Decomposed resolution of finite-state aggregative optimal control problems[*]

Kang Liu[†] Nadia Oudjane[‡] Laurent Pfeiffer[§]

Abstract

A class of finite-state and discrete-time optimal control problems is introduced. The problems involve a large number of agents with independent dynamics, which interact through an aggregative term in the cost function. The problems are intractable by dynamic programming. We describe and analyze a decomposition method that only necessitates to solve at each iteration small-scale and independent optimal control problems associated with each single agent. When the number of agents is large, the convergence of the method to a nearly optimal solution is ensured, despite the absence of convexity of the problem. The procedure is based on a method called Stochastic Frank-Wolfe algorithm, designed for general nonconvex aggregative optimization problems. Numerical results are presented, for a toy model of the charging management of a battery fleet.

1 Introduction.

This article is dedicated to a class of aggregative optimal control problems in discrete time and discrete state space. These problems involve a large number N of agents, indexed by $i = 1, \dots, N$, and time steps ranging over $t = 0, 1, \dots, T$. For any agent i, we fix a finite *state set* S_i and a finite *control set* U_i. The evolution of agent i is described by *transition functions* $\pi_i^t \colon S_i \times U_i \to S_i$, where $t = 0, \dots, T$. We also fix mappings $U_i^t \colon S_i \to 2^{U_i}$ describing the feasible controls of the agents: at time t, if the agent i is in state s_i^t, he can make use of all controls in $U_i^t(s_i^t)$. The initial state of each agent i is constrained to be in S_i^0, a subset of S_i. The problem also involves some functions $f_t \colon \mathbb{R} \to \mathbb{R}$ which we call *social* cost (at time t) and some functions $h_i^t \colon S_i^t \times U_i \to \mathbb{R}$, which we call *contribution* functions. We also make use of functions $\ell_i^t \colon S_i^t \times U_i \to \mathbb{R}$, which we call *individual* costs. The optimal control problem of interest writes:

(1.1)
$$\begin{cases} \inf\limits_{(s,u)} & J(s,u) := \sum_{t=0}^T f_t\Big(\frac{1}{N}\sum_{i=1}^N h_i^t(s_i^t,u_i^t)\Big) \\ & \qquad + \frac{1}{N}\sum_{i=1}^N \sum_{t=0}^T \ell_i^t(s_i^t,u_i^t), \\ \text{s.t.} & s_i^{t+1} = \pi_i^t(s_i^t,u_i^t),\ u_i^t \in U_i^t(s_i^t),\ s_i^0 \in S_i^0, \\ & \forall t = 0,1,\dots,T-1,\ i = 1,2,\dots,N, \end{cases}$$

where $(s,u) = (s_i^t, u_i^t)_{i=1,\dots,N}^{t=0,\dots,T}$. The problem is motivated by energy management problems involving flexibilities: they are small consumption (or production) units, which are able to shift their production or their consumption over time, typically by storing energy. We refer the reader to [13, 18]. In these models, the dynamical systems are usually continuous in time and space, they can however be discretized in problems in the form (1.1). We focus here on fully discrete problems for simplicity, but also to emphasize the fact that our approach does not require any assumption on the dynamics of the agents: they could result from a discretization of a non-linear system, involving non-smooth terms. We will only require that the functions f_t are convex, with a Lipschitz-continuous gradient.

A standard approach to deal with problem (1.1) relies on the dynamic programming principle (see [3]), in which a key step is to compute the value function $V \colon \{0,1,\dots,T\} \times S \to \mathbb{R}$, where the state space S is defined by $\prod_{i=1}^N S_i$. For our problem, Bellman's equation writes as follows: for any $t \in \{0, \dots T\}$, for any $s \in S$,

$$V^t(s) = \min_{u \in U^t(s)} f_t\Big(\frac{1}{N}\sum_{i=1}^N h_i^t(s_i,u_i)\Big) + \frac{1}{N}\sum_{i=1}^N \ell^t(s_i,u_i) + V^{t+1}(\pi^t(s,u)),$$

where $U^t(s) = \prod_{i=1}^N U_i^t(s_i)$ and where $\pi^t(s,u) = (\pi_i^t(s_i,u_i))_{i=1}^N$. We observe that the complexity of Bellman's equation increases exponentially with N; this phenomenon is the well-known curse of dimensionality. As a consequence, the dynamic programming approach is not tractable for problem (1.1) when the number of agents N is large. Let us mention here that problem (1.1) can be formulated as a large-scale mixed integer

[*]This article benefited from the support of the FMJH Program PGMO and from the support to this program from EDF. N. Oudjane was partially supported by the FiME Lab Research Initiative (Institut Europlace de Finance).

[†]CMAP, Ecole Polytechnique, CNRS, Institut Polytechnique de Paris, Palaiseau, France and Inria, Laboratoire des Signaux et des Systèmes, CentraleSupélec, CNRS, Université Paris-Saclay, Gif-sur-Yvette, France, kang.liu@polytechnique.edu

[‡]OSIRIS department, EDF Lab, Paris-Saclay, France, nadia.oudjane@edf.fr

[§]Inria, Laboratoire des Signaux et des Systèmes, CentraleSupélec, CNRS, Université Paris-Saclay, Gif-sur-Yvette, France, laurent.pfeiffer@inria.fr

convex program (MICP), that is to say, an optimization problem with integrity constraints which becomes a convex program if the integrity constraints are removed. We refer to [10] and to the references therein for the resolution of such problems. However, the number of variables in the MICP corresponding to our problem is equal to $\sum_{i=1}^N T|S_i||U_i|$ and is quickly prohibitive for solvers such as GUROBI [12] and SCIP [6] as N increases.

Another general approach for solving large-scale optimal control problems (in particular) and large-scale optimization problems with an aggregative structure (in general) relies on decomposition. Decomposition methods are particularly powerful when the associated Lagrangian has a separable structure, in which case the dual criterion takes the form of a sum that can be evaluated in parallel, making the resolution of the dual problem easier, with dual subgradient methods [4, Chapter 6], with cutting plane algorithms [14, Chapter XII, Section 4] or with the alternating direction of multipliers method (ADMM) [5]. Such approaches have been successfully applied to stochastic optimal control problem, see for example [1, 8, 18] However, the convergence of such methods is only guaranteed in the case of convex optimization problems, in general, which limits their application to linear dynamical systems.

This article provides a full description of a decomposition method that allows to circumvent the curse of dimensionality. Its convergence is ensured, despite the lack of convexity of the problem, when the number of agents is sufficiently large. At each iteration, only small-scale optimal control problems associated with each agents need to be solved. Our method relies on the Stochastic Frank-Wolfe (SFW) algorithm proposed by the authors in the recent work [7], dedicated to abstract aggregative optimization problems, as defined in [20]. The SFW algorithm is a combination of (i) the classical Frank-Wolfe algorithm, which is applied to a relaxed version of the optimization problem, with (ii) a selection method that allows to recover a solution to the original problem. The SFW relies on an oracle, which must solve efficiently some subproblems associated with each agent. It turns out that in the present context, those subproblems are small-scale optimal control problems which can be solved by dynamic programming. A key point in the analysis of the SFW algorithm is the fact that the relaxed problem is a good approximation of the original problem. We have obtained an estimate for the relaxation gap in [7] of order $\mathcal{O}(1/N)$, improving previous results by Wang in [20], when N is large. Let us note that another numerical method for aggregative problems is proposed in that reference, relying on Shapley-Folkman decompositions [19]. We refer the reader to [7, Section 5.1] for a thorough comparison of the two methods.

This article is organized as follows. Sections 2 to 4 provide the reader with a summary of the SFW algorithm and some of the theoretical findings of the article [7]. In Section 2, we give a general formulation of aggregative optimization problems. Their convex relaxation is introduced in Section 3. We decribe the SFW algorithm and its convergence properties in Section 4. Section 5 provides a reformulation of the optimal control problem (1.1) as a general aggregative problem of the form (P) and details the implementation of the SFW algorithm for (1.1). We briefly discuss the MICP approach in Section 6. Numerical simulations on the charging management of a battery fleet are presented in Section 7.

2 Abstract aggregative problems

The general aggregative optimization problem investigated in [20] and [7] is given by

$$\text{(P)} \quad \inf_{x \in \mathcal{X}} J(x) := f(G(x)),$$
$$\text{where: } \mathcal{X} = \prod_{i=1}^N \mathcal{X}_i \quad \text{and} \quad G(x) = \frac{1}{N} \sum_{i=1}^N g_i(x_i).$$

The sets $\mathcal{X}_i$ are given and the maps g_i are defined from $\mathcal{X}_i$ to some Hilbert space $\mathcal{E}$. The function f is defined from $\mathcal{E}$ to $\mathbb{R}$. An interpretation of problem (P) is as follows: N is the number of agents; the agents are indexed by i and each variable $x_i \in \mathcal{X}_i$ corresponds to the decision attributed to agent i. The mapping g_i is the contribution of agent i to some common good, defined by $\frac{1}{N}\sum_{i=1}^N g_i(x_i)$. We will refer to it as the aggregate. The function f is the social cost associated with the aggregate. In addition to applications in aggregative optimal control mentioned in the introduction, problem (P) itself has important applications in various domains, such as the *resource allocation problems* [2, 15], supervised learning problems, see [9, 17, 16].

We make some non-restrictive structural assumptions on (P). The aggregate space $\mathcal{E}$ is assumed to be the Cartesian product of M Hilbert spaces $\mathcal{E}_1, \ldots, \mathcal{E}_M$. Moreover, we suppose that f is of the form

$$f(y) = \sum_{j=1}^M f_j(y_j), \quad \forall y = (y_1, \ldots, y_M) \in \mathcal{E}.$$

The functions f_j are defined from $\mathcal{E}_j$ to $\mathbb{R}$, for $j = 1, \ldots, M$. Finally, the contribution mappings g_i are of the form $g_i(x_i) = (g_{ij}(x_i))_{j=1,\ldots,M}$. Therefore, the cost functional J writes:

$$J(x) = \sum_{j=1}^{M} f_j \left(\frac{1}{N} \sum_{i=1}^{N} g_{ij}(x_i) \right).$$

Some general notations will be used all along the article: given two subsets A and $B \subseteq \mathcal{E}$, we denote by $A+B$ their Minkowski sum. Given $\lambda \in \mathbb{R}$, we denote λA the set defined by $\{\lambda a \mid a \in A\}$. We denote by $\operatorname{conv}(A)$ the convex hull of A. Next we introduce the main assumptions of the work. For any $i = 1, \dots, N$ and for any $j = 1, \dots, M$, we denote

$$Y_{ij} = \big\{ g_{ij}(x_i) \mid x_i \in \mathcal{X}_i \big\} \quad \text{and} \quad Y_j = \frac{1}{N} \sum_{i=1}^{N} Y_{ij}.$$

Assumption A. *For $i = 1, 2, \dots, N$ and $j = 1, 2 \dots, M$:*

- *The range set Y_{ij} has finite diameter d_{ij}.*
- *The function f_j is L_j-Lipschitz on* $\operatorname{conv}(Y_j)$.
- *The function f_j is continuously differentiable on a neighborhood of* $\operatorname{conv}(Y_j)$, *and ∇f_j is $\tilde{L}_j-$Lipschitz on* $\operatorname{conv}(Y_j)$.

We next define two constants C_0 and C_1 by

$$C_0 = \sum_{j=1}^{M} \Big(L_j \max_{1 \leq i \leq N} \{ d_{ij} \} \Big),$$
$$C_1 = \frac{1}{N} \sum_{j=1}^{M} \left(\tilde{L}_j \sum_{i=1}^{N} d_{ij}^2 \right).$$

Assumption B. *For all $j = 1, \dots, M$, the function $f_j \colon \mathcal{E}_j \to \mathbb{R}$ is convex.*

Assumption C. *For all $i = 1, \dots, N$ and for all $y \in$* $\operatorname{conv}(G(\mathcal{X}))$, *the problem*

$$\inf_{x_i \in \mathcal{X}_i} \langle \nabla f(y), g_i(x_i) \rangle \tag{2.2}$$

has at least a solution. For all $i = 1, \dots, N$, we fix a map $\mathbb{S}_i \colon$ $\operatorname{conv}(G(\mathcal{X})) \mapsto \mathcal{X}_i$ *such that, for any $y \in$* $\operatorname{conv}(G(\mathcal{X}))$, *$\mathbb{S}_i(y)$ is a solution to* (2.2).

3 Convex relaxation

We introduce in this section a convex relaxation of problem (P). It will motivate the stochastic Frank-Wolfe algorithm presented in the following section. The reader only interested in a practical implementation of the algorithm can move to the next section. We first need to reformulate problem (P). Let us define

$$\mathcal{Y}_i = g_i(\mathcal{X}_i), \quad \forall i = 1, \dots, N, \quad \text{and} \quad \mathcal{Y} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{Y}_i.$$

Problem (P) is equivalent to

$$\inf_{y \in \mathcal{E}} f(y), \quad \text{subject to: } y \in \mathcal{Y}.$$

Indeed, by definition of $\mathcal{Y}$, any $y \in \mathcal{E}$ lies in $\mathcal{Y}$ if and only if there exists $x \in \mathcal{X}$ such that $y = \frac{1}{N} \sum_{i=1}^{N} g_i(x_i)$. For such an x, we have $f(y) = J(x)$. It is natural to consider the following relaxation:

$$\text{(PR)} \qquad \inf_{y \in \mathcal{E}} f(y), \quad \text{subject to: } y \in \operatorname{conv}(\mathcal{Y}).$$

Under Assumption B, the relaxed problem is a convex optimization problem. Let J^* denote the value of problem (P) and let $\mathcal{J}^*$ denote the value of problem (PR). We have the following result.

Proposition 3.1. *Let Assumption A hold true. Then*

$$\mathcal{J}^* \leq J^* \leq \mathcal{J}^* + \frac{C_1}{2N}.$$

Proof. The first inequality is straightforward, and the second one is proved in [7, Proposition 2.6]. □

Let us mention that a more precise upper bound is given in [7, Theorem 4.4]. The result of Proposition 3.1 is to be related to Shapley-Folkman's lemma [19] and more precisely to Starr's corollary [19], which gives a bound of the distance to $\mathcal{Y}$ of a point in $\operatorname{conv}(\mathcal{Y})$. Assuming that the coefficients d_{ij} (appearing in Assumption A) are uniformly bounded, we see that C_1 is also bounded, and thus the gap estimate $\frac{C_1}{2N}$ goes to zero as N goes to infinity. In words, there is a convexification of the problem as the number of agents increases.

Remark 3.1. *Consider the particular case where the sets $\mathcal{X}_i$ and the contribution functions g_i do not depend on i. Then, $\mathcal{Y}_1 = \ldots = \mathcal{Y}_N$. It follows that*

$$\operatorname{conv}(\mathcal{Y}) = \operatorname{conv}\left(\frac{1}{N} \sum_{i=1}^{N} \mathcal{Y}_1 \right) = \frac{1}{N} \sum_{i=1}^{N} \operatorname{conv}(\mathcal{Y}_1)$$
$$= \operatorname{conv}(\mathcal{Y}_1).$$

In this case, the relaxed problem does not depend on N, it can be interpreted as a mean-field relaxation, i.e., it can be interpreted as the limit problem as the number of agents $N \to \infty$.

From Proposition 3.1, we note that any ϵ-solution of problem (PR) is an ϵ-solution of problem (P) as soon as it is feasible for problem (P). Since problem (PR) is convex, it is easier to handle numerically. Algorithm 1 generates a minimizing sequence $(y^k)_{y \in \mathbb{N}}$ in $\operatorname{conv}(\mathcal{Y})$ for the relaxed problem, by a direct application of

Algorithm 1: Frank-Wolfe Algorithm for the relaxed problem

```
Initialization: $y^0 \in \mathrm{conv}(\mathcal{Y})$;
for $k = 0, 1, 2, \dots$ do
    Find a solution $\bar{y}^k$ to the sub-problem
        (3.3)    $\inf_{y \in \mathrm{conv}(\mathcal{Y})} \langle \nabla f(y^k), y \rangle$
    Choose $\omega_k \in [0, 1]$;
    Set $y^{k+1} = (1 - \omega_k) y^k + \omega_k \bar{y}^k$;
end
```

the Frank-Wolfe algorithm [11]. The general idea of Algorithm 2 is to introduce an approximation step at each iteration, to recover points in $\mathcal{Y}$.

The algorithm is known to converge for various choices of the parameter ω_k. In particular, for $\omega_k = 2/(k+2)$, one can show the existence of a constant $C > 0$ such that for any k, $f(y^k) \leq \mathcal{J}^* + \frac{C}{k}$. Besides the guaranty of convergence of the algorithm, its interest lies in the decomposability of the sub-problems to be solved at each iteration. Problem (3.3) is indeed equivalent to

$$\inf_{x \in \mathcal{X}} \Big\langle \nabla f(y^k), \sum_{i=1}^N g_i(x_i) \Big\rangle. \tag{3.4}$$

Obviously, x is a solution if and only if x_i is a solution to (2.2) (with $y = y^k$). Therefore a solution to (3.3) is given by

$$\bar{y}^k = \frac{1}{N} \sum_{i=1}^N g_i\big(\mathbb{S}_i(y^k)\big). \tag{3.5}$$

Note that $\bar{y}^k \in \mathcal{Y}$. However, even if y^k also belonged to $\mathcal{Y}$, there would be no reason to have $y^{k+1} \in \mathcal{Y}$.

The stochastic Frank-Wolfe algorithm (Algorithm 2) introduced in the next section allows us to overcome this difficulty. At the iteration k, a point x^k has been constructed, with aggregate $y^k = \frac{1}{N}\sum_{i=1}^N g_i(x_i^k)$. The same sub-problems are solved, yielding a point $\bar{x}^k = (\mathbb{S}_1(y^k), \dots, \mathbb{S}_N(y^k))$ with aggregate $\bar{y}^k$. Next, the algorithm generates a sample of n_k points independently and identically distributed (i.i.d.) in $\mathcal{X}$, denoted $\hat{x}^{k,j} = (\hat{x}_i^{k,j})_{i=1,\dots,N}$, with $j = 1, \dots, n_k$. The point $\hat{x}_i^{k,j}$ is equal to x_i^k with probability $1 - \omega_k$ and to $\bar{x}_i^{k,j}$ with probability ω_k. In practice, we simulate $N n_k$ i.i.d. random variables $\lambda_i^{k,j} \sim \mathrm{Bern}(\omega_k)$, where $\mathrm{Bern}(\omega_k)$ denotes the Bernoulli distribution of parameter $\omega_k \in [0, 1]$ and we set

$$\hat{x}_i^{k,j} = (1 - \lambda_i^{k,j}) x_i^k + \lambda_i^{k,j} \bar{x}_i^k.$$

Then x^{k+1} is taken as a minimizer of J over the union of the set of points randomly generated and $\{x^k\}$.

4 Stochastic Frank-Wolfe algorithm

We provide in Algorithm 2 an explicit implementation of our stochastic Frank-Wolfe algorithm.

Algorithm 2: Stochastic Frank-Wolfe Algorithm

```
Initialization: $x^0 \in \mathcal{X}$;
for $k = 0, 1, 2, \dots$ do
    Step 1: Resolution of the sub-problems.
    Compute $y^k = \frac{1}{N}\sum_{i=1}^N g_i(x_i^k)$;
    for $i = 1, 2, \dots, N$ do
        Compute $\bar{x}_i^k = \mathbb{S}_i(y^k)$;
    end
    Step 2: Update.
    Choose $n_k \in \mathbb{N}^*$ and $\omega_k \in [0, 1]$;
    for $j = 1, 2, \dots, n_k$ do
        for $i = 1, 2, \dots, N$ do
            Simulate $\lambda_i^{k,j} \sim \mathrm{Bern}(\omega_k)$,
              independently of all previously
              defined random variables;
            Set $\hat{x}_i^{k,j} = (1 - \lambda_i^{k,j}) x_i^k + \lambda_i^{k,j} \bar{x}_i^k$;
        end
        Set $\hat{x}^{k,j} = (\hat{x}_i^{k,j})_{i=1,\dots,N}$;
    end
    Find $x^{k+1} \in \mathrm{argmin}\{J(x) \,|\, x \in X^k\}$, where
      $X^k = \{\hat{x}^{k,j},\, j = 1, 2, \dots, n_k\} \cup \{x^k\}$;
end
```

We have the following result, proved in [7, Theorem 3.7].

THEOREM 4.1. *Let Assumptions A, B, and C hold true. Assume that $\omega_k = \frac{2}{k+2}$, for all $k \in \mathbb{N}$ in Algorithm 2. Then, for all $K = 1, \dots, 2N$,*

$$\mathbb{E}[\gamma_K] \leq \frac{4C_1}{K}, \quad \textit{where } \gamma_K = J(x^K) - \mathcal{J}^*.$$

Moreover, for all $\epsilon > 0$,

$$\mathbb{P}\Big[\gamma_K < \frac{4C_1}{K} + \epsilon\Big] \geq 1 - \exp\left(\frac{-\varepsilon^2 N}{2(v_K + \epsilon m_K/3)}\right),$$

where the constants m_K and v_K are given by

$$v_K = \frac{2C_0^2}{K^2(K+1)^2}\left(\sum_{k=1}^{K-1} \frac{k(k+1)^2}{n_k}\right)$$

$$m_K = \frac{C_0}{K(K+1)}\Big(\max_{k=1,\dots,K-1} \frac{(k+1)(k+2)}{n_k}\Big).$$

Note that the constants v_K and m_K can be made arbitrarily small by choosing sufficiently large values of $(n_k)_{k=0,\ldots,K}$. Thus for arbitrarily small values of $\epsilon > 0$ and $\epsilon' > 0$, one can choose appropriate numbers of random simulations so that $\mathbb{P}\big[\gamma_{2N} < \frac{2C_1}{N} + \epsilon\big] \geq 1 - \epsilon'$.

REMARK 4.1. *Theorem 4.1 focuses on the choice of stepsize $\omega_k = 2/(k+2)$, which we have utilized in the numerical simulations. It is also possible to determine ω_k by line search, see [7, Remark 3.10].*

5 Aggregative optimal control

In this section, we reformulate optimal control problem (1.1) as a problem of the form (P). We address its resolution with the SFW algorithm.

5.1 Abstract formulation Let us consider an agent i and let us describe its state-control feasible set $\mathcal{X}_i$. Recall that a state S_i, a control set U_i, mappings $U_i^t \colon S_i \to 2^{U_i}$, and transition mapping $\pi_i^t \colon S_i \times U_i \to S_i$ are given. We call feasible *state-control trajectory* an element $x_i = (s_i, u_i)$, where $s_i = (s_i^0, \ldots, s_i^T) \in (S_i)^{T+1}$ and $u_i = (u_i^0, \ldots, u_i^T) \in (U_i)^{T+1}$, such that

$$s_i^0 \in S_i^0, \quad u_i^t \in U_i^t(s_i^t), \quad s_i^{\theta+1} = \pi_i^\theta(s_i^\theta, u_i^\theta),$$

for any $t = 0, \ldots, T$ and any $\theta = 0, \ldots, T-1$. We denote by $\mathcal{X}_i$ the set of feasible state-control trajectories. We set $\mathcal{X} = \prod_{i=1}^N \mathcal{X}_i$. The non-emptyness of $\mathcal{X}_i$ is a straightforward consequence of the following assumption.

ASSUMPTION 1. *The set S_i^0 is non-empty. For all $t = 0, \ldots, T$, for all $s_i^t \in S_i$, the set $U_i^t(s_i^t)$ is non-empty.*

With each agent i are associated $(T+1)$ contribution functions h_i^t, $t = 0, \ldots, T$ and $(T+1)$ individual costs ℓ_i^t, $t = 0, \ldots, T$. We set $\mathcal{E}_0 = \ldots \mathcal{E}_{T+1} = \mathbb{R}$ and we define $T+2$ functions $g_{it} \colon \mathcal{X}_i \to \mathcal{E}_t$ by

$$g_{it}(x_i) = \begin{cases} h_i^t(s_i^t, u_i^t) & \text{if } t \leq T \\ \sum_{t'=0}^T \ell_i^{t'}(s_i^{t'}, u_i^{t'}) & \text{if } t = T+1. \end{cases}$$

The social costs $f_0, \ldots, f_T$ are the same as in the original problem (1.1). The social cost $f_{T+1} \colon \mathcal{E}_{T+1} \to \mathbb{R}$ is the identity function. With these definitions, problem (1.1) is equivalent to

$$\inf_{(x_i)_{i=1}^N \in \prod_{i=1}^N \mathcal{X}_i} \; \sum_{t=0}^{T+1} f_t\Big(\frac{1}{N}\sum_{i=1}^N g_{it}(x_i)\Big). \tag{5.6}$$

5.2 Assumptions As before, we denote $g_i(x_i) = (g_{it}(x_i))_{t=0,\ldots,T+1}$, $\mathcal{E} = \prod_{t=0}^{T+1} \mathcal{E}_t = \mathbb{R}^{T+2}$ and for $y \in \mathcal{E}$, $f(y) = \sum_{t=0}^{T+1} f_t(y_t)$. For any $i = 1, \ldots, N$ and for any $t = 0, \ldots, T+2$, we denote

$$Y_{it} = \big\{ g_{it}(x_i) \mid x_i \in \mathcal{X}_i \big\} \quad \text{and} \quad Y_t = \frac{1}{N}\sum_{i=1}^N Y_{it}.$$

ASSUMPTION 2. *For $i = 1, 2, \ldots, N$ and for $t = 0, 1, \ldots, T$,*

- *f_t is L_t-Lipschitz on* $\mathrm{conv}(Y_t)$*,*
- *f_t is continuously differentiable on a neighborhood of* $\mathrm{conv}(Y_t)$*, ∇f_t is $\tilde{L}_t$-Lipschitz on* $\mathrm{conv}(Y_t)$
- *f_t is convex on* $\mathrm{conv}(Y_t)$*.*

Assumptions 1 and 2 imply Assumptions A and B for problem (5.6). Assumption C is trivially satisfied since $\mathcal{X}_i$ is a finite set.

5.3 Resolution of the sub-problems We explain now how to solve the sub-problems (2.2) associated with the aggregative optimal control problem (5.6). Let $y \in \mathcal{E}$. Let $\mu \in \mathcal{E}$ be defined by $\mu^t = \nabla f_t(y^t)$. By definition of f_{T+1}, $\mu^{T+1} = 1$. The sub-problem (2.2) reads:

$$\inf_{x_i \in \mathcal{X}_i} \sum_{t=0}^T \Big(\ell_i^t(s_i^t, u_i^t) + \langle \mu^t, h_i^t(s_i^t, u_i^t) \rangle \Big). \tag{5.7}$$

The sub-problem (5.7) can be solved by dynamic programming. Algorithm 3 yields a solution to (5.7). For convenience, we denote

$$\ell_i^t[\mu^t](s_i^t, u_i^t) = \ell_i^t(s_i^t, u_i^t) + \langle \mu^t, h_i^t(s_i^t, u_i^t) \rangle$$

in the algorithm. The algorithm consists of two steps: first in a backward pass, a sequence of value functions $(V_i^t)_{t=0,\ldots,T+1}$ is computed, where $V_i^t \colon S_i \to \mathbb{R}$. A globally optimal solution is obtained in a forward pass. Note that the value of the optimization problem of Step 1 is finite as a consequence of Assumption 1.

REMARK 5.1. *As mentioned in the introduction, problem (5.6) could be addressed by dynamic programming. This would allow the computation of an exact solution. However, this would require to compute a value function of the form $V^t(s^t)$, where $s^t = (s_1^t, \ldots, s_N^t) \in \prod_{i=1}^N S_i$. The resulting complexity, of order $T\prod_{i=1}^N |S_i|$, is prohibitive even for moderate values of N. In contrast, the complexity of each iteration of the stochastic Frank-Wolfe algorithm is linear with respect to N, while the accuracy of the algorithm improves as N increases.*

Algorithm 3: Dynamic programming algorithm

```
Step 1: Backward pass.
Set $V_i^{T+1}(s_i^{T+1}) = 0$, for any $s_i^{T+1} \in S_i$.
for $t = T, T-1, \dots, 0$ do
    for $s_i^t \in S_i$ do
        Define $V_i^t(s_i^t)$ as
            $\min_{u_i^t \in U_i^t(s_i^t)} \ell_i^t[\mu^t](s_i^t, \cdot) + V_i^{t+1}\big(\pi_i^t(s_i^t, \cdot)\big).$
    end
end
Step 2: Forward pass.
Find $\bar{s}_i^0 \in \operatorname{argmin}_{s_i^0 \in S_i^0} V_i^0(s_i^0)$;
for $t = 0, \dots, T$ do
    Find a solution $\bar{u}_i^t$ to the problem
        $\min_{U_i^t(\bar{s}_i^t, u_i^t)} \ell_i^t[\mu^t](\bar{s}_i^t, \cdot) + V_i^{t+1}\big(\pi_i^t(\bar{s}_i^t, \cdot)\big).$
    If $t < T$, set $\bar{s}_i^{t+1} = \pi_i^t(\bar{s}_i^t, \bar{u}_i^t)$.
end
```

6 MICP formulation

We give another equivalent problem of (1.1) in this section as a mixed integer convex program (MICP). For any $t \in \{0, \dots, T\}$, we denote $Z_i^t = \{(s_i^t, u_i^t) \in S_i \times U_i \mid u_i^t \in U_i^t(s)\}$. The optimization variable are denoted $m = (m_i^t(s_i^t, u_i^t))$ with indices $i \in \{1, \dots, N\}$, $t \in \{0, \dots, T\}$, $(s_i^t, u_i^t) \in Z_i^t$. The criterion is defined as:

$$\bar{J}(m) := \sum_{t=0}^{T} f_t\Big(\frac{1}{N}\sum_{i=1}^{N}\sum_{z_i^t \in Z_i^t} h_i^t(z_i^t) m_i^t(z_i^t)\Big) + \frac{1}{N}\sum_{t=0}^{T}\sum_{i=1}^{N}\sum_{z_i^t \in Z_i^t} \ell_i^t(z_i^t) m_i^t(z_i^t),$$

where for simplicity we have written z_i^t instead of (s_i^t, u_i^t). We call now (MICP) the problem which consists in minimizing $\bar{J}(m)$ over the variables m satisfying the following constraints:

(i) $m_i^t(s_i^t, u_i^t) \in \mathbb{Z}$,

(ii) $m_i^t(s_i^t, u_i^t) \geq 0, \quad \sum_{z_i^t \in Z_i^t} m_i^t(z_i^t) = 1$,

(iii) $m_i^0(\hat{s}_i^0, u_i^0) = 0$,

(iv) $\sum_{u_i^\theta \in U_i^\theta(s_i^\theta)} m_i^\theta(s_i^\theta, u_i^\theta) = \sum_{z_i^{\theta-1} \in (\pi_i^\theta)^{-1}(s_i^\theta)} m_i^{\theta-1}(z_i^{\theta-1})$,

for any $i = 1, \dots, N$, $t = 0, \dots, T$, $(s_i^t, u_i^t) \in Z_i^t$, $\hat{s}_i^0 \in S_i \backslash S_i^0$, $\theta = 1, \dots, T$, and $s_i^\theta \in S_i$. The terminology MICP comes from the fact that if the integrity constraint in the problem, constraint (i), then the problem becomes a nonlinear convex program: indeed, $\bar{J}$ is a convex function and constraints (ii)-(iii)-(iv) are affine. Let us emphasize the fact that in problem (1.1), the states s_i^t and the controls u_i^t are directly optimized, while here, we optimize a variable m indexed by all possible states and controls. The following lemma shows the equivalence between (1.1) and (MICP).

Lemma 6.1. *Problems* (1.1) *and (MICP) have the same value; moreover, from any solution to* (1.1)*, a solution to (MICP) can be deduced and vice versa.*

Proof. Let $(\bar{s}, \bar{u})$ be a solution of (1.1). Take

$$m_t^i(s_i^t, u_i^t) = \begin{cases} 1, & \text{if } (s_i^t, u_i^t) = (\bar{s}_i^t, \bar{u}_i^t), \\ 0, & \text{otherwise.} \end{cases}$$

We can verify that m is feasible for (MICP) and that $J(\bar{s}, \bar{u}) = \bar{J}(m)$.

Let $\bar{m}$ be a solution of (MICP). From the constraints (i-ii), we deduce that for any (t, i), there exists a unique $(s_i^t, u_i^t) \in Z_t^i$, such that $\bar{m}(t, i, s_i^t, u_i^t) = 1$. Constraint (iii) implies that $s_i^0 \in S_i^0$. Finally constraint (iv) implies that $s_i^{t+1} = \pi_i^t(s_i^t, u_i^t)$. We also have $J(s, u) = \bar{J}(\bar{m})$. The conclusion follows. □

7 Application example

Let us now turn to the problem of the charging of a fleet of batteries. We propose a very simple model which is essentially illustrative, rather than realistic. However, it is emphasised that the proposed approach can easily incorporate more realistic constraints on battery operation (e.g. taking into account limits on cycles numbers). Indeed, these refinements remain localized at the sub-problem level (impacting only the dynamic programming Algorithm 3). They consist either in adding a state variable or in modifying the local costs in order to penalise undesired behaviour. Suppose that there are N batteries to be charged. Let s_i^t be the state of charge (SoC) for the battery i at the time t.

7.1 Dynamics of the batteries The dynamics of each battery is characterized by three parameters: an initial state of charge $s_i^{\text{in}} \in \mathbb{N}$, a maximal state of charge $s_i^{\max} \in \mathbb{N}$, a maximal load speed $u_i^{\max} \in \mathbb{N}$. We define:

$$S_i = \{s_i^{\text{in}}, \dots, s_i^{\max}\},\ S_i^0 = \{s_i^{\text{in}}\},\ U_i = \{0, \dots, u_i^{\max}\},$$
$$U_i^t(s_i^t) = \{0, \dots, \min(u_i^{\max}, s_i^{\max} - s_i^t)\},$$
$$\pi_i^t(s_i^t, u_i^t) = s_i^t + u_i^t.$$

In words: the initial condition s_i^{in} is given, the charging of the battery is additive, the charging speed is bounded by $u_i^{\max}$ and is such that s_i^t can never exceed $s_i^{\max}$.

7.2 Cost and contribution functions Some positive coefficients $(\beta_i)_{i=1,\dots,N}$, $(\alpha_t)_{t=0,\dots,T-1}$, and $(c_t)_{t=0,\dots,T-1}$ are given. The individual costs are

$$\begin{aligned} \ell_i^t(s_i^t, u_i^t) &= 0, \quad \forall t = 0,\dots,T-1, \\ \ell_i^T(s_i^T, u_i^T) &= \beta_i (s_i^{\max} - s_i^T)^2. \end{aligned}$$

The contributions are defined by $h_i^T(s_i^T, u_i^T) = 0$ and

$$h_i^t(s_i^t, u_i^t) = u_i^t, \quad \forall t = 0,\dots,T-1.$$

The social costs f_t are defined by $f_T(y_T) = 0$ and

$$f^t(y_t) = \alpha^t (y_t - c_t)^2, \quad \forall t = 0,\dots,T-1.$$

Therefore, the cost function J writes

$$\sum_{t=0}^{T-1} \alpha^t \left(\left(\frac{1}{N} \sum_{i=1}^{N} u_i^t \right) - c^t \right)^2 + \frac{1}{N} \sum_{i=1}^{N} \beta_i \left(s_i^T - s_i^{\max} \right)^2 .$$

The cost function has two contributions, one depends on the average of charging levels of all the batteries, the other one depends on the individual final SoC of each battery. To be more precise, for $t \le T-1$, the average charging level needs to approach some target power c_t. For $t = T$, the batteries expect to approach their maximum SoCs.

7.3 Numerical simulations The parameters are chosen as follows:

- $N = 100$, $T = 24$
- s_i^{in} (resp. $s_i^{\max}$) is chosen randomly and uniformly in $\{0, 1, \dots, 20\}$ (resp. $\{20, 21, \dots, 40\}$), $u_i^{\max} = 4$
- α^t is chosen randomly and uniformly in $[1, 2]$, β_i is chosen randomly and uniformly in $[0, 1]$
- $c^t = 1.5\lfloor \sin(\pi t/12) + 1 \rfloor$.

Thus, for $t = 0, 1, \dots, 23$, the diameter of the range set Y_{it} is less than $u_i^{\max} = 4$, and the Lipschitz constant $\tilde{L}_t$ is $2\alpha^t$, which is less than 4. Then, we have the following upper bound for the relaxation gap $C_1/2N$:

$$\frac{C_1}{2N} \le \frac{1}{200} \cdot \frac{1}{100} \cdot \sum_{t=0}^{23} \left(4 \cdot \sum_{i=1}^{100} 4^2 \right) = 7,68.$$

Let us estimate the number of variables in the MICP corresponding to this example, that we denote by $d(m)$. First, $|Z_i^t| \ge (s_i^{\max} - s_i^{\text{in}} - u_i^{\max})(u_i^{\max} + 1)$. Then, in view of the distributions of $s_i^{\max}$ and s_i^{in},

$$\mathbb{E}(d(m)) \ge NT\mathbb{E}(|Z_i^t|) \ge 100 \cdot 24 \cdot 16 \cdot 5 = 192\,000,$$

making the MICP appraoch intractable.

Fig. 1 shows the outcome of Algorithm 1 with 500 iterations to get an approximation of the minimum $\mathcal{J}^*$ of the relaxed problem. The curve represents the relaxed cost. Fig. 2 shows the outcome of Algorithm 2, for different choices of n_k with 100 iterations. Since the algorithm is stochastic, we ran it 50 times independently to evaluate its efficiency; the curves represent the average value of $\gamma_k = J(x^k) - \mathcal{J}^*$. The standard deviation is displayed on Fig. 3. In all cases, an average value of the gap significantly smaller than $7,68$ can be reached; the standard deviation is also significantly smaller than $7,68$ at the last iterations. We have initialized the algorithm with values of x_i^0 such that $u_i^t = 0$, for any $t = 0, \dots, T-1$.

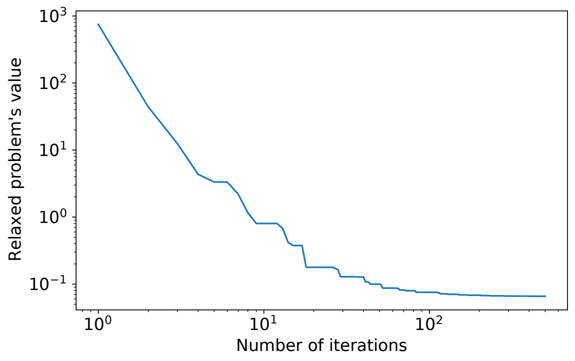

Figure 1: Frank-Wolfe Algorithm with 500 iterations for the relaxed problem.

References

[1] K. Barty, P. Carpentier, and P. Girardeau. Decomposition of large-scale stochastic optimal control problems. *RAIRO-Operations Research*, 44(3):167–183, 2010.

[2] O. Beaude, P. Benchimol, S. Gaubert, P. Jacquot, and N. Oudjane. A privacy-preserving method to optimize distributed resource allocation. *SIAM Journal on Optimization*, 30(3):2303–2336, 2020.

[3] D. Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 1. Athena scientific, 2012.

[4] D.P. Bertsekas. *Nonlinear programming.* Belmont, MA: Athena Scientific, 2nd ed. edition, 1999.

[5] D.P. Bertsekas. *Constrained optimization and Lagrange multiplier methods.* Academic press, 2014.

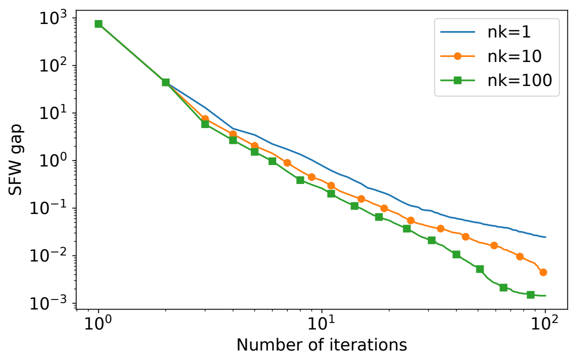

Figure 2: Stochastic Frank-Wolfe Algorithm with 100 iterations, expectation of the gap.

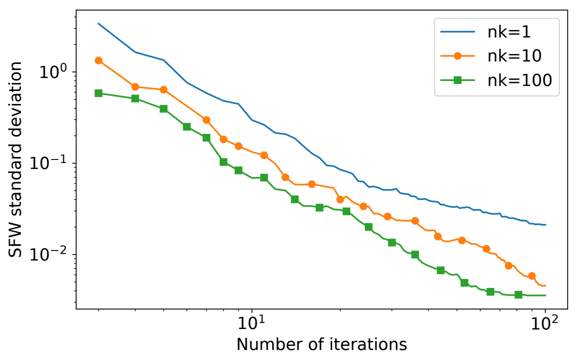

Figure 3: Stochastic Frank-Wolfe Algorithm with 100 iterations, standard deviation of the gap.

[6] K. Bestuzheva, M. Besançon, W.K. Chen, A. Chmiela, T. Donkiewicz, J. van Doornmalen, L. Eifler, O. Gaul, G. Gamrath, A. Gleixner, et al. The scip optimization suite 8.0. *arXiv preprint arXiv:2112.08872*, 2021.

[7] J.F. Bonnans, K. Liu, N. Oudjane, L. Pfeiffer, and C. Wan. Large-scale nonconvex optimization: randomization, gap estimation, and numerical resolution. *ArXiv preprint*, 2022.

[8] P. Carpentier, J.-P. Chancelier, V. Leclère, and F. Pacaud. Stochastic decomposition applied to large-scale hydro valleys management. *European Journal of Operational Research*, 270(3):1086–1098, 2018.

[9] L. Chizat and F. Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[10] C. Coey, M. Lubin, and J.P. Vielma. Outer approximation with conic certificates for mixed-integer convex problems. *Mathematical Programming Computation*, 12(2):249–293, 2020.

[11] J.C. Dunn and S. Harshbarger. Conditional gradient algorithms with open loop step size rules. *Journal of Mathematical Analysis and Applications*, 62(2):432–444, 1978.

[12] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018.

[13] H. Hao, B.M. Sanandaji, K. Poolla, and T.L. Vincent. Aggregate flexibility of thermostatically controlled loads. *IEEE Transactions on Power Systems*, 30(1):189–198, 2014.

[14] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms II: Advanced theory and bundle methods*, volume 306 of *Grundlehren Math. Wiss.* Berlin: Springer-Verlag, 1993.

[15] P. Jacquot, O. Beaude, S. Gaubert, and N. Oudjane. Analysis and implementation of an hourly billing mechanism for demand response management. *IEEE Transactions on Smart Grid*, 10(4):4265–4278, 2018.

[16] S. Mei, T. Misiakiewicz, and A. Montanari. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. In *Conference on Learning Theory*, pages 2388–2464. PMLR, 2019.

[17] S. Mei, A. Montanari, and P.-M. Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[18] A. Seguret, C. Alasseur, J.F. Bonnans, A. De Paola, N. Oudjane, and V. Trovato. Decomposition of high dimensional aggregative stochastic control problems. *ArXiv preprint*, 2021.

[19] R.M. Starr. Quasi-equilibria in markets with non-convex preferences. *Econometrica: journal of the Econometric Society*, pages 25–38, 1969.

[20] M. Wang. Vanishing price of decentralization in large coordinative nonconvex optimization. *SIAM J. Optim.*, 27(3):1977–2009, 2017.