EMSx: an Energy Management System numerical benchmark

Adrien Le Franc, Pierre Carpentier, Jean-Philippe Chancelier, Michel de Lara

Noname manuscript No.
(will be inserted by the editor)

EMSx: an Energy Management System numerical benchmark

Adrien Le Franc · Pierre Carpentier · Jean-Philippe Chancelier · Michel De Lara

December 31, 2019

Abstract Inserting renewable energy in the electric grid in a decentralized manner is a key challenge of the energy transition. However, at local stage, both production and demand display erratic behavior, be it for the dependence upon local weather conditions. Thus, the match between supply and demand is delicate to achieve in such a stochastic context. It is the goal of Energy Management Systems (EMS) to achieve such balance at least cost. We present EMSx, a numerical benchmark for testing control algorithms for the management of electric microgrids doted with a photovoltaic unit and an energy storage system. Our benchmark is based on a rich collection of historical observations and forecasts collected by the company Schneider Electric. Besides the dataset, we also publicly release `EMSx.jl`, a package implemented in the Julia language which enables to easily implement a microgrid controller and to test it on the EMSx benchmark. Eventually, we showcase the results of standard microgrid control methods, including Model Predictive Control, Open Loop Feedback Control and Stochastic Dynamic Programming.

Keywords Electric microgrid · Multistage stochastic optimization · Numerical benchmark

Adrien Le Franc
CERMICS ENPC, Efficacity
E-mail: adrien.le-franc@enpc.fr

Pierre Carpentier
UMA ENSTA Paris
E-mail: pierre.carpentier@ensta-paris.fr

Jean-Philippe Chancelier
CERMICS ENPC
E-mail: jpc@cermics.enpc.fr

Michel De Lara
CERMICS ENPC
E-mail: michel.delara@enpc.fr

1 Introduction

Inserting renewable energy in the electric grid is a key challenge of the energy transition. Compared with standard power units, wind and solar generators are subject to more uncertainties, due to their dependence on local weather conditions. Their integration in the power system is often envisaged at a local scale, where the demand is more erratic than at a global scale. In this work, we focus on photovoltaic units integrated in local power networks, with uncertainties arising both from the electric load and from the photovoltaic power generation. Ideally, the local production would match the consumption at all time. In practice, the stochasticity of both signals lowers the covering of the load by the photovoltaic power. Introducing an energy storage system can help equating supply with demand in such electric microgrids. The resulting complex management problem aims at reducing the energy bill.

The optimal operation of such systems is a well-known challenge. We refer to [8, 9, 13] for examples of battery management for renewable power integration in electric microgrids. Our work is concentrated on Energy Management Systems (EMS) that set target state of charge or average power for the storage system over time spans of a few minutes. An EMS is a high level layer of hierarchical control, responsible for operating the microgrid while optimizing security and economic criteria [11].

In this paper, we are concerned with the numerical assessment of EMS strategies, yielded by control algorithms. Several works have proposed to compare such control algorithms on a problem that they provide [9, 13, 14]. We introduce the EMSx benchmark, an electric microgrid management problem designed for evaluating controllers on a unified challenge. Compared with previous EMS experimental benchmarks, the strength of EMSx relies on its open-data and open-software inclination. Besides, the data that we release offer a diversified and realistic pool of microgrids collected by the Schneider Electric company on 70 industrial sites. While these data can be used for various tasks related to microgrid simulation, we also release the `EMSx.jl` package implemented in the Julia [4] language, that enables simulating a large range of controllers on Schneider Electric's dataset. We also propose a score for measuring the aggregated performance of controllers across different sites. We illustrate the application of the EMSx benchmark on a selection of controllers derived from multistage deterministic and stochastic optimization techniques. Our selection includes standard Model Predictive Control (MPC, [2, 7]), Open Loop Feedback Control (OLFC, [1, Vol. 1, §6.2], sometimes referred to as stochastic MPC), Stochastic Dynamic Programming (SDP [3, 12]), and an extended state formulation of a plain SDP controller that models uncertainties with an auto-regressive process (SDP-AR , [15, §3.1.1] and [10]).

This paper is structured as follows. We introduce our dataset in Sect. 2, then we formalize the EMSx benchmark in Sect. 3 by providing a microgrid simulation model, a score and simulation parameters. In Sect. 4, we introduce some control methods and outline their performance on the EMSx benchmark.

The Appendix A gathers additional material on the numerical experiments of Sect. 4.

2 Dataset

We present the Schneider Electric dataset which offers a large collection of field data, from 70 sites, for studying the management of electric microgrid. The dataset includes battery parameters, and both historical observations (§2.1) and historical forecasts (§2.2). In §2.3, we illustrate how the 70 sites differ in terms of "predictability".

Schneider Electric has collected a large data base of photovoltaic and load profiles on real operated microgrids deployed on a various collection of sites. We provide a *set I of 70 sites*. On each site $i \in I$, we provide battery parameters $(c^i, \overline{l}^i, \rho_c^i, \rho_d^i)_{i \in I}$ (see the storage dynamics part in §3.1 below). We also sample the continuous time every 15 minutes, giving, for each site $i \in I$, a *time index* $t \in \{1, 2, \ldots, \theta^i - 1, \theta^i\}$, over a *horizon* θ^i (with at least one year of historical observations per site). Every interval $[t, t+1]$ corresponds to 15 minutes.

2.1 Historical observations

We dispose of measures of the *photovoltaic generation* g_t and of the *energy demand* d_t over the last 15 minutes, providing vectors of *historical observations* for each site $i \in I$

$$g^i = (g_1^i, \ldots, g_{\theta^i}^i) \in \mathbb{R}^{\theta^i} \ , \ \ \forall i \in I \ , \tag{1a}$$

$$d^i = (d_1^i, \ldots, d_{\theta^i}^i) \in \mathbb{R}^{\theta^i} \ , \ \ \forall i \in I \ . \tag{1b}$$

Examples of observed daily chronicles are given for photovoltaic generation (Figure 1) and energy demand (Figure 2). Our data set is publicly available at the url `https://adrien-le-franc.github.io/home/data.html`

2.2 Historical forecasts

On top of observed data, Schneider Electric also provides, every 15 minutes, *historical forecasts* $\hat{g}_{t,t+1}^i, \ldots, \hat{g}_{t,t+96}^i$ *of photovoltaic profiles* and *historical forecasts* $\hat{d}_{t,t+1}^i, \ldots, \hat{d}_{t,t+96}^i$ *of demand profiles* for the next 24 hours, hence giving vectors for all sites $i \in I$ and for all times $t \in \{1, \ldots, \theta^i\}$

$$\hat{g}_t^i = (\hat{g}_{t,t+1}^i, \ldots, \hat{g}_{t,t+96}^i) \in \mathbb{R}^{96} \ , \ \ \forall t \in \{1, \ldots, \theta^i\} \ , \ \ \forall i \in I \ , \tag{2a}$$

$$\hat{d}_t^i = (\hat{d}_{t,t+1}^i, \ldots, \hat{d}_{t,t+96}^i) \in \mathbb{R}^{96} \ , \ \ \forall t \in \{1, \ldots, \theta^i\} \ , \ \ \forall i \in I \ , \tag{2b}$$

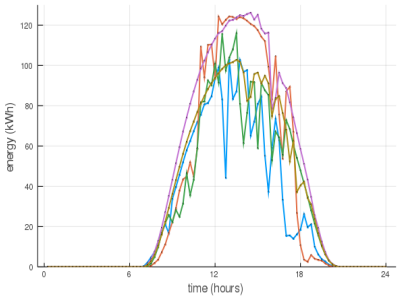

Fig. 1: Examples of daily photovoltaic profiles

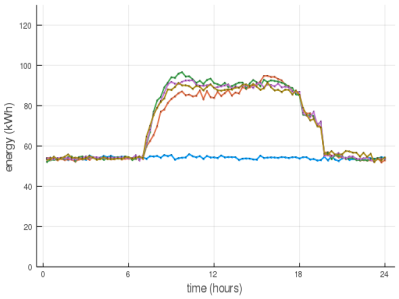

Fig. 2: Examples of daily load profiles

hence sequences of *historical forecasts* for all sites $i \in I$

$$\hat{g}^i = (\hat{g}^i_1, \dots, \hat{g}^i_{\theta^i}) \in \mathbb{R}^{96\times\theta^i} \ , \ \ \forall i \in I \ , \tag{3a}$$

$$\hat{d}^i = (\hat{d}^i_1, \dots, \hat{d}^i_{\theta^i}) \in \mathbb{R}^{96\times\theta^i} \ , \ \ \forall i \in I \ . \tag{3b}$$

By combining the stream (1) of *historical observations* with the stream (3) of *historical forecasts*, we can thus closely reproduce the information available to an online microgrid controller operating a real site.

2.3 Illustrating how sites differ in terms of predictability

The dataset covers a large spectrum of situations regarding variability and predictability. To assess the predictability of the data at a given site $i \in I$, we first introduce the *historical net demand observations*

$$z^i_t = d^i_t - g^i_t \in \mathbb{R} \ , \ \ \forall t \in \{1, \dots, \theta^i\} \ , \ \ \forall i \in I \ , \tag{4a}$$

deduced from (1), and the *historical net demand forecasts*

$$\hat{z}^i_t = \hat{d}^i_t - \hat{g}^i_t \in \mathbb{R}^{96} \ , \ \ \forall t \in \{1, \dots, \theta^i\} \ , \ \ \forall i \in I \ , \tag{4b}$$

deduced from (3). Second, we normalize the historical net demand observations by

$$\widetilde{z}^i_t = \frac{z^i_t - \underline{z}^i}{\overline{z}^i - \underline{z}^i} \in [0,1] \ , \ \ \forall t \in \{1, \dots, \theta^i\} \ , \ \ \forall i \in I \ , \tag{5a}$$

$$\text{where} \ \begin{cases} \overline{z}^i = \max_{1\leq t\leq\theta^i}\{z^i_t\} \ , \ \ \forall i \in I \ , \\ \underline{z}^i = \min_{1\leq t\leq\theta^i}\{z^i_t\} \ , \ \ \forall i \in I \ , \end{cases} \tag{5b}$$

and we apply the same transformation to the components $\hat{z}^i_{t,t+1}, \dots, \hat{z}^i_{t,t+96}$ of the historical net demand forecasts, yielding

$$\widetilde{\hat{z}}^i_{t,t+k} = \frac{\hat{z}^i_{t,t+k} - \underline{z}^i}{\overline{z}^i - \underline{z}^i} \in \mathbb{R} \ , \ \ \forall k \in \{1, \dots, 96\} \ , \ \ \forall t \in \{1, \dots, \theta^i\} \ , \ \ \forall i \in I \ . \tag{6}$$

Thus normalized, predictability can be compared across the pool of sites. Third, we define the Root Mean Square Error (RMSE) of the site $i \in I$ by

$$\mathrm{RMSE}^{\,i} = \sqrt{\frac{1}{96 \times \theta^i} \sum_{t=1}^{\theta^i} \sum_{k=1}^{96} \left(\tilde{\hat{z}}^{\,i}_{t,t+k} - \tilde{z}^{\,i}_{t+k}\right)^2} \,. \tag{7}$$

The diversity of the forecast error over the pool of 70 sites is exposed in Figure 3. Here, we ranked sites in increasing order of RMSE. We observe that the error is quite stable in the wide flat central part of sites distribution, but for the first 20% (with low forecast error) and for the last 20% (with high forecast error) of the sites. Thus, our dataset offers a diversified pool of microgrids which we expect to be a profitable playground for research in the field of photovoltaic migrogrid management.

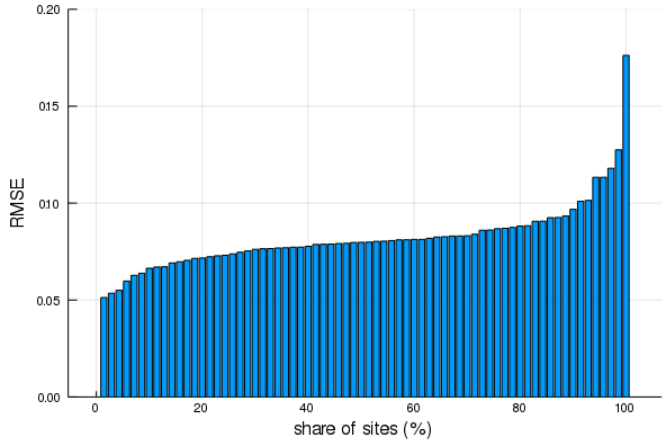

Fig. 3: RMSE of the net demand forecast over the pool of 70 sites ranked in increasing order of forecast error

3 The EMSx controller benchmark

Disposing of Schneider Electric's data, we formulate a benchmark problem for evaluating generic microgrid controllers (§3.1). Then, we describe the structure of a controller, and how it is assessed by simulation along a partial chronicle, in §3.2. In §3.3, we detail a score to assess controllers, and we extend it into a score to assess a controller design technique in §3.4. Finally, we present in §3.5 the microgrid simulation package `EMSx.jl`, implemented in the Julia language and destined to ease the simulation loop.

3.1 Microgrid control simulation

We consider an electric microgrid composed of a photovoltaic power unit, an electric load and an energy storage system. We assume that all components of the microgrid share a single point of connection with the global grid. At this point, electric power can be imported or exported so as to satisfy the electric power demand (total load) at all time. We provide a schematic model of such a system (Figure 4). We outline the mathematical control of a microgrid over a finite number of discrete time steps $t \in \{0, 1, 2, \dots, T-1, T\}$, where unit steps are spaced by $\Delta_t = 15$ minutes.

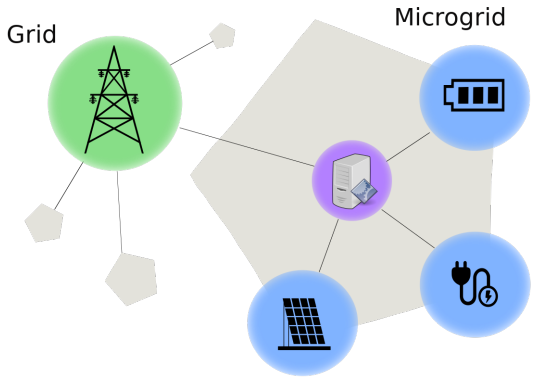

Fig. 4: Schematic microgrid model

Storage dynamics. The storage system is assumed to be a lithium-ion battery (or a container of aggregated batteries), characterized by the coefficients $(c, \bar{l}, \rho_c, \rho_d)$ referring respectively to the battery's capacity (kWh), maximum load (kW), charge and discharge efficiency coefficients. The dynamics of its *state of charge* x_t is given by

$$x_{t+1} = f(x_t, u_t) \ , \ \ \forall t \in \{0, \dots, T-1\} \ , \tag{8a}$$

where the dynamics f is given by

$$f(x, u) = x + \frac{\rho_c}{c} u^+ - \frac{1}{\rho_d c} u^- \ , \ \ \forall (x, u) \in [0, 1] \times \mathbb{R} \ , \tag{8b}$$

with $u^+ = \max(0, u)$ and $u^- = \max(0, -u)$.

Constraints. The *decision* u_t, taken at the beginning of every time interval $[t, t+1[$, accounts for *the energy charged* ($u_t \geq 0$) or *discharged* ($u_t \leq 0$) during $[t, t+1[$. Combined with the dynamics (8), constraints of the form

$$u_t \in \mathcal{U}(x_t) \ , \ \ \forall t \in \{0, \dots, T-1\} \tag{9a}$$

restrict decisions u_t to the admissibility set (related to some of the battery parameters $(c, \bar{l}, \rho_c, \rho_d)$)

$$\mathcal{U}(x) = \{u \in \mathbb{R} \mid \underline{u} \leq u \leq \overline{u} \text{ and } 0 \leq f(x,u) \leq 1\} \ , \tag{9b}$$

where $\overline{u} = \bar{l} \times \Delta_t$ and $\underline{u} = -\bar{l} \times \Delta_t$ are the bounds on the energy that can be exchanged with the battery during a 15 minutes interval.

Uncertainties and scenarios. For the purpose of defining the management costs, we introduce the *uncertainties*

$$w_t = (g_t, d_t) \in \mathbb{R}^2 \ , \ \ \forall t \in \{1, \dots, T\} \tag{10a}$$

— which represent a couple of photovoltaic generation g_t and of energy demand d_t — and we call *scenario* a sequence

$$(w_1, \dots, w_T) \in \mathbb{R}^{2 \times T} \tag{10b}$$

of uncertainties. As defined, both uncertainties and scenarios are generic variables, and not necessarily extracted from the historical observations of §2.1.

Costs. We now turn to the management costs. The *stage cost* during the time interval $[t, t+1]$ is

$$L_t(u_t, w_{t+1}) = p_t^+ \cdot e_{t+1}^+ - p_t^- \cdot e_{t+1}^- \ , \ \ \forall t \in \{0, \dots, T-1\} \ , \tag{11a}$$

where

$$e_{t+1} = d_{t+1} - g_{t+1} + u_t \ , \ \ \forall t \in \{0, \dots, T-1\} \ , \tag{11b}$$

is the *energy exchanged with the grid* — which, like the uncertainty (g_{t+1}, d_{t+1}), materializes at the end of the time interval $[t, t+1[$, hence the index $t+1$ — and where (p_t^+, p_t^-) is the energy tariff (buying at price p_t^+ and selling at price p_t^-) applied during the time interval $[t, t+1]$.

Given a scenario $(w_1, \dots, w_T)$ and a sequence $(u_0, \dots, u_{T-1})$ of controls, we obtain the *total operating cost*

$$\mathcal{L}(u_0, \dots, u_{T-1}, w_1, \dots, w_T) = \sum_{t=0}^{T-1} L_t(u_t, w_{t+1}) \ . \tag{11c}$$

Eventually, we have introduced a dynamical system with dynamics (8), constraints (9) and a cost structure (11).

3.2 Microgrid controller

A microgrid controller is a mathematical device that, given some information at time t, yields a decision u_t. We now detail the structure of the controllers that we will consider. For this purpose, we introduce chronicles and management cost.

Partial chronicles. At the beginning of the time interval $[t, t+1]$, we dispose of all the past observations and past forecasts to make a decision u_t. For practical computational reasons, we have chosen to restrict this information to the *partial observations*

$$(w_t, w_{t-1}, \dots, w_{t-95}) \in \mathbb{R}^{2\times 96} \ , \ \ \forall t \in \{0, \dots, T-1\} \tag{12a}$$

of uncertainties (10a) over the last 24 hours (hence $96 = 24 \times 60/15$), and to the *partial forecasts*

$$(\hat{w}_{t,t+1}, \dots, \hat{w}_{t,t+96}) = \begin{pmatrix} \hat{g}_{t,t+1}, \dots, \hat{g}_{t,t+96} \\ \hat{d}_{t,t+1}, \dots, \hat{d}_{t,t+96} \end{pmatrix} \in \mathbb{R}^{2\times 96} \ , \ \ \forall t \in \{0, \dots, T-1\} \ , \tag{12b}$$

which represent a prediction of the uncertainties (10a) for the next 24 hours. Combined together, we obtain the *partial observations-forecasts*

$$h_t = \begin{pmatrix} w_t, w_{t-1}, \dots, w_{t-95} \\ \hat{w}_{t,t+1}, \dots, \hat{w}_{t,t+96} \end{pmatrix} \in \mathbb{H} \ , \ \ \forall t \in \{0, \dots, T-1\} \ , \tag{12c}$$

where

$$\mathbb{H} = \mathbb{R}^{2\times 96} \times \mathbb{R}^{2\times 96} \ , \tag{12d}$$

and, stacking them all over the whole time span, we obtain the *partial chronicle*

$$h = (h_0, \dots, h_{T-1}) \in \mathbb{H}^T \ . \tag{12e}$$

Controller. A *controller* ϕ is a sequence of mappings

$$\phi = (\phi_0, \dots, \phi_{T-1}) \tag{13a}$$

where, for all $t \in \{0, \dots, T-1\}$, we have

$$\phi_t : [0,1] \times \mathbb{H} \to \mathbb{R} \ , \tag{13b}$$

$$\phi_t(x_t, h_t) \in \mathcal{U}(x_t) \ , \ \ \forall (x_t, h_t) \in [0,1] \times \mathbb{H} \ , \ \ \forall t \in \{0, \dots, T-1\} \ , \tag{13c}$$

where the constraint set $\mathcal{U}(x_t)$ is defined in (9b).

Management cost of a controller along a partial chronicle on a given site. All dynamics (8), constraints (9) and cost structure (11) depend on parameters relative to a site. Therefore, on a given site $i \in I$, we denote by f^i the dynamics of the battery in (8) and by $\mathcal{U}^i$ the constraint set in (9b), as they depend on the local parameters $(c^i, \bar{l}^i, \rho_c^i, \rho_d^i)$. We also denote by L_t^i the stage cost in (11a) — as it depends on the energy tariff $(p_t^{+,i}, p_t^{-,i})$ which could possibly be local — and by $\mathcal{L}^i$ the total operating cost in (11c), as it depends on dynamics f^i, constraint set $\mathcal{U}^i$ and stage costs L_t^i.

Besides battery parameters and energy tariffs, sites differ from each others in their historical data. For instance, the RMSE of the historical forecasts

(Figure 3) is diversified across the pool of sites. Therefore, controllers in (13) may differ accordingly. This is why we denote by ϕ^i a controller for the site $i \in I$.

The application of a controller ϕ^i in (13) along a partial chronicle $h \in \mathbb{H}^T$ in (12e) yields the *management cost*

$$J^i(\phi^i, h) = \sum_{t=0}^{T-1} L_t^i(u_t, w_{t+1}) \ , \tag{14a}$$

where, for all $t \in \{0, \dots, T-1\}$, the sequence $(u_0, \dots, u_{T-1})$ of controls is given by

$$x_0 = 0 \ , \tag{14b}$$

$$x_{t+1} = f^i(x_t, u_t) \ , \tag{14c}$$

$$u_t = \phi_t^i(x_t, h_t) \ . \tag{14d}$$

The management cost $J^i(\phi^i, h)$ will serve the assessment of the controller ϕ^i in §3.3.

3.3 Designing and assessing a controller on a given site

We consider a given site $i \in I$. We outline how to design and we detail how to assess a controller ϕ^i as in (13).

Data partitioning. We have split the data base in periods of one week ranging from Monday 00:00 to Sunday 23:45, each week containing thus $T = 672 = 7 \times 24 \times 4$ time steps. With this, the data base is now organized in chronicles $\mathcal{D}^i \subset \mathbb{H}^T$, elements of the set $\mathbb{H}^T$, where $\mathbb{H}$ has been defined in (12).

Then, we partition the chronicles in the data set $\mathcal{D}^i$ in two disjoint subsets, $\mathcal{C}^i$ for calibration (training, in-sample) and $\mathcal{S}^i$ for simulation (testing, out-of-sample):

$$\mathcal{C}^i \cup \mathcal{S}^i = \mathcal{D}^i \subset \mathbb{H}^T \ , \ \ \mathcal{C}^i \cap \mathcal{S}^i = \emptyset \ . \tag{15}$$

For the EMSx benchmark, we select randomly 40% of the weeks for simulation and let the other 60% be available for calibration.

Calibration data. The calibration data in $\mathcal{C}^i$ is available for the design of microgrid controllers as in (13). The design can resort from any sort of technique (see examples in §4.1 and in §4.2 below).

Simulation data. On top of the weekly periods, every simulation chronicle in $\mathcal{S}^i$ is augmented with the data of the Sunday before the period starts, following our definition (12e), so that, when simulating a microgrid controller, 24 hours of past history data is always available to the decision-maker. Simulation chronicles serve for testing only; as such, they cannot be employed for the design of a controller.

Parameters. Additionally, we dispose of the battery parameters $c^i, \bar{l}^i, \rho_c^i, \rho_d^i$, whose dimension we designed with Schneider Electric. For computing the management cost in (14), we use the energy tariff and time of use (p_t^+, p_t^-), in €/kWh, from the French electricity provider Électricité de France (EDF); it is the same for all sites. At the beginning of any simulation, we assume the battery to be empty (i.e. $x_0 = 0$), and we do not impose any final cost.

Lower bound for the management cost. For any controller ϕ^i as in (13) and any partial chronicle $h \in \mathbb{H}^T$ in (12e), the management cost $J^i(\phi^i, h)$ in (14) always has the so-called "anticipative" lower bound $\underline{J}^i(h)$, computed as the minimum of (14a) under the same constraints, initial state (14b) and dynamics (14c), but where the last constraint $u_t = \phi_t^i(x_t, h_t)$ in (14d) is now enlarged as $u_t = \psi_t^i(x_t, h)$, for all $t \in \{0, \dots, T-1\}$, for any $\psi_t^i : [0,1] \times \mathbb{H}^T$. This gives a lower bound, because the minimization is done over such "anticipative" control laws $\psi_t^i : [0,1] \times \mathbb{H}^T$, which encompass any controller ϕ^i as in (13). Therefore, we easily get that, for any controller ϕ^i as in (13),

$$\underline{J}^i(h) \leq J^i(\phi^i, h) \ , \ \ \forall h \in \mathcal{S}^i \tag{16}$$

Performance score. With the battery parameters and energy tariff, and a given site controller ϕ^i as in (13), the simulator in (14) yields as many management costs $J^i(\phi^i, h)$ as there are chronicles h in the simulation chronicles $\mathcal{S}^i$. Because the volume of energy production and consumption is variable from one site to another, raw management costs $J^i(\phi^i, h)$ are not suitable for a global performance analysis if we want to assess not only a given controller, but a controller design technique (see examples in §4.1 and in §4.2, and see §3.4 below).

Therefore, we normalize the average management cost of controller ϕ^i over chronicles $h \in \mathcal{S}^i$. For this normalization, the lower bound is furnished by taking the average of the lower bound $\underline{J}^i(h)$ in (16). For the upper bound, we chose the average management cost of a dummy management policy that does not use the battery, i.e. a dummy controller ϕ^{d} such that $\phi_t^{\mathrm{d}} = 0$, for $t \in \{0, \dots, T-1\}$. The *performance score* (of a single controller ϕ^i) is

$$G^i(\phi^i) = \frac{\frac{1}{|\mathcal{S}^i|}\sum_{h \in \mathcal{S}^i} J^i(\phi^{\mathrm{d}}, h) - J^i(\phi^i, h)}{\frac{1}{|\mathcal{S}^i|}\sum_{h \in \mathcal{S}^i} J^i(\phi^{\mathrm{d}}, h) - \underline{J}^i(h)} = \frac{\sum_{h \in \mathcal{S}^i} J^i(\phi^{\mathrm{d}}, h) - J^i(\phi^i, h)}{\sum_{h \in \mathcal{S}^i} J^i(\phi^{\mathrm{d}}, h) - \underline{J}^i(h)} \ . \tag{17}$$

This score $G^i(\phi^i)$ can be interpreted as the gain of introducing an energy storage system controlled by ϕ^i for the site i and the time span $\{0, 1, \dots, T\}$ (one week) considered, relatively to the maximum expected cost saving of $\frac{1}{|\mathcal{S}^i|}\sum_{h \in \mathcal{S}^i} J^i(\phi^{\mathrm{d}}, h) - \underline{J}^i(h)$. It permits an immediate interpretation for practitioners interested in deploying controllers on real microgrids. The higher the score, the higher the gain allowed by the controller ϕ^i. Beware that, with this normalization, if a controller ϕ^i performs worse than the dummy controller ϕ^{d}, it will receive a negative score.

3.4 Assessing a controller design technique

Controller design technique. In addition to assessing a given controller, we also aim at assessing a *design technique for controllers*. We provide examples of design techniques in §4.1 and in §4.2. In particular, when the same design technique is used across all sites, we will consider a collection $\{\phi^i\}_{i\in I}$ of controllers derived from the application of a single design technique, but adapted to each site $i \in I$.

Performance score of a collection of controllers. For a given collection $\{\phi^i\}_{i\in I}$ of controllers, one per site, we average the performance score (17) over sites, yielding the *performance score* (of a collection $\{\phi^i\}_{i\in I}$ of controllers)

$$G\big(\{\phi^i\}_{i\in I}\big) = \frac{1}{|I|}\sum_{i\in I} G^i(\phi^i)\ . \tag{18}$$

When the controllers ϕ^i in the collection $\{\phi^i\}_{i\in I}$ have been designed by the same technique, the score $G\big(\{\phi^i\}_{i\in I}\big)$ in (18) is a proxy to measure the performance of a controller design technique over a large range of situations, both in time of the year and in type of microgrid. It permits an immediate interpretation for practitioners interested in deploying a technique to design ontrollers on real microgrids. The higher the score, the higher the gain allowed by the technique.

3.5 The `EMSx.jl` package

In order to ease the simulation loop for computing the management cost (14) — which has to be repeated on several testing periods, sampled from all sites, for a various collection of controllers — we developed `EMSx.jl`, a microgrid simulation package implemented in the Julia language. With the help of `EMSx.jl`, it is easy to implement a large range of controllers, as defined in §3.2, and to apply them on Schneider Electric's dataset described in Sect. 2.

Given a site $i \in I$, a controller ϕ^i in (13) and a partial chronicle $h \in \mathbb{H}^T$ in (12e) (in practice, $h \in \mathcal{S}^i$, the simulation chronicles in (15)), the `EMSx.jl` software returns the sequence of states of charge of the battery, controls and stagewise costs, and, above all, the management cost $J^i(\phi^i, h)$ in (14) and the corresponding computing time.

Figure 5 illustrates how a dummy controller $\phi^{\mathrm{d}}_t = 0,\ t \in \{0,\dots,T-1\}$ can be implemented and tested in a few lines of code. First, we define a new type for our controller, named `DummyController`, as a subtype of `AbstractController`, a built-in type of `EMSx.jl` (line 3). Then, we implement the specialized expression (in Julia jargon, *method*) of the `compute_control` function for the new `DummyController` type (line 5). In the given example, the expression is trivial as the *method* always returns zero, but indeed more complex ones could be implemented and could use the `information` object given

in the input arguments of the function. The `information` object contains all the data available to a controller as in (13). Eventually, we create an instance of a `DummyController` (line 8) and launch the simulation over all simulation chronicles (line 10) passing the created instance as argument. Battery parameters $(c, \bar{l}, \rho_c, \rho_d)$ (referred to as metadata, line 13) and energy tariff (p^+, p^-) (line 12) can be changed by the user to run a custom simulation. The code of `EMSx.jl` and more illustrative controller examples are publicly available at `https://github.com/adrien-le-franc/EMSx.jl`.

```
1  using EMSx
2
3  mutable struct DummyController <: EMSx.AbstractController end
4
5  EMSx.compute_control(controller::DummyController,
6   information::EMSx.Information) = 0.
7
8  const controller = DummyController()
9
10 EMSx.simulate_sites(controller,
11  "home/xxx/path_to_save_folder",
12  "home/xxx/path_to_price",
13  "home/xxx/path_to_metadata",
14  "home/xxx/path_to_simulation_data")
```

Fig. 5: Example of the implementation and simulation of a controller with the `EMSx.jl` package

All in all, our benchmark includes a simulation algorithm packaged in the `EMSx.jl` software, a set of simulation chronicles and parameters, and a score for measuring controllers performance. All of these components are made publicly available in order to encourage the testing of various controllers on the EMSx benchmark. Now, we turn to an example of use of the `EMSx.jl` software.

4 Numerical experiments

To illustrate how we can use the EMSx controller benchmark of Sect. 3, we present several controllers and provide their scores. These controllers ϕ all have the same structure: for any step $t \in \{0, \dots, T-1\}$, the quantity $\phi_t(x_t, h_t)$ in (13) is not given by an analytical formula, but as the solution of a (reference) optimization problem. Controller design techniques differ according to the nature of this latter problem, which is solved at every step $t \in \{0, \dots, T-1\}$, that is, *online* ("on the fly") as a function of the current available quantities (x_t, h_t). A comprehensive overview of such techniques and algorithms is given in [1].

In §4.1, we present a class of so-called lookahead methods where the reference optimization problem is multistage open-loop. In contrast, in §4.2, we

present a class of so-called cost-to-go methods where the reference optimization problem, solved online, is one-stage but depends on cost-to-go functions that are computed *offline*. In §4.3, we comment the results obtained when applying the controllers above on the EMSx controller benchmark.

4.1 Controllers obtained by lookahead computation methods

In lookahead methods, one solves, for every step $t \in \{0, \ldots, T-1\}$, a reference multistage ("looking ahead" from the current step t) optimization problem which is open-loop, be it deterministic (MPC) or stochastic (OLFC). Stochastic (scenario-based) lookahead methods are limited by the exponential growth of the computing time with respect to the number of scenarios.

4.1.1 Model Predictive Control

The *Model Predictive Control* (MPC) method is one of the most famous lookahead techniques [1, Vol.1, §6.1]. The MPC method mainly exploits the forecast data. It yields a controller $\phi_t^{\mathrm{MPC}} = (\phi_0^{\mathrm{MPC}}, \ldots, \phi_{T-1}^{\mathrm{MPC}})$ by solving a sequence of *multistage deterministic* optimization problems over a fixed[1] horizon H (in the numerical application, $H = 96$)

$$\left\{ \begin{array}{ll} u_t^\star \in \arg\min\limits_{u_t} \min\limits_{(u_{t+1},\ldots,u_{t+H-1})} & \sum\limits_{s=t}^{t+H-1} L_s(u_s, \hat{w}_{t,s+1}) \ , \\ & x_{s+1} = f(x_s, u_s) \ , \ \forall s \in \{t, \ldots, t+H-1\} \ , \\ & u_s \in \mathcal{U}(x_s) \ , \ \forall s \in \{t, \ldots, t+H-1\} \ , \\ \phi_t^{\mathrm{MPC}}(x_t, h_t) = u_t^\star \ , & \end{array} \right. \tag{19}$$

where only the first value $u_t^\star$ of an optimal sequence $(u_t^\star, u_{t+1}^\star, \ldots, u_{t+H-1}^\star)$ is kept.

When used in simulation, only the simulation (testing) chronicles in $\mathcal{S}^i$ in the partition (15) are used in (19), and not the calibration (training) chronicles in $\mathcal{C}^i$; moreover, only a subvector (of available forecasted values) of the whole vector (12) of partial observations-forecasts is used, namely $(\hat{w}_{t,t+1}, \ldots, \hat{w}_{t,t+H-1})$

Due to the mathematical expressions for the dynamics (8), the constraints (9) and the cost function (11), the multistage deterministic optimization problem (19) formulates here as a Linear Program.

4.1.2 Open Loop Feedback Control

The *Open Loop Feedback Control* method belongs to the family of online lookahead methods and its approach is similar to MPC, but for a stochastic component. It yields a controller $\phi_t^{\mathrm{OLFC}} = (\phi_0^{\mathrm{OLFC}}, \ldots, \phi_{T-1}^{\mathrm{OLFC}})$ by a sequence of

[1] When the horizon extends further than the period, we truncate the lookahead window to $\min(H, T-t+1)$.

multistage open-loop stochastic optimization problems over a fixed[2] horizon H (in the numerical application, $H = 96$)

$$\begin{cases} u^\star \in \arg\min_{u_t} \min_{(u_{t+1},\dots,u_{t+H-1})} & \sum_{\sigma\in\mathbb{S}} \pi_t^\sigma \Big(\sum_{s=t}^{t+H-1} L_s(u_s, w_{t,s+1}^\sigma) \Big) \ , \\ & x_{s+1} = f(x_s, u_s) \ , \ \forall s \in \{t, \dots, t+H-1\} \ , \\ & u_s \in \mathcal{U}(x_s) \ , \ \forall s \in \{t, \dots, t+H-1\} \ , \\ \phi_t^{\mathrm{OLFC}}(x_t, h_t) = u_t^\star \ . \end{cases} \tag{20}$$

The optimization problem (20) is *open-loop* because the sequence of controls $(u_{t+1}, \dots, u_{t+H-1})$ in the (second) min is not indexed by the scenarios $\sigma \in \mathbb{S}$. Only the first value $u_t^\star$ of an optimal sequence $(u_t^\star, u_{t+1}^\star, \dots, u_{t+H-1}^\star)$ is kept.

The optimization problem (20) is *stochastic* because of the scenarios $(w_{t,t+1}^\sigma, \dots, w_{t,t+96}^\sigma)_{\sigma\in\mathbb{S}}$, together with their probabilities $(\pi_t^\sigma)_{\sigma\in\mathbb{S}}$. These scenarios and their probabilities are built from two sources: on the one hand, from the sub-vector $(\hat{w}_{t,t+1}, \dots, \hat{w}_{t,t+H-1})$ of available forecasted values given in the whole vector (12) of partial observations-forecasts in the simulation (testing) chronicles in $\mathcal{S}^i$ in (15); on the other hand, from partial observations-forecasts in the calibration (training) chronicles in $\mathcal{C}^i$ in (15), from which we calibrate a scenario generation model. In the forcoming numerical experiments in §4.3, we generate scenarios by modeling the deviations from the net demand 24-hours-forecast as a Markov chain. We detail our scenario generation method in §A.1. In numerical implementations, the number of samples used varies between 10, 50 or 100 scenarios.

4.2 Controllers obtained by cost-to-go computation methods

In cost-to-go methods, one solves online, for every step $t \in \{0, \dots, T-1\}$, a reference single stage stochastic optimization problem, which itself depends on cost-to-go functions, computed offline. These functions are called cost-to-go because, ideally, they map any state of the system to the optimal, over closed-loop strategies, expected future cost from a given time step to the final horizon. The Stochastic Dynamic Programming method is the most famous of cost-to-go computation techniques [1]. Cost-to-go methods are limited by the exponential growth of the computing time with respect to the dimension of the state space.

4.2.1 Stochastic Dynamic Programming

Stochastic Dynamic Programming is a stochastic method but, to the difference with OLFC, it is *closed-loop*.

[2] See Footnote 1.

– In the *offline phase of the* SDP *algorithm*, one computes a sequence of so-called *value functions* $(V_t)_{t=0,1,\dots,T-1,T}$ by the Bellman (or dynamic programming) equation, backward for $t \in \{0,\dots,T-1\}$,

$$V_T(x) = 0 \ , \tag{21a}$$

$$V_t(x) = \min_{u \in \mathcal{U}(x)} \sum_{\sigma \in \mathbb{S}^{\text{off}}} \pi_{t+1}^{\text{off},\sigma} \Big(L_t(u, w_{t+1}^{\text{off},\sigma}) + V_{t+1}\big(f(x,u)\big) \Big) \ . \tag{21b}$$

– In the *online phase of the* SDP *algorithm*, one computes

$$\begin{cases} u_t^\star \in \arg\min_{u_t} \sum_{\sigma \in \mathbb{S}^{\text{on}}} \pi_{t+1}^{\text{on},\sigma} \Big(L_t(u, w_{t+1}^{\text{on},\sigma}) + V_{t+1}\big(f(x,u)\big) \Big) \\ \phi_t^{\text{SDP}}(x_t, h_t) = u_t^\star \ . \end{cases} \tag{22}$$

The optimization problem (21)–(22) is *stochastic* because of the scenarios $(w_{t+1}^{\text{off},\sigma})_{\sigma \in \mathbb{S}^{\text{off}}}$, together with their probabilities $(\pi_{t+1}^{\text{off},\sigma})_{\sigma \in \mathbb{S}^{\text{off}}}$, and of the scenarios $(w_{t+1}^{\text{on},\sigma})_{\sigma \in \mathbb{S}^{\text{on}}}$, together with their probabilities $(\pi_{t+1}^{\text{on},\sigma})_{\sigma \in \mathbb{S}^{\text{on}}}$.

The scenarios indexed by $\sigma \in \mathbb{S}^{\text{off}}$, and their probabilities, are built *exclusively* from partial observations-forecasts in the calibration (training) chronicles in $\mathcal{C}^i$, in (15). The scenarios indexed by $\sigma \in \mathbb{S}^{\text{on}}$, and their probabilities, could additionally integrate partial observations-forecasts in the simulation (testing) chronicles in $\mathcal{S}^i$ in (15). The reader will find details of our scenario generation method in §A.2.

The optimization problem (21)–(22) is *closed-loop* because, under proper assumptions, it provides the optimal solution to the following *multistage stochastic optimization problem* (where the minimum is over closed-loop strategies ψ)

$$\begin{aligned} \min_{\psi} \quad & \mathbb{E}\Big[\sum_{t=0}^{T-1} L_t(\mathbf{U}_t, \mathbf{W}_{t+1}) \Big] \ , \\ & \mathbf{X}_{t+1} = f(\mathbf{X}_t, \mathbf{U}_t) \ , \ \ \forall t \in \{0,\dots,T-1\} \ , \\ & \mathbf{X}_0 = x_0 \ , \\ & \mathbf{U}_t = \psi(\mathbf{W}_0, \dots, \mathbf{W}_t) \ , \ \ \forall t \in \{0,\dots,T-1\} \ , \\ & \mathbf{U}_t \in \mathcal{U}(\mathbf{X}_t) \ , \ \ \forall t \in \{0,\dots,T-1\} \ . \end{aligned} \tag{23}$$

Problem (23) is optimally solved by the Bellman equation (21) in the case where the random variables [3] $(\mathbf{W}_0, \dots, \mathbf{W}_T)$ (noise process) are stagewise independent [1, 5].

4.2.2 SDP-AR

To account for possible stagewise dependence in the uncertainties (noise process in (23)), one can extend the state space with the observations of the i^{th} net demand lags $z_{t-i} = d_{t-i} - g_{t-i}$, $i = 1, \dots, k$. This gives the new elements

[3] Capital bold letters denote random variables.

of a model like in §3.1 wih new state $\widetilde{x}_t$, new dynamics $\widetilde{f}_t$, new stage cost $\widetilde{L}_t$ as follows, for all $t \in \{0, \dots, T-1\}$:

$$\widetilde{x}_t = (x_t, z_t, \dots, z_{t-k+1}) \in [0,1] \times \mathbb{R}^k \ , \tag{24a}$$

$$\widetilde{x}_{t+1} = \widetilde{f}_t(\widetilde{x}, u_t, \epsilon_{t+1}) \ , \tag{24b}$$

$$\widetilde{f}_t(\widetilde{x}_t, u_t, \epsilon_{t+1}) = \begin{pmatrix} f(x_t, u_t) \\ \sum_{j=0,\dots,k-1} \alpha_t^j z_{t-j} + \beta_t + \epsilon_{t+1} \\ z_t, \dots, z_{t-k+2} \end{pmatrix} , \tag{24c}$$

$$\widetilde{L}_t(\widetilde{x}_t, u_t, \epsilon_{t+1}) = L_t(u_t, \sum_{j=0,\dots,k-1} \alpha_t^j z_{t-j} + \beta_t + \epsilon_{t+1}) \ , \tag{24d}$$

where the coefficients $\alpha_t^0, \dots, \alpha_t^{k-1}$ and the additive terms β_t are the elements of an auto-regressive model of order k (noted AR(k))

$$\mathbf{Z}_{t+1} = \sum_{j=0,\dots,k-1} \alpha_t^j \mathbf{Z}_{t-j} + \beta_t + \boldsymbol{\epsilon}_{t+1} \ , \ \ \forall t \in \{0, \dots, T-1\} \ .$$

for the net demand process $\mathbf{Z}$. When the error process $\boldsymbol{\epsilon}$ is assumed to be stagewise independent, the following algorithm provides an optimal solution to the multistage stochastic optimization problem (23).

– In the *offline phase of the* SDP-AR *algorithm*, one computes a sequence of new value functions $(\widetilde{V}_t)_{t=0,\dots,T}$, backward for $t \in \{0, \dots, T-1\}$, by

$$\begin{aligned} &\widetilde{V}_T(\widetilde{x}) = 0 \ , \\ &\widetilde{V}_t(\widetilde{x}) = \min_{u \in \mathcal{U}(x)} \sum_{\sigma \in \mathbb{S}^{\text{off}}} \pi_{t+1}^{\text{off},\sigma} \Big(\widetilde{L}_t(\widetilde{x}_t, u, \epsilon_{t+1}^{\text{off},\sigma}) + \widetilde{V}_{t+1}\big(\widetilde{f}_t(\widetilde{x}_t, u, \epsilon_{t+1}^{\text{off},\sigma})\big) \Big) \end{aligned} \tag{25a}$$

– In the *online phase of the* SDP-AR *algorithm*, one computes

$$\begin{cases} u_t^\star \in \arg\min_{u_t} \sum_{\sigma \in \mathbb{S}^{\text{on}}} \pi_{t+1}^{\text{on},\sigma} \Big(\widetilde{L}_t(\widetilde{x}_t, u, \epsilon_{t+1}^{\text{on},\sigma}) + \widetilde{V}_{t+1}\big(\widetilde{f}_t(\widetilde{x}_t, u, \epsilon_{t+1}^{\text{on},\sigma})\big) \Big) \\ \phi_t^{\text{SDP-AR}}(x_t, h_t) = u_t^\star \ . \end{cases} \tag{26}$$

4.3 Results

We now comment the results obtained when applying the controller design techniques introduced in §4.1 and in §4.2 to the EMSx benchmark. Numerical experiments were run on an Intel Core Processor of 2.5 GHz with 22 Go RAM. We used the LP solver CPLEX 12.9 for MPC, OLFC and to compute the lower bounds $\underline{J}^i(h) \ , \ \ h \in \mathcal{S}^i \ , \ \ i \in I$ in (16).

4.3.1 Cost performance

Results of Table 1 show that lookahead methods (MPC, OLFC) obtain much lower scores on the EMSx benchmark than SDP -based methods. Whereas MPC uses a single scenario (the forecast), OLFC uses multiple scenarios; this helps OLFC improving the average gain of MPC from 48.7% to 51.3%. As expected, increasing the number of scenarios from 10 to 50 improves the score of OLFC. However, the performance remains stagnant when pushing up to 100 scenarios, with a slight decrease of OLFC-100 to 51.0%. We expect that the improvement between MPC and OLFC should be sensitive to the scenario generation method employed. With our method, the progress of OLFC is modest regarding the performance of a plain SDP controller which yields gains jumping to 69.1%. The results of SDP can be improved up to 79.4% in the SDP-AR formulation. However, We observe that the gain from extending the lags of an AR(1) model to an AR(2) model is almost null.

Table 1: Scores (second column) $G\big(\{\phi^i\}_{i\in I}\big)$ in (18) and time performances (third and fourth column) for collections $\{\phi^i\}_{i\in I}$ of controllers ϕ^i designed with techniques (first column) from §4.1 and §4.2 on the EMSx benchmark

	Average score	**Offline time** (seconds)	**Online time** (seconds)
MPC	0.487	0.00	$9.82\ 10^{-4}$
OLFC-10	0.506	0.00	$1.14\ 10^{-2}$
OLFC-50	0.513	0.00	$8.62\ 10^{-2}$
OLFC-100	0.510	0.00	$1.87\ 10^{-1}$
SDP	0.691	2.67	$3.09\ 10^{-4}$
SDP-AR(1)	0.794	38.1	$4.44\ 10^{-4}$
SDP-AR(2)	0.795	468	$5.55\ 10^{-4}$
Upper bound	**1.0**	-	-

In Figure 6, we display the average score per site $G^i(\phi^i)$ in (17) for controllers ϕ^i designed with MPC, SDP and SDP-AR (1) techniques, together with the maximum expected gain $\frac{1}{|\mathcal{S}^i|}\sum_{h\in\mathcal{S}^i} J^i(\phi^{\mathrm{d}},h) - \underline{J}^i(h)$. We provide this latter value to reveal the order of magnitude of the effective gains allowed by the insertion of a battery controlled by one of the benchmarked method on the pool of sites. The OLFC technique is omitted to preserve the readability of Figure 6, given that its performance appears as a slight improvement on MPC. Regarding controller design techniques, SDP-AR (1) outperforms MPC for all sites and scores higher than SDP for 69 of the 70 sites. Looking closer at the per period performance, for our total pool of 2474 testing weeks, SDP-AR (1) returns higher gains for 96% of the periods against MPC and for 90% of the periods against SDP . As for SDP , it achieves better average scores than MPC on all sites, and returns higher gains on 88% of the test periods. Figure 6 highlights the dominance of the cost-to-go methods. We observe that MPC lags behind other methods for almost all sites, and that the score gap

can be significant on a few outlying sites. The underperformance of MPC on these sites explains the score gaps of Table 1.

In particular, one site stands out in Figure 6 as the worst MPC performance with a negative score of -28.0%, meaning that MPC performs worst on average than the dummy controller ϕ^{d}. Interestingly, this site also stands out in Figure 3 as the most stochastic site of the pool, with a RMSE of 0.176. In comparison with MPC, SDP scores 18.4% and SDP-AR (1) scores 56.6% on this site. These results highlight the resilience of the cost-to-go methods for the management of a microgrid in a highly stochastic context.

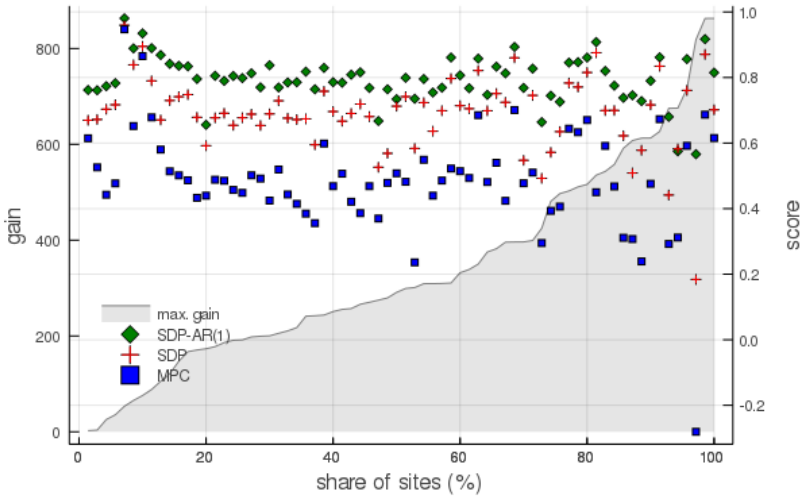

Fig. 6: Average score $G^i(\phi^i)$ per sites $i \in I$ of controller design techniques MPC, SDP and SDP-AR (1) (right Y-axis) and maximum expected gain $\frac{1}{|S^i|}\sum_{h\in S^i} J^i(\phi^{\mathrm{d}}, h) - \underline{J}^i(h)$ (left Y-axis). Sites are ranked in increasing order of maximum expected gain along the X-axis.

4.3.2 Computing time performance

Table 1 also reports average computing time performances.

Online time. We report the online time as the computing time required to yield a single control u_t. With this definition, lookahead methods require a longer online computing time for they call a LP solver at each time step; MPC achieves online computing time of the same order of magnitude as SDP-based methods; OLFC is much slower. As expected, the more we add sample scenarios, the longer the OLFC computing time. We observe that improving

the amount of scenarios from 50 to 100 doubles the online time while not returning higher gains, which discredits OLFC-100.

Offline time. Concerning the offline time, we report the average CPU time for computing value functions for cost-to-go methods. The complexity of SDP is well known for growing exponentially with the state space, which is well illustrated by our results. Given the low improvement of gain from SDP-AR (1) to SDP-AR (2), the offline time of the latter methods is dissuasive.

5 Conclusion

We have introduced the data and the key components of EMSx, an Energy Management System benchmark to compare electric microgrid controllers, hence controller design techniques. The dataset provided by Schneider Electric ensures a diversified pool of realistic microgrids with photovoltaic power integration. Besides, we make our simulation code accessible in the `EMSx.jl` package which is able to welcome various sorts of control algorithms. All components of the benchmark are publicly available, so that other researchers willing to test controllers on EMSx may reproduce experiments easily.

Regarding our numerical results, we have exhibited a gap between cost-to-go methods and lookahead methods. The SDP-AR (1) controller design technique stands out as the best trade-off between cost optimality and computing time performance. However, there is a range of possible improvements to explore. Among interesting directions, other scenario generation techniques could be tested to see how does the OLFC controller react. Beyond plain score improvements, enriching contributions could arise from the reduction of the computing time, or from changing the performance metrics (for instance with the use of risk measures), and from further comparative analysis of methods, especially to better explain the gap between lookahead and cost-to-go methods. We are also looking forward to controllers inspired from other research fields than multistage deterministic or stochastic optimization, including heuristics, robust optimization and reinforcement learning.

Acknowledgements We thank Efficacity and Schneider Electric for the PhD funding of Adrien Le Franc. Additionally, we are grateful for the feedbacks and data supply from Peter Pflaum and Claude Le Pape (Schneider Electric) and thank our colleague Tristan Rigaut (Efficacity) for insightful tips about the Julia language.

A Scenario generation

A.1 Generation of scenarios for the Open Loop Feedback Control algorithm

Our scenario generation method is inspired by [16,17], which address such generation for so-called day-ahead energy management problems. We use a simplified model for tractable generation adapted to dynamical control. Since uncertainties $w_t = (g_t, d_t)$ are directly plugged in the cost (11a) as the net demand $z_t = d_t - g_t$, we compress the generation

of scenarios $(w^{\sigma}_{t,t+1},\dots,w^{\sigma}_{t,t+96}) \in \mathbb{R}^{2\times 96}$ in (20) to the generation of net demand scenarios $(z^{\sigma}_{t,t+1},\dots,z^{\sigma}_{t,t+96}) \in \mathbb{R}^{96}$. Following the original methods, we choose day part separators to reduce the 96 dimensional vector to a few skeleton points for scenario construction (intermediate values are linearly interpolated). We choose to concentrate on the forecast error at t+15 minutes, t+1 hour, t+2 hours, t+4 hours, t+12 hours and t+24 hours, so that our model only samples values $(z^{\sigma}_{t,t+1}, z^{\sigma}_{t,t+4}, z^{\sigma}_{t,t+8}, z^{\sigma}_{t,t+16}, z^{\sigma}_{t,t+48} z^{\sigma}_{t,t+96})$. We select 10 relevant values of the net demand error at each separator $j \in \{1,4,8,16,48,96\}$ by applying the K-means algorithm [6, §13] on the historical error $z^{i}_{t+j} - \hat{z}^{i}_{t,t+j}$, combining the historical observations (1) and forecasts (3) of the calibration data of the site $i \in I$ considered. Then, we compute 10×10 transition matrices for the error between consecutive separators. With this model, we are able to sample net demand scenarios $(z^{\sigma}_{t,t+1},\dots,z^{\sigma}_{t,t+96})$ given a single value forecast and to compute their probabilities π^{σ}_{t}. Separate probability distributions of the initial error at t+1 are calibrated depending on the time of the day, and on whether the initial time step t corresponds to a week day or a weekend day. We alleviate computing costs by reusing transition matrices regardless of the initial value of t. However we calibrate separate matrices for week days and weekend days. Figure 7 provides examples of scenarios generated with our method.

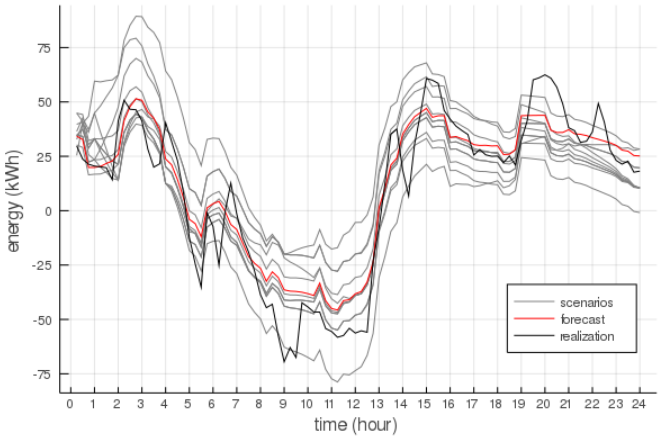

Fig. 7: Example of scenarios generated from a 24 hours net demand forecast

A.2 Generation of scenarios for the SDP and SDP-AR algorithms

For SDP methods, we choose to discretize each dimension of the state space in 10 values, whereas the control space is restricted to 20 values and the noise space to 10 values. Since uncertainties $w_t = (g_t, d_t)$ are directly plugged in the cost (11a) as the net demand $z_t = d_t - g_t$, we compress the calibration of the distributions of $(\mathbf{W}_1,\dots,\mathbf{W}_T)$ to the calibration of the distributions of $(\mathbf{Z}_1,\dots,\mathbf{Z}_T)$. We use the K-means algorithm to fit discrete probabilities $\pi^{\mathrm{off},\sigma}_{t+1}, \pi^{\mathrm{on},\sigma}_{t+1}$ on the historical observations (g^i, d^i) (1) of the calibration data of the site $i \in I$ considered. These discrete distributions serve the computation of expectations in (21)–(22). While we could leverage the data available on the fly in the *online phase*, we use the same probability distributions in the *offline phase* and in the *online phase*. Separate distributions (of $\mathbf{Z}_t$) are calibrated depending on the time of the day and on whether the time step $t+1$ corresponds to a week day or a weekend day. We compute one value function

per site for the horizon of one week. In the SDP-AR formulation, we calibrate the AR(k) models using least square regression and calibrate distributions of the residual error ($\boldsymbol{\epsilon}_t$) with the same approach as for $\mathbf{Z}_t$.

References

1. Bertsekas, D.P.: Dynamic programming and optimal control, vol. 1. Athena scientific Belmont, MA (1995)
2. Bertsekas, D.P.: Dynamic programming and suboptimal control: A survey from ADP to MPC. European Journal of Control **11**(4-5), 310–334 (2005)
3. Bertsekas, D.P.: Dynamic Programming and Optimal Control: Approximate Dynamic Programming, fourth edn. Athena Scientific (2012)
4. Bezanson, J., Karpinski, S., Shah, V.B., Edelman, A.: Julia: A fast dynamic language for technical computing. arXiv preprint arXiv:1209.5145 (2012)
5. Carpentier, P., Chancelier, J.P., Cohen, G., De Lara, M.: Stochastic multi-stage optimization. In: Probability Theory and Stochastic Modelling, vol. 75. Springer (2015)
6. Friedman, J., Hastie, T., Tibshirani, R.: The elements of statistical learning. Springer series in statistics New York (2001)
7. Garcia, C.E., Prett, D.M., Morari, M.: Model predictive control: theory and practice—a survey. Automatica **25**(3), 335–348 (1989)
8. Haessig, P., Kovaltchouk, T., Multon, B., Ahmed, H.B., Lascaud, S.: Computing an optimal control policy for an energy storage. arXiv preprint arXiv:1404.6389 (2014)
9. Hafiz, F., Awal, M., de Queiroz, A.R., Husain, I.: Real-time stochastic optimization of energy storage management using rolling horizon forecasts for residential pv applications. In: 2019 IEEE Industry Applications Society Annual Meeting, pp. 1–9. IEEE (2019)
10. Lohndorf, N., Shapiro, A.: Modeling time-dependent randomness in stochastic dual dynamic programming. European Journal of Operational Research **273**(2), 650–661 (2019)
11. Olivares, D.E., Mehrizi-Sani, A., Etemadi, A.H., Cañizares, C.A., Iravani, R., Kazerani, M., Hajimiragha, A.H., Gomis-Bellmunt, O., Saeedifard, M., Palma-Behnke, R., et al.: Trends in microgrid control. IEEE Transactions on smart grid **5**(4), 1905–1919 (2014)
12. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming, 1st edn. John Wiley & Sons, Inc. (1994)
13. Rigaut, T., Carpentier, P., Chancelier, J.P., De Lara, M., Waeytens, J.: Stochastic optimization of braking energy storage and ventilation in a subway station. IEEE Transactions on Power Systems **34**(2), 1256–1263 (2018)
14. Riseth, A.N., Dewynne, J.N., Farmer, C.L.: A comparison of control strategies applied to a pricing problem in retail. arXiv preprint arXiv:1710.02044 (2017)
15. Shapiro, A., Dentcheva, D., Ruszczynski, A.: Lectures on Stochastic Programming: Modeling and Theory, Second Edition. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2014)
16. Staid, A., Watson, J.P., Wets, R.J.B., Woodruff, D.L.: Generating short-term probabilistic wind power scenarios via nonparametric forecast error density estimators. Wind Energy **20**(12), 1911–1925 (2017)
17. Woodruff, D.L., Deride, J., Staid, A., Watson, J.P., Slevogt, G., Silva-Monroy, C.: Constructing probabilistic scenarios for wide-area solar power generation. Solar Energy **160**, 153–167 (2018)