k = P_i e_{y^k}$. Therefore, the linear equality (59) can be written as

$$P_i \left(e_z - \sum_{k=1}^{p} \beta_k e_{y^k} \right) = 0,$$

which allows defining e_z based on the other vectors of coefficients $e_z = \sum_{k=1}^{p} \beta_k e_{y^k}$, without adding a new column in P_i. ◀

Concerning the consensus steps, Theorem 23 applies to the set of averaging matrices $\mathcal{W}_{(\lambda^-, \lambda^+)}$, defined in Section 2. Necessary interpolation constraints for this set of matrices are given in (23), (24) and (25), from Corollary 13. Proposition 36 below shows that these interpolation constraints are agent-independently Gram-representable, by expressing them in terms of G_C and G_D. In principle, the compact PEP formulation could apply to a general set of averaging matrices $\mathcal{M}$ for which there exist Gram-representable interpolation constraints. Future work may include the description of other classes of averaging matrices, in particular, the extension to non-symmetric stochastic matrices with a bound on their singular value.

▶ **Proposition 36** (Averaging matrix interpolation constraints)**.** *The averaging matrix interpolation constraints* (23)*,* (24) *and* (25)*, from Corollary 13, are agent-independently Gram-representable and can be expressed using* G_C *and* $G_D \in \mathbb{R}^{p\times p}$ *as:*

$$(e_X - e_Y)^T G_C (e_X - e_Y) = 0, \tag{60}$$

$$(e_Y - \lambda^- e_X)^T G_D (e_Y - \lambda^+ e_X) \preceq 0, \tag{61}$$

$$e_Y^T (G_D) e_X - e_X^T (G_D) e_Y = 0, \tag{62}$$

where $e_X, e_Y \in \mathbb{R}^p$ *are matrices of coefficients such that* $P_i e_X = X_i$*,* $P_i e_Y = Y_i$ *with* $X_i, Y_i \in \mathbb{R}^{d\times t}$ *the matrices with iterates* $x_i^k, y_i^k \in \mathbb{R}^d$ *as columns.*

Proof. Firstly, we express (23) as (60). Equation (23) $\overline{X} = \overline{Y}$ means that the agent average is preserved and can be written with scalar products as:

$$\left(\frac{1}{n}\sum_{i=1}^n (X_i - Y_i)\right)^T \left(\frac{1}{n}\sum_{j=1}^n (X_j - Y_j)\right) = 0,$$

where matrices $X_i, Y_i \in \mathbb{R}^{d\times t}$ contain the different consensus iterates $x_i, y_i \in \mathbb{R}^d$ of agent i as columns. Using relation (40), applied to columns of $X_i - Y_i$ and $X_j - Y_j$, this equation can be written as

$$\frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (X_i - Y_i)^T (X_j - Y_j) = (e_X - e_Y)^T G_C (e_X - e_Y) = 0.$$

Secondly, we express (16) as (61). Equation (16) involve centered matrices $X_\perp, Y_\perp \in \mathbb{R}^{nd\times t}$,

$$(Y_\perp - \lambda^- X_\perp)^T (Y_\perp - \lambda^+ X_\perp) \preceq 0, \tag{63}$$

We can write this product as a sum over the agents of products of centered variables, which then allows using relation (58) to express it with G_D. To this end, let us define $U_i = Y_i - \lambda^- X_i$, $V_i = Y_i - \lambda^+ X_i$, and $\overline{U}, \overline{V}$ their corresponding agent average: $\overline{U} = \frac{1}{n}\sum_{i=1}^n U_i$, $\overline{V} = \frac{1}{n}\sum_{i=1}^n V_i$. Equation (63) can be written as

$$\sum_{i=1}^n (U_i - \overline{U})^T (V_i - \overline{V}) \preceq 0.$$

This can be reformulated using relation (58) applied to columns of U_i and V_i and dividing the equation by n:

$$\frac{1}{n}\sum_{i=1}^n (U_i - \overline{U})^T (V_i - \overline{V}) = e_r^T G_D e_V = (e_Y - \lambda^- e_X)^T G_D (e_Y - \lambda^+ e_X) \preceq 0$$

Finally, the symmetry condition (17) can be expressed as (62) using a similar technique

$$X_\perp^T Y_\perp - Y_\perp^T X_\perp = \sum_{i=1}^n (X_i - \overline{X})^T (Y_i - \overline{Y}) - \sum_{i=1}^n (Y_i - \overline{Y})^T (X_i - \overline{X}) = 0.$$

This can be reformulated using relation (58) applied to columns of X_i and Y_i and dividing the equation by n:

$$e_Y^T (G_D) e_X - e_X^T (G_D) e_Y = 0$$ ◀

Points common to all agents (x^*, $\overline{x}$, etc)

As explained in Section 3, to build our compact PEP formulation for distributed optimization, we divide the agent-dependent PEP solutions into blocks of variables related to each agent, see (27) and (28), recalled below

$$\mathbf{f} = \begin{bmatrix} f_1^T & \dots & f_n^T \end{bmatrix}, \qquad \text{where } f_i = \left[f_i^k\right]_{k\in I} \in \mathbb{R}^q \text{ is a vector with the } q \text{ function values of agent } i,$$

$$P = \begin{bmatrix} P_1 & \dots & P_n \end{bmatrix}, \qquad \text{where } P_i \in \mathbb{R}^{d\times p} \text{ contains the } p \text{ vector variables related to agent } i,$$

e.g. $P_i = \begin{bmatrix} y_i^k & x_i^k & g_i^k \end{bmatrix}_{k\in I}$, and the Gram matrix $G \in \mathbb{R}^{np\times np}$ is thus defined as

$$G = P^T P = \begin{bmatrix} P_1^T P_1 & P_1^T P_2 & \dots \\ P_2^T P_1 & \ddots & \\ \vdots & & P_n^T P_n \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & \dots \\ G_{21} & \ddots & \\ \vdots & & G_{nn} \end{bmatrix}. \tag{64}$$

The variables common to all agents, such as x^* or $\bar{x}^t$, are copied into each agent block P_i and their definitions are added as constraints of the problem, e.g.

$$\bar{x}_i = \bar{x} = \frac{1}{n}\sum_{j=1}^n x_j \quad \text{for all } i \in \mathcal{V}, \qquad \text{or} \qquad \frac{1}{n}\sum_{i=1}^n \nabla f_i(x_i^*) = 0, \quad \text{with } x_i^* = x_j^* = x^* \text{ for all } i,j \in \mathcal{V}, \tag{65}$$

Such definition constraints are agent-independently Gram-representable, as shown in Propositions 37 and 38 below. If used in the problem, each agent i also holds the variables for its local function and gradient values associated with this common point. For example, for x^*, we consider a new triplet (x_i^*, g_i^*, f_i^*) for each agent i, which is taken into account in the interpolation constraints of its local function. These interpolation constraints ensure that g_i^* and f_i^* correspond to a gradient vector and function value at x^* that are consistent with the given class of functions. Vectors x_i^* and g_i^* are columns of P_i and value f_i^* is an element of vector f_i.

To summarize, for any point x_c common to all the agents in the problem, we need two elements in the PEP:

i. the definition constraints for x_c, e.g. (65),

ii. the interpolation constraints for the new triplet (x_c, g_c, f_c), if g_c or f_c used in the PEP.

The addition of a new triplet of points to consider in the interpolation constraints does not alter the result of Proposition 40 and they can still be written as agent-independently Gram-representable constraints with f_A and G_A. The definition constraints can be treated as any other constraint of the problem, e.g. the initial constraints, and these constraints can thus be formulated independently of n if they can be written with scale-invariant expressions (see Definition 21 and Theorem 23). Proposition 37 gives an agent-independent formulation for the definition constraint of the optimal solution x^* of the decentralized optimization problem (1).

▶ **Proposition 37** (Optimality constraint)**.** *Let x^* be an optimal point for the decentralized optimization problem* (1)*. The definition constraints for the common variable x^* in a PEP restricted to fully symmetric solutions* $(\mathbf{f}^s, G^s)$ (32)–(33)*, given by the system-level stationarity constraint*

$$\frac{1}{n}\sum_{i=1}^n \nabla f_i(x_i^*) = 0, \quad \textit{with } x_i^* = x_j^* = x^* \textit{ for all } i,j \in \mathcal{V}, \tag{66}$$

is agent-independently Gram-representable and can be expressed as

$$e_{g^*}^T(G_C)e_{g^*} = 0 \quad \textit{and} \quad e_{x^*}^T(G_D)e_{x^*} = 0,$$

where $e_{g^*}, e_{x^*} \in \mathbb{R}^p$ are vectors of coefficients such that $P_i e_{g^*} = g_i^* = \nabla f_i(x^*)$, and $P_i e_{x^*} = x_i^*$ for all $i = 1,\dots,n$.

Proof. The first constraint in (66) guarantees that the optimal point x_i^* held by each agent satisfies the system-level stationarity constraint $\frac{1}{n}\sum_{i=1}^n \nabla f_i(x_i^*) = 0$. Then, the second set of constraints imposes that the local optimal point x_i^* is the same for any agent i. Let g_i^* denote the gradient of function f_i at the optimal point x_i^*. We can formulate the stationarity constraint using only scalar products as

$$\left(\frac{1}{n}\sum_{i=1}^n g_i^*\right)^T\left(\frac{1}{n}\sum_{j=1}^n g_j^*\right) = \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (g_i^*)^T g_j^* = 0,$$

Using relation (40), applied to vectors g_i^*, g_j^*, we recover condition $e_{g^*}^T(G_C)e_{g^*} = 0$. For the second set of constraints, we can impose them all with one constraint involving only scalar products:

$$\frac{1}{2n}\sum_{i=1}^n\sum_{j=1}^n (x_i^* - x_j^*)^T(x_i^* - x_j^*) = \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (x_i^*)^T x_i^* - (x_j^*)^T x_j^* = 0$$

which can be written, using relation (58), as

$$\frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (x_i^*)^T x_i^* - (x_i^*)^T x_j^* = \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (e_{x^*})^T(G_{ii} - G_{ij})(e_{x^*}) = (e_{x^*})^T(G_D)(e_{x^*}).$$ ◀

▶ Remark. Without loss of generality, the optimal solution x^* of problem (1) can be set to $x^* = 0$. In that case, the constraint $x_i^* = x_j^*$, written as $e_{x^*}^T(G_D)e_{x^*} = 0$, is not needed to define x^* properly. Indeed, it is sufficient to say that the coefficient vector is zero: $e_{x^*} = 0$. This improves the numerical conditioning of the resulting SDP PEP.

Proposition 38 gives an agent-independent formulation for the definition constraint of the agent average iterate $\bar{x}^k$. To simplify notation, we omit the k index for iterations.

▶ **Proposition 38** (Agent average definition). *We consider a PEP for distributed optimization, restricted to fully symmetric solutions* $(\mathbf{f}^s, G^s)$ (32)–(33)*. The definition constraints for* $\bar{x}$

$$\bar{x}_i = \bar{x} = \frac{1}{n}\sum_{j=1}^{n} x_j \quad \textit{for all } i \in \mathcal{V}, \tag{67}$$

are agent-independently Gram-representable, and can be written as

$$e_{\bar{x}}^T(G_A)e_{\bar{x}} + e_x^T G_C(e_x - 2e_{\bar{x}}) = 0,$$

where $e_{\bar{x}}, e_x \in \mathbb{R}^p$ *are vectors of coefficients such that* $P_i e_{\bar{x}} = \bar{x}_i$*, and* $P_i e_x = x_i$ *for all* $i = 1, \ldots, n$*.*

Proof. We can formulate the set of constraints (67) with one constraint involving only scalar products of variables:

$$\frac{1}{n}\sum_{i=1}^{n}\left(\bar{x}_i^t - \frac{1}{n}\sum_{j=1}^{n} x_j^t\right)^T\left(\bar{x}_i^t - \frac{1}{n}\sum_{l=1}^{n} x_l^t\right) = 0$$

This expression can be expanded to be written with G_A and G_C:

$$\frac{1}{n}\sum_{i=1}^{n}(\bar{x}_i)^T(\bar{x}_i) + \frac{1}{n^2}\sum_{j=1}^{n}\sum_{l=1}^{n} x_j^T x_l - \frac{2}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} x_j^T\bar{x}_i = e_{\bar{x}}^T(G_A)e_{\bar{x}} + e_x^T G_C(e_x - 2e_{\bar{x}}) = 0,$$

where we used relation (39) for the first term and (40) for the other two. ◀

Initial conditions and performance measures

By combining terms of the form of (38), (39), (40), (41), (42), and (58), possibly involving common points defined above, it is possible to express many initial conditions and performance measures in terms of f_A, G_A, G_C, and G_D, without dependence on n. Examples of usual initial conditions and their reformulation are given in Proposition 39 below. The proposition focuses on the initial conditions, but all the expressions involved in these constraints can also be adapted as a performance measure, using the last point x^t instead of the initial point x^0.

▶ **Proposition 39** (Initial conditions). *Let* $x^* \in \mathbb{R}^d$ *denote the optimal solution of the distributed optimization problem* (1)*,* $x_i^0 \in \mathbb{R}^d$ *the initial iterate of agent* i*,* $\bar{x}^0$ *the average initial iterate and* $R \in \mathbb{R}$ *a constant. Here is a list of initial conditions that are agent-independently Gram-representable when restricted to agent-symmetric solutions, together with their reformulations in terms of* f_A*,* G_A*,* G_C*, and* G_D*:*

Initial condition	*Reformulation*
$\frac{1}{n}\sum_{i=1}^{n}\|x_i^0 - x^*\|^2 \le R^2$,	$(e_{x^0} - e_{x^*})^T(G_A)(e_{x^0} - e_{x^*}) \le R^2$,
or $\|x_i^0 - x^*\|^2 \le R^2$ *for all* $i = 1, \ldots, n$	
$\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(x_i^0)\|^2 \le R^2$,	$(e_{g^0})^T(G_A)(e_{g^0}) \le R^2$,
or $\|\nabla f_i(x_i^0)\|^2 \le R^2$ *for all* $i = 1, \ldots, n$	
$\frac{1}{n}\sum_{i=1}^{n}\|x_i^0 - \bar{x}^0\|^2 \le R^2$,	$(e_{x^0})^T G_D(e_{x^0}) \le R^2$,
$x_i^0 - x_j^0 = 0$ *for all* $i, j = 1, \ldots, n$	$(e_{x^0})^T G_D(e_{x^0}) = 0$,
$\frac{1}{n}\sum_{i=1}^{n}\big(f_i(\bar{x}^0) - f_i(x^*)\big) \le R$	$\big(e_{f(\bar{x}^0)} - e_{f(x^*)}\big)^T f_A\big(e_{f(\bar{x}^0)} - e_{f(x^*)}\big) \le R$

where $e_{x^0}, e_{x^*}, e_{g^0} \in \mathbb{R}^p$ *and* $e_{f(\bar{x}^0)}, e_{f(x^*)} \in \mathbb{R}^q$ *are vectors of coefficients such that* $P_i e_{x^0} = x_i^0$, $P_i e_{x^*} = x^*$, $P_i e_{g^0} = \nabla f_i(x_i^0)$, $e_{f(\bar{x}^0)}^T f_i = f_i(\bar{x}^0)$ *and* $e_{f(x^*)}^T f_i = f_i(x^*)$, *for all* $i = 1, \dots, n$.

Proof. To obtain the first two reformulations, we can use relation (39) applied respectively to vectors $x_i^0 - x^*$ and $\nabla f_i(x_i^0)$

$$\frac{1}{n}\sum_{i=1}^n \|x_i^0 - x^*\|^2 = \frac{1}{n}\sum_{i=1}^n (x_i^0 - x^*)^T(x_i^0 - x^*) = (e_{x^0} - e_{x^*})^T(G_A)(e_{x^0} - e_{x^*}) \leq R^2.$$
$$\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(x_i^0)\|^2 = \frac{1}{n}\sum_{i=1}^n (\nabla f_i(x_i^0))^T(\nabla f_i(x_i^0)) = (e_{g^0})^T(G_A)(e_{g^0}) \leq R^2.$$

When considering the alternative uniform conditions for all agents, we obtain the same reformulations using relation (42), applied respectively to vectors $x_i^0 - x^*$ and $\nabla f_i(x_i^0)$. For the third condition, we can directly use relation (58) to obtain it. Then, the fourth condition is equivalent to

$$\frac{1}{2n^2}\sum_{i=1}^n\sum_{j=1}^n (x_i^0 - x_j^0)^T(x_i^0 - x_j^0) = \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (x_i^0)^T x_i^0 - (x_i^0)^T x_j^0 = 0,$$

which can be written, using relation (58), as

$$\frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (x_i^0)^T x_i^0 - (x_i^0)^T x_j^0 = \frac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n (e_{x^0})^T(G_{ii} - G_{ij})(e_{x^0}) = (e_{x^0})^T(G_D)(e_{x^0}).$$

Finally, the fifth and last condition can be reformulated with f_A using relation (38) applied to $\frac{1}{n}\sum_{i=1}^n f_i(\bar{x}^0)$ and $\frac{1}{n}\sum_{i=1}^n f_i(x^*)$. ◀

A.2 With multiple equivalence classes of agents

Let $\mathcal{T}$ be a partition of the agent set $\mathcal{V}$ into r equivalence classes of agents (see Definition 16). Considering the PEP restricted to symmetrized agent-class solutions $G_{\mathcal{T}}^s$ and $\mathbf{f}_{\mathcal{T}}^s$ (45)–(46), we now detail how we can express all the PEP elements in terms of f_A^u, G_A^u, G_B^u, and G_E^{uv} $(u, v = 1 \dots, r)$, to be able to use this blocks as variable of the compact SDP PEP formulation. For the SDP condition $G^s \succeq 0$, we can be express it with G_A^u, G_B^u, and G_E^{uv} using Lemma 30. For the other PEP elements, we need to adapt the reformulation techniques (38), (39), (40), (41), (42) and (58) to the situation where the agent set $\mathcal{V}$ is partitioned into r subsets of equivalent agents $\mathcal{T} = \{\mathcal{V}_1, \dots, \mathcal{V}_r\}$. The terms that we can be expressed in terms of f_A^u, G_A^u, G_B^u, and G_E^{uv} are the following (for any variables x_i and y_j related to agents i and j, including the case $x_i = y_j$):

- The function value of a given agent $j \in \mathcal{V}_u$

$$f_j(x_j) = \frac{1}{n_u}\sum_{i \in \mathcal{V}_u} f_i(x_i) = e_{f(x)}^T f_A^u, \qquad \text{for any } j = 1, \dots, n, \tag{68}$$

  where the first equality holds because $f_i(x_i) = f_j(x_j)$ for any $i, j \in \mathcal{V}_u$, by definition of the symmetric agent-class solution $\mathbf{f}_{\mathcal{T}}^s$ (45). This implies that

$$\frac{1}{n}\sum_{i=1}^n f_i(x_i) = \frac{1}{n}\sum_{u=1}^r \sum_{j \in V_u}^n f_j(x_j) = \frac{1}{n}\sum_{u=1}^r n_u e_{f(x)}^T f_A^u = e_{f(x)}^T f_{A,\mathcal{T}}, \tag{69}$$

  where vector $f_{A,\mathcal{T}} \in \mathbb{R}^q$ is given by

$$f_{A,\mathcal{T}} = \frac{1}{n}\sum_{u=1}^r n_u f_A^u. \tag{70}$$

- The scalar product between two variables related to the same agent $i \in \mathcal{V}_u$:

$$x_i^T y_i = \frac{1}{n_u}\sum_{j \in V_u}^n x_j^T y_j = e_x^T G_A^u e_y \qquad \text{for any } i = 1, \dots, n, \tag{71}$$

where the first equality holds because $x_i^T y_i = x_j^T y_j$ for any $i, j \in \mathcal{V}_u$, by definition of the symmetric agent-class solution $G_{\mathcal{T}}^s$ (46). This implies that

$$\frac{1}{n}\sum_{i=1}^{n} x_i^T y_i = \frac{1}{n}\sum_{u=1}^{r}\sum_{j\in V_u}^{n} x_j^T y_j = \frac{1}{n}\sum_{u=1}^{r} n_u e_x^T G_A^u e_y = e_x^T (G_{A,\mathcal{T}}) e_y, \tag{72}$$

where matrix $G_{A,\mathcal{T}} \in \mathbb{R}^{p\times p}$ is given by

$$G_{A,\mathcal{T}} = \frac{1}{n}\sum_{u=1}^{r} n_u G_A^u. \tag{73}$$

- The average over the n^2 pairs of agents of the scalar products between two variables, each related to any agent:

$$\begin{aligned}
&\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^T y_j = \frac{1}{n^2}\sum_{u=1}^{r}\sum_{v=1}^{r}\sum_{i\in\mathcal{V}_u}\sum_{j\in\mathcal{V}_v} x_i^T y_j = \frac{1}{n^2}\sum_{u=1}^{r}\sum_{i\in\mathcal{V}_u}\sum_{j\in\mathcal{V}_u} x_i^T y_j + \frac{1}{n^2}\sum_{u=1}^{r}\sum_{v\neq u}\sum_{i\in\mathcal{V}_u}\sum_{j\in\mathcal{V}_v} x_i^T y_j,\\
&\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^T y_j = e_x^T\left(\frac{1}{n^2}\sum_{u=1}^{r}\left(n_u G_A^u + n_u(n_u-1)G_B^u\right) + \frac{1}{n^2}\sum_{u=1}^{r}\sum_{v\neq u} n_u n_v\, G_E^{uv}\right) e_y = e_x^T(G_{C,\mathcal{T}})e_y,
\end{aligned} \tag{74}$$

where matrix $G_{C,\mathcal{T}} \in \mathbb{R}^{p\times p}$ is the sum of all the $p\times p$ blocks of $H_{\mathcal{T}}$ divided by n^2:

$$G_{C,\mathcal{T}} = \frac{1}{n^2}\left(\sum_{u=1}^{r} n_u(G_A^u + (n_u-1)G_B^u) + \sum_{u=1}^{r}\sum_{v\neq u} n_u n_v\, G_E^{uv}\right). \tag{75}$$

- The average over the n agents of the scalar products of two centered variables related to the same agents:

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^T(y_i-\bar{y}) &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i^T y_i - x_i^T y_j) = \frac{1}{n^2}\sum_{u=1}^{r}\sum_{v=1}^{r}\sum_{i\in\mathcal{V}_u}\sum_{j\in\mathcal{V}_v}(x_i^T y_i - x_i^T y_j),\\
&= e_x^T\left(\frac{1}{n^2}\sum_{u=1}^{r} n_u(n_u-1)(G_A^u - G_B^u) + \frac{1}{n^2}\sum_{u=1}^{r}\sum_{v\neq u} n_u n_v (G_A^u - G_E^{uv})\right) e_y,
\end{aligned} \tag{76}$$

$$\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^T(y_i-\bar{y}) = e_x^T(G_{D,\mathcal{T}})e_y$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$, and matrix $G_{D,\mathcal{T}} \in \mathbb{R}^{p\times p}$ is given by

$$G_{D,\mathcal{T}} = \frac{1}{n^2}\left(\sum_{u=1}^{r} n_u(n_u-1)(G_A^u - G_B^u) + \sum_{u=1}^{r}\sum_{v\neq u} n_u n_v (G_A^u - G_E^{uv})\right) \tag{77}$$

By definitions (73), (75) and (77), we have $G_{A,\mathcal{T}} = G_{C,\mathcal{T}} + G_{D,\mathcal{T}}$. Using relations (68), (69), (71), (72), (74) and (76), we can easily adapt Propositions 34–39 to this general case with multiple equivalence classes of agents to build all the PEP elements for distributed optimization only based on blocks f_A^u, G_A^u, G_B^u, and G_E^{uv}.

▶ **Proposition 40** (Function interpolation constraints)**.** *Let $\mathcal{F}_u$ be a set of functions for which there exist linearly Gram-representable interpolation constraints. When considering symmetric agent-class PEP solutions $(\mathbf{f}_{\mathcal{T}}^s - G_{\mathcal{T}}^s)$ (45)–(46), the set of interpolation constraints for $\mathcal{F}_u$, applied independently to each agent of a given equivalence class $\mathcal{V}_u$, can be expressed with f_A^u and G_A^u.*

Proof. The constraint $f_i \in \mathcal{F}_u$ for each $i \in \mathcal{V}_u$ can be written with a set of Gram-representable interpolation constraints involving function values of agent i and scalar products between quantities (i.e. points and gradients) related to agent i. These are thus single-agent constraints, that are identical for all $i \in \mathcal{V}_u$ when we restrict to a symmetrized agent-class solution $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ (45)–(46). Therefore, they can all be expressed with f_A^u and G_A^u using (68) and (71). ◀

Algorithm description

The class of methods $\mathcal{A}_D$ that we consider is defined in Definition 2 and may combine gradient evaluation, consensus steps, and linear combinations of variables. In the latter, the agents from the same equivalence class should define the same linear combinations of their local variables, otherwise they are not equivalent. Agents from different equivalence classes may apply different combinations. This allows considering heterogeneity in the algorithm parameters, for example, to analyze the effect of uncoordinated step-sizes. In any case, all the variables in the combinations are related to the same agent and can therefore be squared and formulated in PEP using G_A^u, as in (71).

Concerning the consensus steps, Theorem 29 applies to the set of averaging matrices $\mathcal{W}_{(\lambda^-,\lambda^+)}$, defined in Section 2. Necessary interpolation constraints for this set of matrices are given in (23), (24) and (25), from Corollary 13. Proposition 36 below shows that these interpolation constraints can be expressed in terms of $G_{C,\mathcal{T}}$ (75) and $G_{D,\mathcal{T}}$ (77), and so in terms of G_A^u, G_B^u, and G_E^{uv} $(u, v = 1, \dots, r)$.

▶ **Proposition 41** (Averaging matrix interpolation constraints)**.** *The averaging matrix interpolation constraints* (23), (24) *and* (25)*, from Corollary 13 can be expressed with G_A^u, G_B^u, and G_E^{uv} $(u, v = 1, \dots, r)$, as:*

$$\begin{aligned}(e_X - e_Y)^T G_{C,\mathcal{T}}(e_X - e_Y) &= 0,\\ (e_Y - \lambda^- e_X)^T G_{D,\mathcal{T}}(e_Y - \lambda^+ e_X) &\preceq 0,\\ e_Y^T(G_{D,\mathcal{T}})e_X - e_X^T(G_{D,\mathcal{T}})e_Y &= 0,\end{aligned}$$

where $G_{C,\mathcal{T}}$ and $G_{D,\mathcal{T}}$ are defined in (75) *and* (77) *and $e_X, e_Y \in \mathbb{R}^p$ are matrices of coefficients such that $P_i e_X = X_i$, $P_i e_Y = Y_i$ with $X_i, Y_i \in \mathbb{R}^{d\times t}$ the matrices with iterates $x_i^k, y_i^k \in \mathbb{R}^d$ as columns.*

Proof. The proof is similar to the one of Proposition 36 but using relation (74) instead of (40) and (76) instead of (58) ◀

Points common to all agents

As detailed in Appendix A.1 in the case where all agents are equivalent, for any point x_c common to all the agents in the problem, we need two elements in the PEP:

i. the definition constraints for x_c, e.g. (65),

ii. the interpolation constraints for the new triplet (x_c, g_c, f_c), if g_c or f_c used in the PEP.

The addition of a new triplet of points to consider in the interpolation constraints does not alter the result of Proposition 34 and they can still be written with f_A^u and G_A^u. The definition constraints can be treated as any other constraint of the problem. When there are several equivalence classes of agents in PEP, this allows more types of constraints to be expressed in the compact PEP formulation, and thus enlarges the type of common points that can be defined in the problem. Optimal point x^* and agent-average $\bar{x}$ can still be expressed in the compact problem, as shown in Propositions 42 and 43.

▶ **Proposition 42** (Optimality constraint)**.** *Let x^* be an optimal point for the decentralized optimization problem* (1)*. The definition constraints for the common variable x^* in a PEP restricted to symmetrized agent-class solutions* $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ (45)–(46)*, given by the system-level stationarity constraint*

$$\frac{1}{n}\sum_{i=1}^{n} \nabla f_i(x_i^*) = 0, \quad \text{with } x_i^* = x_j^* = x^* \text{ for all } i, j \in \mathcal{V},$$

can be expressed with G_A^u, G_B^u, and G_E^{uv} $(u, v = 1, \dots, r)$ as

$$e_{g^*}^T(G_{C,\mathcal{T}})e_{g^*} = 0 \quad \text{and} \quad e_{x^*}^T(G_{D,\mathcal{T}})e_{x^*} = 0,$$

where $G_{C,\mathcal{T}}$ and $G_{D,\mathcal{T}}$ are defined in (75) *and* (77) *and $e_{g^*}, e_{x^*} \in \mathbb{R}^p$ are vectors of coefficients such that $P_i e_{g^*} = g_i^* = \nabla f_i(x^*)$, and $P_i e_{x^*} = x_i^*$ for all $i = 1, \dots, n$.*

Proof. The proof is similar to the one of Proposition 37 but using relation (74) instead of (40) and (76) instead of (58). ◀

▶ Remark. Without loss of generality, the optimal solution x^* of problem (1) can be set to $x^* = 0$. In that case, the constraint $x_i^* = x_j^*$, written as $e_{x^*}^T(G_{D,\mathcal{T}})e_{x^*} = 0$, is not needed to define x^* properly. Indeed, it is sufficient to say that the coefficient vector is zero: $e_{x^*} = 0$. This improves the numerical conditioning of the resulting SDP PEP.

▶ **Proposition 43** (Agent average definition)**.** *We consider a PEP for distributed optimization, restricted to symmetrized agent-class solutions* $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ (45)–(46)*. The definition constraints for* $\bar{x}$

$$\bar{x}_i = \bar{x} = \frac{1}{n}\sum_{j=1}^{n} x_j \quad \text{for all } i \in \mathcal{V},$$

can be written with G_A^u*,* G_B^u*, and* G_E^{uv} $(u, v = 1, \ldots, r)$ *as*

$$e_{\bar{x}}^T(G_{A,\mathcal{T}})e_{\bar{x}} + e_x^T G_{C,\mathcal{T}}(e_x - 2e_{\bar{x}}) = 0,$$

where $G_{A,\mathcal{T}}$ *and* $G_{C,\mathcal{T}}$ *are defined in* (73) *and* (75)*, and* $e_{\bar{x}}, e_x \in \mathbb{R}^p$ *are vectors of coefficients such that* $P_i e_{\bar{x}} = \bar{x}_i$*, and* $P_i e_x = x_i$ *for all* $i = 1, \ldots, n$*.*

Proof. The proof is similar to the one of Proposition 38 but using relation (72) instead of (39) and (74) instead of (40). ◀

A point common to all agents, that can only be defined using different equivalence classes of agents, is the iterate of a given agent x_1^t, at which all the local functions are evaluated $f_1(x_1^t), \ldots, f_n(x_1^t)$. This can be used to evaluate the performance of the worst agent, see Section 6.1. The definition of such a point is treated in the following proposition.

▶ **Proposition 44** (Specific agent definition)**.** *We consider a PEP for distributed optimization, restricted to symmetrized agent-class solutions* $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ (45)–(46)*. Let* $\mathcal{V}_1 = \{1\}$ *be one of the* r *equivalence classes of the PEP. If* x_1 *is a common variable used by all the agents in the PEP, its definition constraints are given by*

$$x_{c,i} = x_1 \quad \text{for all } i \in \mathcal{V} \tag{78}$$

and can be written with G_A^u *and* G_E^{uv} $(u, v = 1, \ldots, r)$ *as*

$$e_{x_c}^T(G_A^u)e_{x_c} + e_x^T(G_A^1)e_x - 2e_{x_c}^T(G_E^{u1})e_x = 0, \qquad \text{for all } u = 2, \ldots, r, \tag{79}$$

where $e_{x_c}, e_{x_1} \in \mathbb{R}^p$ *are vectors of coefficients such that* $P_i e_{x_c} = X_{c,i}$*, and* $P_i e_x = x_i$ *for all* $i = 1, \ldots, n$*.*

Proof. Constraint (78) can be written as $(x_{c,i} - x_1)^2 = 0$, which can be expanded as

$$x_{c,i}^T x_{c,i} + x_1^T x_1 - 2x_{c,i}^T x_1 = 0 \quad \text{for all } i \in \mathcal{V}.$$

This constraint can be expressed as (79), by using relation (71) for the first two terms and definition of G_E^{uv} (46) for the third term. ◀

Initial conditions and performance measures

We adapt Proposition 39 to the case where there are multiple equivalence classes of agents in the PEP, in order to present usual initial conditions and their reformulation. The proposition focuses on the initial conditions, but all the expressions involved in these constraints can also be adapted as a performance measure, using the last point x^t instead of the initial point x^0. Other initial conditions or performance measures could involve only one of the equivalence classes of agents.

▶ **Proposition 45** (Initial conditions). *Let $x^* \in \mathbb{R}^d$ denote the optimal solution of the distributed optimization problem, $x_i^0 \in \mathbb{R}^d$ the initial iterate of agent i, $\bar{x}^0$ the average initial iterate and $R \in \mathbb{R}$ a constant. When the PEP is restricted to symmetric agent-class solutions $(\mathbf{f}_{\mathcal{T}}^s, G_{\mathcal{T}}^s)$ (45)–(46), here is a list of initial conditions that can be expressed with f_A^u, G_A^u, G_B^u, and G_E^{uv} (for $u, v = 1, \dots, r$):*

Initial condition	*Reformulation*
$\frac{1}{n}\sum_{i=1}^n \|x_i^0 - x^*\|^2 \leq R^2$ *for all* $i = 1, \dots, n$	$(e_{x^0} - e_{x^*})^T G_A^u (e_{x^0} - e_{x^*}) \leq R^2$, *for all* $u = 1, \dots, r$
$\frac{1}{n}\sum_{i=1}^n \|x_i^0 - x^*\|^2 \leq R^2$,	$(e_{x^0} - e_{x^*})^T G_{A,\mathcal{T}} (e_{x^0} - e_{x^*}) \leq R^2$,
$\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(x_i^0)\|^2 \leq R^2$, *for all* $i = 1, \dots, n$	$(e_{g^0})^T (G_A^u)(e_{g^0}) \leq R^2$, *for all* $u = 1, \dots, r$
$\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(x_i^0)\|^2 \leq R^2$, *for all* $i = 1, \dots, n$	$(e_{g^0})^T G_{A,\mathcal{T}} (e_{g^0}) \leq R^2$,
$\frac{1}{n}\sum_{i=1}^n \|x_i^0 - \bar{x}^0\|^2 \leq R^2$,	$(e_{x^0})^T G_{D,\mathcal{T}} (e_{x^0}) \leq R^2$,
$x_i^0 = x_j^0$ *for all* $i, j = 1, \dots, n$	$(e_{x^0})^T G_{D,\mathcal{T}} (e_{x^0}) = 0$,
$\frac{1}{n}\sum_{i=1}^n \big(f_i(\bar{x}^0) - f_i(x^*)\big) \leq R$	$\big(e_{f(\bar{x}^0)} - e_{f(x^*)}\big)^T f_{A,\mathcal{T}} \big(e_{f(\bar{x}^0)} - e_{f(x^*)}\big) \leq R$

where $f_{A,\mathcal{T}}$, $G_{A,\mathcal{T}}$ and $G_{D,\mathcal{T}}$ are defined in (70), (73) *and* (77), *and $e_{x^0}, e_{x^*}, e_{g^0} \in \mathbb{R}^p$ are vectors of coefficients such that $P_i e_{x^0} = x_i^0$, $P_i e_{x^*} = x^*$ and $P_i e_{g^0} = \nabla f_i(x_i^0)$, for $i = 1, \dots, n$.*

Proof. The proof is similar to the one of Proposition 39 but using relation (71) instead of (39), relation (76) instead of (58) and relation (69) instead of (38). ◀

References

1 Nizar Bousselmi, Julien M. Hendrickx, and François Glineur. Interpolation Conditions for Linear Operators and applications to Performance Estimation Problems. `https://arxiv.org/abs/2302.08781`, 2023.

2 Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Found. Trends Mach. Learn.*, 3:1–122, 2011.

3 Sébastien Colla. *Computer-Aided Analysis of Decentralized Optimization Methods*. PhD thesis, UCLouvain, Belgium, 2024.

4 Sebastien Colla and Julien M. Hendrickx. Automated Worst-Case Performance Analysis of Decentralized Gradient Descent. In *2021 IEEE 60th Conference on Decision and Control (CDC)*, pages 2627–2633. IEEE Press, 2021.

5 Sebastien Colla and Julien M. Hendrickx. Automated Performance Estimation for Decentralized Optimization via Network Size Independent Problems. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 5192–5199. IEEE Press, 2022.

6 Sébastien Colla and Julien M. Hendrickx. Automatic Performance Estimation for Decentralized Optimization. *IEEE Trans. Autom. Control*, 68(12):7136–7150, 2023.

7 Yoel Drori and Marc Teboulle. Performance of first-order methods for smooth convex minimization: A novel approach. *Math. Program.*, 145:451–482, 2014.

8 John C. Duchi, Alekh Agarwal, and Martin J. Wainwright. Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling. *IEEE Trans. Autom. Control*, 57(3):592–606, 2012.

9 Baptiste Goujaud, Céline Moucer, François Glineur, Julien Hendrickx, Adrien Taylor, and Aymeric Dieuleveut. PEPit: computer-assisted worst-case analyses of first-order optimization methods in Python. `https://arxiv.org/abs/2201.04040`, 2022.

10 Hanley and Chi-Kwong Li. Generalized doubly stochastic matrices and linear preservers. *Linear Multilinear Algebra*, 53:1–11, 2005.

11 Dušan Jakovetić. A unification and generalization of exact distributed first-order methods. *IEEE Trans. Signal Inf. Process. Networks*, 5(1):31–46, 2018.

12 Dmitry Kovalev, Adil Salim, and Peter Richtárik. Optimal and practical algorithms for smooth and strongly convex decentralized optimization. *Adv. Neural Inf. Process. Syst.*, 33:18342–18352, 2020.

13 Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and Design of Optimization Algorithms via Integral Quadratic Constraints. *SIAM J. Optim.*, 26(1):57–95, 2016.

14 Huan Li, Cong Fang, Wotao Yin, and Zhouchen Lin. Decentralized accelerated gradient methods with increasing penalty parameters. *IEEE Trans. Signal Process.*, 68:4855–4870, 2020.

15 Huan Li and Zhouchen Lin. Revisiting extra for smooth distributed optimization. *SIAM J. Optim.*, 30(3):1795–1821, 2020.

16 Zhi Li, Wei Shi, and Ming Yan. A Decentralized Proximal-Gradient Method With Network Independent Step-Sizes and Separated Convergence Rates. *IEEE Trans. Signal Process.*, 67(17):4494–4506, 2019.

17 Qing Ling, Yaohua Liu, Wei Shi, and Zhi Tian. Weighted ADMM for fast decentralized network optimization. *IEEE Trans. Signal Process.*, 64(22):5930–5942, 2016.

18 Angelia Nedić, Alex Olshevsky, and Michael G. Rabbat. Network Topology and Communication-Computation Tradeoffs in Decentralized Optimization. *Proc. IEEE*, 106(5):953–976, 2018.

19 Angelia Nedić, Alex Olshevsky, and Wei Shi. Achieving Geometric Convergence for Distributed Optimization Over Time-Varying Graphs. *SIAM J. Optim.*, 27(4):2597–2633, 2017.

20 Angelia Nedić, Alex Olshevsky, Wei Shi, and César A Uribe. Geometrically convergent distributed optimization with uncoordinated step-sizes. In *2017 American Control Conference (ACC)*, pages 3950–3955. IEEE Press, 2017.

21 Angelia Nedić and Asuman Ozdaglar. Distributed Subgradient Methods for Multi-Agent Optimization. *IEEE Trans. Autom. Control*, 54:48–61, 2009.

22 Guannan Qu and Na Li. Accelerated Distributed Nesterov Gradient Descent. *IEEE Trans. Autom. Control*, 65(6):2566–2581, 2020.

23 Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal Algorithms for Smooth and Strongly Convex Distributed Optimization in Networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3027–3036. PMLR, 2017.

24 Kevin Scaman, Francis Bach, Sébastien Bubeck, Yin Tat Lee, and Laurent Massoulié. Optimal Convergence Rates for Convex Distributed Optimization in Networks. *J. Mach. Learn. Res.*, 20: article no. 159 (31 pages), 2019.

25 Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. Extra: An Exact First-Order Algorithm for Decentralized Consensus Optimization. *SIAM J. Optim.*, 25(2):944–966, 2015.

26 Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. A proximal gradient algorithm for decentralized composite optimization. *IEEE Trans. Signal Process.*, 63(22):6013–6023, 2015.

27 Wei Shi, Qing Ling, Kun Yuan, Gang Wu, and Wotao Yin. On the Linear Convergence of the ADMM in Decentralized Consensus Optimization. *IEEE Trans. Signal Process.*, 62(7):1750–1761, 2014.

28 Zhuoqing Song, Lei Shi, Shi Pu, and Ming Yan. Optimal Gradient Tracking for Decentralized Optimization. `https://arxiv.org/abs/2110.05282v4`, 2024.

29 Akhil Sundararajan, Bryan Van Scoy, and Laurent Lessard. Analysis and Design of First-Order Distributed Optimization Algorithms Over Time-Varying Graphs. *IEEE Transactions on Control of Network Systems*, 7:1597–1608, 2020.

30 Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. Exact Worst-Case Performance of First-Order Methods for Composite Convex Optimization. *SIAM J. Optim.*, 27(3):1283–1313, 2017.

31 Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. Performance estimation toolbox (PESTO): Automated worst-case analysis of first-order optimization methods. In *2017 IEEE 56th Conference on Decision and Control (CDC)*, pages 1278–1283. IEEE Press, 2017.

32 Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. Smooth Strongly Convex Interpolation and Exact Worst-case Performance of First-order Methods. *Math. Program.*, 161:307–345, 2017.

33 César A Uribe, Soomin Lee, Alexander Gasnikov, and Angelia Nedić. A dual approach for optimal algorithms in distributed optimization over networks. In *2020 Information Theory and Applications Workshop (ITA)*, pages 1–37. IEEE Press, 2020.

34 Lin Xiao and Stephen Boyd. Fast linear iterations for distributed averaging. *Syst. Control Lett.*, 53(1):65–78, 2004.

35 Jinming Xu, Ye Tian, Ying Sun, and Gesualdo Scutari. Accelerated primal-dual algorithms for distributed smooth convex optimization over networks. In *International Conference on Artificial Intelligence and Statistics*, pages 2381–2391. PMLR, 2020.

36 Jinming Xu, Shanying Zhu, Yeng Chai Soh, and Lihua Xie. Augmented distributed gradient methods for multi-agent optimization under uncoordinated constant stepsizes. In *2015 IEEE 54th Conference on Decision and Control (CDC)*, pages 2055–2060. IEEE Press, 2015.