2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

A coordinator-driven communication reduction scheme for distributed optimization using the projected gradient method

Giorgos Stathopoulos[1] and Colin N. Jones[1]

Abstract— **We propose a way to estimate the value function of a convex proximal minimization problem. The scheme constructs a convex set within which the optimizer resides and iteratively refines the set every time that the value function is sampled, namely every time that the proximal minimization problem is solved exactly. The motivation stems from multi-agent distributed optimization problems, where each agent is described by a proximal minimization problem unknown to the global coordinator. We prove convergence results related to the solution of such distributed optimization problems in the special case where the projected gradient method is used and demonstrate that the developed scheme significantly reduces communication requirements when applied to a microgrid setting.**

I. Introduction

The presence of a population of independent agents with a global coordinator, along with the fact that certain data is supposed to remain private to the agents, call for distributed optimization schemes. Methods like the Proximal Gradient Method (PGM), the Alternating Direction Method of Multipliers (ADMM) and several others are suitable for these problems (see [1] and the references therein).

In the majority of these schemes, the local subproblems that need to be solved are cast as *proximal minimization problems*, a term used to describe optimization problems that are regularized by the addition of a properly scaled quadratic term in their objective [2]. In the course of the execution, the agents need to communicate the solutions to the proximal minimization subproblems to the coordinator, who will, in turn, manipulate the agents' objectives by broadcasting incentives that skew their local policies toward the global target.

In such multi-agent frameworks, extensive communication might be undesirable for a variety of reasons. One such reason might be the existence of delays due to a weak network. Another reason might be that the agents run on energy-limited resources which are drained rapidly with frequent activations. Furthermore, it is often the case that an agent's update might be insignificant relative to the global objective's value decrease, hence rendering it more beneficial to skip the update in the first place. This is a pronounced issue in (very) large scale optimization with multiple processes per machine, where the machine can only be executing one optimization at a time. It would, therefore, be useful if the coordinator could 'guess' the optimizers of its agents and base the selection of the agent to update on the satisfaction of some criterion.

We propose a reduced communication framework for distributed optimization problems by estimating a convex set containing the solution of a proximal minimization problem. Construction of the set is achieved using the theory of *the Moreau envelope function* and its important connections with the proximal operator. The structural properties of the Moreau envelope, allows the coordinator to restrict the area of all possible optimizers within a convex set, explicitly described as the intersection of ellipsoids and iteratively refined every time that a communication round occurs. Subsequently, the coordinator can make a guess regarding the solution of the agent's optimizer by choosing a value from the constructed set and spare a communication round provided that the guess is adequately good. The proposed scheme is a *coordinator-driven communication reduction scheme, i.e., the agents cannot decide when to communicate*.

Our analysis focuses on a distributed optimization algorithm instance, namely on the projected gradient method. In this context we propose a certification test for deciding whether communication should occur or not. In addition, we model the effect incurred by the lack of communication as an error in the solution of the optimization problem. By analyzing this result in the context of fixed-point iterations with errors, *we prove convergence of the sequence to either the optimizer, or to a ball centered at the optimizer, based on the convergence rate of the error sequence.*

The manuscript is organized as follows: Section II explains the notation that we use throughout this work. In the beginning of Section III we set the groundwork for the upcoming results, *i.e.*, we introduce the Moreau envelope function and its connection to the proximal operator. We then show how we can explicitly construct a set that contains an optimizer, and we propose a scheme that decides when the approximation should be updated, namely when a communication round should be triggered. At the close of the section we derive the results related to convergence. In Section IV we provide evidence about the performance of the proposed scheme by solving a load sharing problem for microgrids using the projected gradient method with block coordinate updates.

[1]The authors are with Laboratoire d'Automatique, EPFL, CH-1015 Lausanne, Switzerland

{georgios.stathopoulos,colin.jones}@epfl.ch

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 307608: BuildNet.

II. Notation

The subdifferential of an (extended-real-valued) closed proper convex function $f : \mathbb{R}^n \mapsto \mathbb{R} \cup \{+\infty\}$ at a point x is denoted by $\partial f(x)$. The *proximal operator* $\mathbf{prox}_{\gamma f} : \mathbb{R}^n \mapsto \mathbb{R}^n$ evaluated at z is denoted as $\mathbf{prox}_{\gamma f}(z) = \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ f(x) + \frac{1}{2\gamma}\|x - z\|^2 \right\}$, $\gamma > 0$. The indicator function of a closed convex set $\mathcal{Z}$ is denoted as

$$\delta(z \mid \mathcal{Z}) = \begin{cases} 0 & z \in \mathcal{Z} \\ \infty & \text{otherwise.} \end{cases}$$

When $f = \delta_{\mathcal{Z}}$, $\mathbf{prox}_{\gamma f}(z) = \mathcal{P}_{\mathcal{Z}}(z)$, where $\mathcal{P}_{\mathcal{Z}}$ denotes the projection of z onto the set $\mathcal{Z}$.

Given some $x \in \mathbb{R}^n$ and $n = \sum_{i=1}^{M} n_i$, we denote the i^{th} component of the gradient of some $h : \mathbb{R}^n \mapsto \mathbb{R}$ by $\nabla_i h(x) \in \mathbb{R}^{n_i}$.

Finally, the expected value of a random variable X is denoted by $\mathbb{E}[X]$, the conditional expectation of X given Y by $\mathbb{E}[X \mid Y]$, while the probability that $Y = y$ by $\mathbb{P}[Y = y]$.

III. Estimating the solution to a proximal minimization problem

Consider the proximal minimization problem

$$\mathbf{prox}_{\gamma f_i}(z_i) = \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ f_i(x) + \frac{1}{2\gamma}\|x - z_i\|^2 \right\} \quad , \tag{1}$$

associated to some agent $i \in \{1, \dots, N\}$. In a distributed optimization setting, a global coordinator generates a sequence of points $\{z_i^k\}$ for each agent, at which the proximal operator (1) is evaluated. The agents subsequently transmit their optimizers $\mathbf{prox}_{\gamma f_i}(z_i)$ to the coordinator.

Our purpose is to estimate, to the best possible accuracy, the optimizer of (1) for a given (arbitrary) sequence $\{z_i^k\}$, without having to solve the optimization. In other words, we want *to learn an approximation of the value function of* (1), also known as *the Moreau envelope function of* f, a notion instrumental to our upcoming analysis.

A. The Moreau envelope

The Moreau envelope of f is defined as

$$\phi^\gamma(z) = \min_{x\in\mathbb{R}^n} \left\{ f(x) + \frac{1}{2\gamma}\|x - z\|^2 \right\} \quad . \tag{2}$$

When f is a closed proper convex function, the Moreau envelope is *convex and differentiable, with Lipschitz continuous gradient with constant* $1/\gamma$. Moreover, *the set of minimizers of* f *and of* ϕ^γ *coincide*. A discussion about the envelope and its properties can be found in [3, Section 5.1]. It is also shown in [3, Proposition 5.1.7] that *the unique solution to the proximal minimization* $x^\gamma(z) = \mathbf{prox}_{\gamma f}(z)$ *can be written as*

$$x^\gamma(z) = z - \gamma \nabla \phi^\gamma(z) \quad , \tag{3}$$

i.e., it is the point at which the gradient descent iteration of ϕ^γ*, evaluated at* z*, lands.*

The Moreau envelope of a scalar linear function is depicted in Figure 1. An outer polyhedral approximation of ϕ^γ can be constructed by generating a cut every time that problem (2) is solved centered at some point z_j. The cut is a hyperplane tangent to the epigraph of ϕ^γ at z_j and is described by the equation

$$\phi^\gamma(z_j) + \langle \nabla \phi^\gamma(z_j), z - z_j \rangle \quad . \tag{4}$$

We are going to refer to points at which the Moreau envelope is evaluated (*i.e.*, points at which (2) is solved), as *query points*, the name stemming from the fact that communication has to occur between the coordinator and the agent in order to have the exact solution.

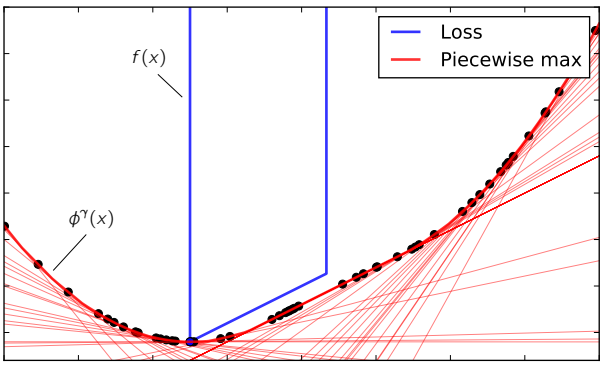

Fig. 1: The Moreau envelope of a linear function constrained in [-3,1]. The proximal evaluations at several randomly generated points gradually form the envelope. Note that the minimum values coincide.

It is evident from (3) that, given an arbitrary point v, the gradient $\nabla \phi^\gamma(v)$ is all that is needed in order to reconstruct the optimizer of (1). However, the outer polyhedral approximation of the envelope given by a collection of cuts (4) for several query points j, can only provide us with a rough estimate of the gradient. This estimate is derived by identifying the hyperplane that is closest to ϕ^γ at v, namely by finding the index

$$j^* = \underset{j}{\operatorname{argmax}} \left\{ \phi^\gamma(z_j) + \langle \nabla \phi^\gamma(z_j), v - z_j \rangle \right\}_j \quad . \tag{5}$$

The quality of the approximated gradient deteriorates with the distance of z_{j^*} to v, as well as with the rate of change of the slope of ϕ^γ. In what follows, we will get a better characterization of the possible gradients of ϕ^γ at v by making use of some fundamental properties of the gradient of a convex function, applied to the Moreau envelope (2).

B. Characterization of the gradient set

We saw in the previous section that the active hyperplane can be a poor approximation of the gradient at a point v. What we opt for is rather *a set of possible gradients and a way to evaluate the worst case gradient (how far away one can be from the actual gradient) contained in the set*. The Lipschitz continuity property of the gradient of ϕ^γ, along with its convexity, imply the existence of numerous conditions that can be used in order to construct a set for the possible gradients, see, *e.g.*, [3, Section 6.1]. One

such necessary condition for convexity is based on the *co-coercivity property* of the gradient and reads as follows:

$$\langle \nabla\phi^\gamma(v) - \nabla\phi^\gamma(z), v - z\rangle \geq \gamma\|\nabla\phi^\gamma(v) - \nabla\phi^\gamma(z)\|_2^2, \quad \forall z, v \in \mathbb{R}^n \ .$$

Consequently, a set of possible gradients of ϕ^γ at v, denoted by g, can be described by

$$\mathcal{G}(v) := \bigcap_{j=1}^{M} \Big\{ g \ \mid \ \gamma\|g - \nabla\phi^\gamma(z_j)\|_2^2 - \langle g - \nabla\phi^\gamma(z_j), v - z_j\rangle \leq 0 \Big\}. \tag{6}$$

The set is an intersection of a finite number (M) of 2-norm balls, and hence it is closed. The closedness of the set is important for finding the worst case gradient, as described in the next section.

Remark 1: Several conditions can be used instead of the 2-norm ball in (6), see, *e.g.*, [4, Theorem 2.1.5]. Some of them might perform better than others in terms of the computational description of the set when it is used for optimization purposes, and they are, in principle, equivalent (see also [3, Exercise 6.1]).

C. When should we communicate?

We will now capitalize on the knowledge that we acquired regarding the location of the actual gradient in the set (6) in order to reduce communication in a distributed optimization framework. Let us consider a problem of the form

$$\text{minimize} \quad h(z) + \sum_{i=1}^{N} f_i(z_i) \ , \tag{7}$$

where $f_i : \mathbb{R}^{n_i} \mapsto \mathbb{R} \cup \{+\infty\}, \ i = 1, \ldots, N$ are convex functions private to the corresponding agents and $h : \mathbb{R}^n \mapsto \mathbb{R} \cup \{+\infty\}$, $n = \sum_{i=1}^N n_i$, is the convex objective of the coordinator.

We are going to focus our attention on a special case of problems, namely when $f_i(z_i) = \delta(z_i \mid \mathcal{Z}_i)$, and $\mathcal{Z}_i, \ i = 1, \ldots, N$ are compact sets, h is differentiable with L-Lipschitz continuous gradient and strongly convex with modulus μ, and a relaxed version of the Projected Gradient Method is used in order to solve (7), namely the iteration becomes

$$z_i^{k+1} = (1 - \eta^k) z_i^k + \eta^k \mathcal{P}_{\mathcal{Z}_i}(z_i^k - c\nabla_i h(z^k)) \ , \tag{8}$$

with a stepsize $c \in (0, 2/L)$. The (possibly varying) relaxation parameter $\eta^k \in [0, 1]$ is widely used in fixed point iterations like (8) (see, *e.g.*, [3, Proposition A.4.2]) and can be particularly useful in cases where the iteration is inexact due to accumulated errors [5], or when asynchronicity is present in the updates [6].

The projected gradient method is a special case of a proximal minimization problem. The coordinator is agnostic to the sets $\mathcal{Z}_i$, and routinely needs to communicate the gradient $\nabla h(z^k)$ and subsequently wait to receive z_i^{k+1}.

It is customary in many distributed optimization applications to let the agents update only one at a time, in contrast to waiting for all of them to become available. Under this assumption, and without restricting the applicability of the scheme we are about to propose, *only one agent updates at each iteration, thus we operate in a (block) coordinate descent fashion.* Adopting the framework for distributed asynchronous optimization proposed in [6], a probability of update $p_i, \ i = 1, \ldots, N$ is assigned to each agent. At every algorithmic iteration, agent i_k is selected to update according to its probability. The relaxation parameter η^k can be set to $\eta^k = \frac{\rho}{N p_{i_k}}$, for some $\rho > 0$.

Given (6), we can now devise a reduced communication scheme to solve the problem. We have from (3) that

$$\mathcal{P}_{\mathcal{Z}_i}(v_i^k) = v_i^k - \gamma\nabla\phi_i^\gamma(v_i^k) \ ,$$

where $v_i^k = z_i^k - c\nabla_i h(z^k)$. The coordinator can, thus, keep in its memory a set $\mathcal{G}_i$ for every agent i, that is updated whenever a communication round occurs. With every query point, $\mathcal{G}_i$ is augmented by one more inequality according to (6). Observe that $\mathcal{G}_i$ acts indeed as a subdifferential set, namely *it contains gradients g_i that might not result in an objective value decrease for h (see [3, Section 3.2])*. Say, *e.g.*, that at time k a point g_i^k is picked from $\mathcal{G}_i$ and $g_i^k \neq \nabla\phi_i^\gamma(v_i^k)$. Then the estimated optimizer becomes $\hat{z}_i^{k+1} = (1 - \eta^k) z_i^k + \eta^k (v_i^k - \gamma g_i^k)$, and $z^{k+1} = (\ldots, z_{i-1}^{k+1}, \hat{z}_i^{k+1}, z_{i+1}^{k+1}, \ldots)$. This choice of z^{k+1} might result in $\langle \nabla h(z^k), z^{k+1} - z^k\rangle > 0$ due to the fact that $\langle \nabla_i h(z^k), \hat{z}_i^{k+1} - z_i^k\rangle > -\sum_{j\neq i}\langle \nabla_j h(z^k), z_j^{k+1} - z_j^k\rangle$.

Based on the discussion above, *the coordinator can always look for the worst case point in $\mathcal{G}_i$ relative to the decrease of h. If the resulting estimated optimizer $\hat{z}_i^{k+1}$ preserves the property of a descent direction, then there is no need to query agent i. In the opposite case, communication is preferable.*

The worst case optimizer can be found be solving the following Quadratically Constrained Quadratic Program (QCQP):

$$\begin{aligned} \text{maximize} \quad & \Big\langle \begin{bmatrix} \vdots \\ \nabla_i h(z^k) \\ \vdots \end{bmatrix}, \begin{bmatrix} \vdots \\ \eta^k(v_i^k - z_i^k) - \gamma g_i \\ \vdots \end{bmatrix} \Big\rangle =: H_i(v_i^k) \\ \text{subject to} \quad & g_i \in \mathcal{G}_i(v_i^k) \ , \end{aligned} \tag{9}$$

with variable $g_i \in \mathbb{R}^{n_i}$. The sequence of steps taken to solve problem (7) is presented below as Algorithm 1.

Looking at the 5^{th} step of Algorithm 1, the gradient vector g_i can be chosen in an arbitrary manner. When g_i is chosen to be the solution of (9), the algorithm tends to behave conservatively since that would correspond to having agents that are adversarial to the global coordinator. In a more realistic setting, however, the actual gradient $\nabla\phi_i^\gamma(v)$ of some agent i would probably lie somewhere in the set $\mathcal{G}_i$, rather than on its worst case boundary point.

Consequently, we propose an alternative approach, where z_i^{k+1} of Algorithm 1 is updated based on the g_i^k that resides *in the Chebyshev center of the set $\mathcal{G}_i$, i.e., g_i^k is the center of the largest ball inscribed in the set $\mathcal{G}_i$*. The center can be computed by solving an additional Semidefinite Program (SDP) [7, Section 8.5.1].

Remark 2: Although the QCQPs and the SDPs that need to be solved from the coordinator at every algorithmic

Algorithm 1 Projected Gradient Method with Estimated Proximal Operator

Require: $z_{i,0} \in \mathbb{R}^{n_i}$, $\mathcal{G}_i = \varnothing$, $c \in (0, 2/L)$, $\gamma = c$, discrete probability distribution $(p_1, \ldots, p_N)$, $\sum_{i=1}^{N} p_i = 1$, $p_i > 0$. Iteration counter is set to $k = 0$, $k_{\text{stop}} > 0$.

while $k < k_{\text{stop}}$, choose agent i_k with probability $p_{i_k} := \mathbb{P}[i_k = i] = p_i$, **do**
  Compute $v_i^k = z_i^k - c\nabla_i h(z^k)$ ▷ Coordinator
  Solve (9) ▷ Coordinator
  if $H_i(v_i^k) < 0$ **then**
    Choose any $g_i \in \mathcal{G}_i$, set $g_i^k = g_i$ ▷ Coordinator
    Compute $z_i^{k+1} = (1 - \eta^k) z_i^k + \eta^k (v_i^k - \gamma g_i^k)$ ▷ Coordinator
  else
    Transmit v_i^k to Agent i ▷ Coordinator
    Solve (8) ▷ Agent i
    Transmit z_i^{k+1} to the Coordinator ▷ Agent i
    Update set $\mathcal{G}_i$ with z_i^{k+1} ▷ Coordinator
  $k \leftarrow k + 1$

iteration grow with the number of query points, *both the ellipsoidal and SDP constraints that are added with every new query point are independent of the existing ones.* This suggests that the optimization problems are separable and can be solved in a parallel fashion by employing as many processors as the number of the generated query points.

D. Convergence

The convex program (9) is a communication test drawing on the validity of the gradient estimate g_i^k, for some agent $i \in \{1, \ldots, N\}$, $g_i^k \neq \nabla \phi_i^\gamma(v_i^k)$. For as long as the test is passed, the scheme will keep iterating toward the right direction with respect to the cost decrease, but with some error in the local optimizers. These errors accumulate as Algorithm 1 progresses. It is, therefore, essential for the coordinator to *take corrective action in the form of feedback from the agents, so as to avoid drifting toward a non-solution of* (7).

Let us assume that every $K > 0$ iterations a full correction takes place, namely all the agents communicate their actual optimizers to the coordinator, and that K is bounded. In this case, the following theorem holds:

Theorem 1: Let N agents update with probabilities p_i and $p_{\min} = \min_i p_i$. Let $\nu = 1 - \sqrt{(1 - 2\gamma\mu + \mu\gamma^2 L)}$, where L is the Lipschitz constant of ∇h and μ the strong convexity constant of h, while $\gamma < 2/L$. If z^* is the unique optimizer of (7), for any time instant $k > K$, the sequence $\{z^k\}$ generated by Algorithm 1 satisfies

$$\mathbb{E}[\|z^{k+K} - z^*\|^2] \leq \left(1 - \tfrac{\rho(\nu-\epsilon)}{N}\right)^k \mathbb{E}[\|z^K - z^*\|^2] + \tfrac{\rho}{N}\left(\tfrac{1}{\epsilon} + \tfrac{\rho(1+\delta)}{N p_{\min}\delta}\right) \sum_{j=1}^{k} \left(1 - \tfrac{\rho(\nu-\epsilon)}{N}\right)^{k-j} \mathbb{E}[\|e^{K-1+j}\|^2] \ , \tag{10}$$

for $\rho \in (0, N p_{\min}/(2(1+\delta)))$, $\delta > 0$, $\nu > \epsilon > 0$ and $e^k = (e_1^k, \ldots, e_N^k) \in \mathbb{R}^{Nn}$ the vector that is constituted of the components $e_i^k = \gamma(\nabla \phi_i^\gamma(v^k) - g_i^k)$, $i = 1, \ldots, N$, while $e^K = 0$.

Proof: We sketch the idea behind the proof in the Appendix. Due to its excessive length, the complete proof has been deferred to the supplementary document [8]. ■

Theorem 1 states that if the algorithm is terminated earlier, the sequence $\{z^k\}$ will converge (in expectation) to a ball that is centered at the optimum and has a radius that increases with the expected error that has been accumulated since the last correction occured.

Remark 3: Our focus on coordinate projected gradient descent does not affect the generalizability of the approach. As a matter of fact, convergence of the proximal gradient method with errors in the classical setting can be found in several works (*e.g.*, [5]). In that setting, the coordinator would need to solve (9) for the vector $g = (g_1, \ldots, g_N) \in \mathbb{R}^{Nn}$, and subsequently decide which agents should be communicated based on some heuristic criterion.

IV. Application: Distributed Load Sharing

This section considers the application of Algorithm 1 to a problem where the minimizer of a convex function lies in the Cartesian product of a number of convex sets. This classical problem is of particular interest in several contexts. We consider the problem of cooperative tracking of a reference signal from a population of buildings. These problems typically arise in the context of microgrids, where a mixture of energy generation, energy storage elements and loads are coupled together in order to satisfy a predicted power demand profile, as is the case in day ahead electricity markets.

The sharing problem takes the form

$$\text{minimize} \quad \frac{1}{2}\sum_{t=0}^{T-1}\Big(\sum_{i=1}^{N} z_i(t) - r(t)\Big)^2 + \sum_{i=1}^{N} f_i(z_i) \ , \tag{11}$$

with variables $z_i = (z_i(0), \ldots, z_i(T-1)) \in \mathbb{R}^T$ and $z = (z_1, \ldots, z_N) \in \mathbb{R}^{NT}$. The reference power profile is denoted by $r(t)$, while time spans from $t = 0, \ldots, T-1$, *i.e.*, we have a T-timesteps ahead prediction of the power profile. The variable $z_i(t)$ refers to the total consumption of the i^{th} building at time instant t, where $i = 1, \ldots, N$. The first term, $h(z) = (1/2)\sum_{t=0}^{T-1}(\sum_{i=1}^{N} z_i(t) - r(t))^2$, penalizes the deviation of the total power contribution to the reference power profile. The individual components $f_i(z_i)$ are implicit descriptions of convex sets, constructed by the intersection of linear equations (agent dynamics) and constraints, details that are hidden from the global node.

A. Modeling of the agents

The microgrid comprises small, medium and large office buildings, generated by the OpenBuild software [9]. The buildings are described as linear dynamical systems, the input to which is the thermal heat (kW) that is entering or leaving each zone, while the output is the temperature at each zone (C°). The energy conversion systems (electrical to thermal) is modeled as a static map, which is represented by a constant

coefficient of performance (COP). The buildings can participate in the ancillary service market by *increasing or decreasing their consumption with respect to some baseline power profile*. An individual building seeks to contribute to the tracking objective while respecting temperature constraints. The local optimization problem for building i becomes

$$g_i(z_i, u_i, x_i) = \delta((z_i, u_i, x_i) \mid \mathcal{C}_{\text{build}}) \ , \tag{12}$$

$$\mathcal{C}_{\text{build}} = \left\{ \begin{array}{l} x_i(t+1) = A_i x_i(t) + B_i u_i(t) \\ x_i(0) = x_i^{\text{init}} \\ z_i(t) = \sum_{j=1}^{M_i} u_{ij}(t) \\ C_i x_i(t) \in X_i(t) \\ \|u_i(t)\|_\infty \leq u_i^{\max} \end{array} \right\} \ ,$$

with $z_i \in \mathbb{R}^T, x_i \in \mathbb{R}^{nT}, u_i = \{u_{ij}\}_{1 \leq j \leq M_i} \in \mathbb{R}^{M_i T}$, where x_i are states $z_i(t)$ is the total amount (electrical equivalent) of the thermal consumption at time t, and M_i is the number of zones of the building. The linear mapping of the states $C_i x_i(t)$ corresponds to zone temperatures that are confined within desired limits.

Simulation characteristics

Data	20^{th} July 2013			
Location	Lausanne			
Time	00:00 - 04:00			
Sampling time	15			min
Horizon	17			–
Buildings				
Minimum temperature	18			°C
Maximum temperature	28			°C
Heat pump COP_{hot}	3.0			–
Heat pump COP_{cold}	3.0			–
	Small	**Medium**	**Large**	
Number of systems (Case A, B)	9	7	4	#
Area	511	4982	46320	m^2
Number of states	15	54	57	–
Number of inputs	5	18	19	–
Average thermal consumption	4	40	75	W/m^2

TABLE I: Micro-grid case study overview

Instead of carrying the full building description (12), which would result in a high-dimensional optimization problem, the authors in [10] propose a low-dimensional modeling abstraction of the building as a 'thermal battery'. A robust optimization problem is solved to ensure that the trackable power profiles $z_i(t) = \sum_{j=1}^{M_i} u_{ij}(t) \in \mathbb{R}^T$ for building i reside inside a convex set, namely $\mathcal{Z}_i$, and satisfy the constraints imposed in (12). Making use of this abstraction, the problem we want to solve reads:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2} \sum_{t=0}^{T-1} (\sum_{i=1}^{N} z_i(t) - r(t))^2 + \frac{\alpha}{2} \|z\|^2 \\ \text{subject to} & z_i \in \mathcal{Z}_i, \ i = 1, \ldots, N \ , \end{array} \tag{13}$$

where the term $(\alpha/2)\|z\|^2$ was added for regularization purposes using $\alpha = 0.1$.

B. Simulation setup

Our purpose is to assign 15 minute reference tracking to a population that consists of agents described in the previous section. We consider $N = 20$ buildings, while all the details of the simulation are given in Table I.

We solve problem (13) using the projected gradient method with random coordinate descent. Uniform updates are considered, *i.e.*, $p_i = 1/N$ while we set $\eta^k = \eta = 0.9 \ \forall k$ (its value should be smaller according to Theorem 1, but the formula is rather conservative, while the aforementioned choice performed well in practice). A comparison is performed between the exact solution of the problem, and the worst case and centering approaches proposed in Section IV. The proximal minimization problems are solved in MATLAB using the YALMIP optimizer [11] with the Gurobi solver, while the QCQPs and the SDPs were solved using CVX [12] with SEDUMI [13].

Figure 2 depicts the number of iterations versus the number of communication rounds using all three approaches, namely exact, inexact worst case and inexact with centering. The termination criterion is a combination of feasibility and optimality, namely we iterate until inclusion of z_i in $\mathcal{Z}_i$ is satisfied for all agents with accuracy 10^{-3}, while the distance to the optimizer $\|z^k - z^*\|/\|z^*\| \leq 10^{-1}$. It is observed that in both cases where the proximal solution is estimated, the communication rounds are significantly reduced, by more than 50%. In addition, with the centering approach the number of iterations for convergence is also reduced by more than 50%. Surprising though it might seem, this is possible since the estimated gradient g_i^k might result in an infeasible approximate optimizer z_i^{k+1} for some i, such that $z_i^{k+1} \notin \mathcal{Z}_i$, thus giving rise to a more aggressive step.

Furthermore, several levels of feasibility and optimality are depicted with crosses and diamonds, respectively. It is noteworthy that feasibility is more of an issue in the worst case approach, while the iterates become feasible much quicker when the centering approach is used.

V. Conclusion

Modern multi-agent setups consist of several heterogeneous components equipped with prediction and control algorithms, that attribute to them some decision making capacity. These attributes ask for distributed solutions which come with an inherent communication overhead. We propose a framework where the communication requests are reduced by enabling the central coordinator to gradually 'learn' the optimization model of the agents and triggers them based on the result of a predesigned certification test.

The proposed approach is currently limited by two aspects: First, the complexity of the optimization problems that need to be solved in order to recover the approximate minimizer scales with the number of the generated query points. Second, both the test design and the demonstration regard the projected gradient method. We reckon that both these aspects give rise to practical limitations, which we will try to address as part of future work.

References

[1] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, 2011.

[2] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

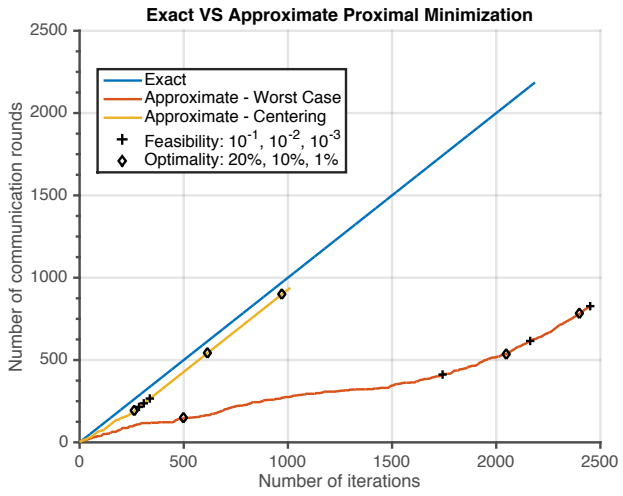

Fig. 2: Worst case: 12% increase in iterations, 62% reduction in communication. Centering: 54% reduction in iterations, 57% reduction in communication. The feasibility measure corresponds to the maximum violation of any of the constraints associated to the 20 agents. Optimality is measured with respect to the relative distance from the optimizer z^*. The endpoint of all curves corresponds to the iteration that both criteria were satisfied at a specified accuracy, *i.e.*, 10^{-1} for the relative optimality and 10^{-3} for feasibility.

[3] D. P. Bertsekas, *Convex Optimization Algorithms*. Athena Scientific, 2015.

[4] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2004.

[5] H. Raguet, J. Fadili, and G. Peyré, "A Generalized Forward-Backward Splitting," *SIAM Journal on Imaging Sciences*, vol. 6, no. 3, pp. 1199–1226, 2013.

[6] Z. Peng, Y. Xu, M. Yan, and W. Yin, "ARock: an Algorithmic Framework for Asynchronous Parallel Coordinate Updates," *SIAM Journal on Scientific Computing* , vol. 38, no. 5, 2016.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[8] G. Stathopoulos and C. N. Jones, "Supplementary material for the paper: A coordinator-driven communication reduction scheme for distributed optimization using the projected gradient method,." `https://infoscience.epfl.ch/record/253180?&amp;ln=fr`, 2018.

[9] T. T. Gorecki, F. A. Qureshi, and C. N. Jones, "Openbuild: An integrated simulation environment for building control," 2015.

[10] T. T. Gorecki, A. Bitlislioglu, G. Stathopoulos, and C. N. Jones, "Guaranteeing input tracking for constrained systems: theory and application to demand response," in *American Control Conference (ACC).*, pp. 232–237, 2015.

[11] J. Löfberg, "Yalmip : A toolbox for modeling and optimization in MATLAB," in *Proceedings of the CACSD Conference*, (Taipei, Taiwan), 2004.

[12] I. CVX Research, "CVX: Matlab software for disciplined convex programming, version 2.0." `http://cvxr.com/cvx`, Aug. 2012.

[13] J. F. Sturm, "Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones," 1998.

[14] J. Liang, J. Fadili, and G. Peyré, "Convergence rates with inexact non-expansive operators," *Mathematical Programming*, pp. 1–32, 2014.

[15] M. W. Schmidt, N. L. Roux, and F. R. Bach, "Convergence Rates of Inexact Proximal-Gradient Methods for Convex Optimization," in *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, Granada, Spain.*, pp. 1458–1466, 2011.

APPENDIX
SKETCH OF THE PROOF OF THEOREM 1

The key point is to observe that the approximate iteration $z_i^{k+1} = v_i^k - \gamma g_i^k$ can be expressed as *an inexact projected gradient iteration*. To this end, we introduce the error sequence $\{e^k\}$ so as to write

$$e_i^k + \mathcal{P}_{\mathcal{Z}_i}\left(v_i^k\right) = v_i^k - \gamma g_i^k \ , \tag{14}$$

while

$$\mathcal{P}_{\mathcal{Z}_i}\left(v_i^k\right) = v_i^k - \gamma \nabla \phi_i^\gamma(v_i^k) \ . \tag{15}$$

Substituting (15) in (14) we have that

$$e_i^k = \gamma(\nabla \phi_i^\gamma(v_{i,k}) - g_i^k) \ . \tag{16}$$

Using the error (16), the randomized coordinate descent iteration can be expressed as

$$\begin{cases} z_{i_k}^{k+1} &= z_{i_k}^k + \eta^k \left(e_{i_k}^k + \mathcal{P}_{\mathcal{Z}_{i_k}}\left(z_{i_k}^k - c\nabla_{i_k} h(z^k)\right) - z_{i_k}^k \right) \\ z_{i\neq i_k}^{k+1} &= z_{i\neq i_k}^k \ , \end{cases}$$

or, more compactly, as

$$z^{k+1} = z^k + \eta^k U_{i_k}\left(\mathcal{P}_{\mathcal{Z}}\left(z^k - c\nabla h(z^k)\right) - z^k + e^k\right) \ . \tag{17}$$

The matrix $U_{i_k} : \mathbb{R}^{Nn} \mapsto \mathbb{R}^{Nn}$ is drawn from a set of orthogonal projection matrices $\{U_i\}_{1\leq i\leq N}$ such that $U_i : z \mapsto (0,\dots,0,z_i,0,\dots,0),\ i = 1,\dots,N$ and $\sum_{i=1}^N U_i = I_{Nn}$. Consequently, U_{i_k} isolates the i_k^{th} component of its argument, thus it updates the corresponding component of z, while the other components (agents) are set to their previous values. The projection operator $\mathcal{P}_{\mathcal{Z}}$ is defined as $\mathcal{P}_{\mathcal{Z}} = \mathcal{P}_{\mathcal{Z}_1} \times \mathcal{P}_{\mathcal{Z}_2} \times \cdots \times \mathcal{P}_{\mathcal{Z}_N}$.

Equation (17) is an instance of a more general *inexact fixed-point iteration*. Such iterations have been heavily studied in the literature. The works [14], [5], [15] consider the case of deterministic fixed-point iterations with errors both in the gradient and in the proximal operator, where the errors are summable, while [6] analyze asynchronous fixed-point iterations, where the errors appear due to outdated samples in the update. It turns out that the asynchronous iteration in the latter works takes the form (17), the only difference being the expression for the error (16). We can thus employ similar arguments for proving convergence. Since the proof is relatively lengthy, we have placed it as supplementary material in the manuscript [8].