Computers and Electrical Engineering 117 (2024) 109283

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Peak shaving and self-consumption maximization in home energy management systems: A combined integer programming and reinforcement learning approach ☆

Riccardo Felicetti, Francesco Ferracuti *, Sabrina Iarlori, Andrea Monteriù

Department of Information Engineering, Università Politecnica delle Marche, Via Brecce Bianche 12, Ancona, 60131, Italy

ARTICLE INFO

Keywords:
Reinforcement learning
Home energy management systems
Building energy management systems
Optimization
Battery storage

ABSTRACT

This paper proposes a novel framework for Home Energy Management System based on the combination of integer programming and Reinforcement Learning (RL) for achieving efficient home-based Demand Response (DR). In particular, RL is exploited to manage the charge and discharge of Battery Energy Storage System (BESS), and Mixed Integer Linear Programming is exploited for load scheduling. The idea is to focus the RL specifically on BESS management, as its behavior is stochastic and is mainly affected by Photovoltaic (PV) production and user behavior changes. The scheduling decisions of household appliances, Electric Vehicles (EVs), and charging/discharging batteries can be subsequently obtained through the newly developed framework, of which the objective is dual, i.e., to minimize the electricity bill as well as the DR-induced dissatisfaction. Simulations are performed on a residential house level with multiple home appliances, an EV, PV panels, and electric storage. The test results demonstrate the effectiveness of the proposed home energy management framework under the application of different demand-side flexibility strategies.

1. Introduction

The digitalization of energy is radically transforming the energy sector, offering products and services to allow everyone to become independent active customers, to be aware of the use of energy, and to surpass the concept of energy manager/distributor. In this context, the concepts of Energy Hubs (EHs) and micro-Energy Hubs (mEHs) have been introduced to accelerate the energy transition towards decentralized and bidirectional management systems, by employing distributed architectures, hardware, and software systems for monitoring and operating the various energy systems at different levels [1,2]. An energy hub is composed of heterogeneous mEHs that refer to energy assets and facilities belonging to the industrial, commercial, and residential sectors. The aim of an EH is to coordinate the different mEHs and manage the multiple energy carriers.

A mEH represents an integrated energy system consisting of multi-energy generation, conversion, and storage technologies to satisfy its own energy needs. So, mEHs are an evolution of the traditional distribution network and mainly have the following advantages. From the energy supply aspects, mEHs can promote local generation from Renewable Energy Sources (RESs) and self-consumption of renewable resources; moreover, it makes it possible to coordinate multi-energy demand through multi-energy

☆ This paper is for CAEE special section VSI-aismg. Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. Nick Papanikolaou.

* Corresponding author.
E-mail addresses: r.felicetti@univpm.it (R. Felicetti), f.ferracuti@univpm.it (F. Ferracuti), s.iarlori@univpm.it (S. Iarlori), a.monteriu@univpm.it (A. Monteriù).

https://doi.org/10.1016/j.compeleceng.2024.109283
Received 7 April 2023; Received in revised form 30 April 2024; Accepted 2 May 2024
Available online 11 May 2024

Table 1
List of acronyms.

Acronym	Definition	Acronym	Definition
BEMS	Building Energy Management System	LR	Learning Rate
BESS	Battery Energy Storage System	mEH	micro-Energy Hub
DR	Demand Response	MILP	Mixed Integer Linear Programming
EH	Energy Hub	PV	PhotoVoltaic
EMS	Energy Management System	RES	Renewable Energy Source
EV	Electric Vehicle	RL	Reinforcement Learning
HEMS	Home Energy Management System	SOC	State Of Charge
HVAC	Heating, Ventilation, and Air Conditioning		

technologies. MEHs can cooperate by sharing all energy carriers, to satisfy the energy needs of the entire local community represented by the EH [3]. A cluster of residential buildings constituting a local energy community and commercial buildings provided by several energy assets (Energy Management Systems (EMSs), generation from RESs, EV charging stations, etc.) are just some of the facilities belonging to mEHs. MEHs are provided with EMSs, which consist of heterogeneous information and telecommunications technologies belonging to both industrial and consumer sectors that, locally, coordinate the operation of multiple carriers, accelerating the development of multi-energy technology and improving the energy efficiency of mEHs [4].

As concerns commercial facilities, Building Energy Management Systems (BEMSs) allow to monitor and control of energy-related building services such as Heating, Ventilation, and Air Conditioning (HVAC), RESs, and lighting, excluding not energy-related systems such as safety systems, closed circuit television, etc. [5,6]. The BEMS allows to improve energy efficiency, provide better comfort for occupants, review the performance of the building, generate alarms for failures or anomaly conditions automatically, identify planned and unplanned maintenance requirements, log and archive data for energy management purposes. A Home Energy Management System (HEMS) is a specific case of BEMS deployed in the home that allows end users to manage and optimize locally their energy consumption and production, thus enabling participation in Demand Response (DR) programs and the electricity market. To achieve these goals, HEMSs usually exploit mathematical methods, particularly linear or nonlinear mixed integer programming. In recent years, however, RL has been considered in the literature for solving the problems in which uncertain environments and human interactions play major roles. One of the key characteristics of RL is its ability to interact with an uncertain environment since RL agents can learn a policy despite uncertainties and stochastic outcomes. Although the use of this approach through different solution methods has been discussed in many studies [7], its application is still a challenge in solving real problems.

The drawback of employing a single RL for the whole HEMS design is related to the dimension, often huge, of the state space needed to model the environment. Consider that a real HEMS requires the need to model different loads in the house such as dishwasher, washing machine, dryer, heat pump, air-conditioning, and EV charger. In addition to loads, RESs such as PV systems or wind turbines, and electric storage should also be considered in the environment and some studies suggest considering micro combined heat and power generators and boilers to manage multiple energy vectors as well [8–15]. Lastly, the RL agent should also consider the user's interaction with the HEMS and then the possibility of the user to own schedule the loads. So, in the end, a huge dimension of the state space (i.e., hundreds or thousands of state variables) is necessary to model a real environment with multiple energy systems, energy vectors, and interactions with the users. This issue implies the need for large datasets for training that as a result will be slow and complex. In addition, if the environment changes, multiple trainings of RL agents are necessary. Another drawback of employing a single RL is related to the difficulties of including all hard constraints in the RL formulation. Indeed, constraints are usually defined as penalties and then included in the reward: in this way, the agent learns to avoid such penalties, but the satisfaction of constraints cannot be guaranteed [16–21].

Given the above-mentioned issues, this article proposes a combined MILP and RL for the design of a HEMS. In particular, in this work, RL is exploited to manage the charge and discharge of a BESS, and MILP is exploited for load scheduling. The idea is to focus RL specifically on BESS management since it is a load/generator whose behavior is stochastic and mainly affected by PV production, a consumption different from estimated/forecasted due to the user's behavior changes. Employing both allowed us to achieve the reduction in state space and thus the ease of RL training at the expense of problem optimality as a consequence of problem decoupling. Finally, the proposed solution is easy to scale to the addition of more appliances and RES. To demonstrate the solution effectiveness, the proposed HEMS framework is tested by making use of the historical data in [22] under the application of different demand-side flexibility strategies such as peak shaving and self-consumption maximization.

The article is structured as follows. In Section 2, the HEMS and its features are introduced. In Section 3, the mathematical models for both the system and the requirements are defined for load scheduling via MILP. In Section 4 the BESS management via RL is described. The simulation results are shown in Section 5. Conclusions and future works are provided in Section 6. The complete list of acronyms is available in Table 1.

2. Problem formulation

In this Section, the HEMS and its features are introduced. In Section 2.1, appliances are classified, according to the literature, to define how they can be managed for self-consumption optimization. Section 2.2 is devoted to the BESS, where advantages and limitations are discussed. In Section 2.3, the main requirements for the HEMS design are stated. Such requirements cover a variety of goals: user satisfaction and DR capabilities are taken into account, as well as efficient load scheduling.

2.1. Load characteristics

Loads are called *interruptible* if they can be stopped and resumed, and *non-interruptible* otherwise [23]. *Shiftable* loads, instead, can be delayed with minor discomfort, thus they can be scheduled. They are also called *deferrable* loads. On the contrary, some appliances such as an oven or a fridge are *non-shiftable*, because either they need to operate when needed or they must operate continuously. Some examples of shiftable loads are a dishwasher, a washing machine, and a dryer. Such shiftable loads are non-interruptible because they rely on a predefined cycle to perform their task. We define *adjustable* (also called flexible or power-shiftable) loads as those whose power consumption can be selected between several power levels. For example, smart wallboxes can set their charge power according to the local grid capacity. Most of the small appliances and consumer electronics devices are non-interruptible and non-shiftable, because of the high impact on user satisfaction. In this work, shiftable loads are going to be scheduled by an optimization algorithm that takes into account consumptions and renewable energy prediction, based on the historical data reported in [22]. Non-shiftable loads are prioritized, while adjustable loads are managed to balance both user and constraint satisfaction.

2.2. Battery energy storage system

BESS are a key factor to increase self-consumption in HEMS. Among the advantages of BESS, they can be employed to perform distributed load balancing, i.e., to flatten the load profile of electricity usage, thus decreasing the peak demand and increasing the load factor in a smart grid framework. On the other hand, BESS are expensive and subject to irreversible aging processes involving the batteries, which are usually Li-ion batteries. The aging process is usually of two types: calendar aging and cycle aging [24]. Calendar aging occurs over time, disregarding the actual use of the battery, and is mainly affected by storage temperature and the State Of Charge (SOC). Very briefly, mild room temperatures (10 °C–25 °C) and medium-low SOC (30%–70%) ensure the best longevity. Cycle aging, in contrast, depends on battery usage. Solid experimental results show that high C-rates, both in charge or discharge, reduce the useful life, as well as extreme temperature and large depth of discharge. Taking into account battery aging, both the sizing and the actual usage of the BESS should be improved for revenue maximization [25]. In this work, we deal with aging by introducing constraints on the C-rate and SOC. In the proposed strategy, the role of the BESS is crucial, as it makes it possible to follow the task scheduling despite the presence of several uncertainties, e.g., consumption and production forecast.

2.3. Requirements

In the following, the requirements are split into two categories. The requirements in Section 2.3.1 come from technical system constraints, and they must be satisfied for safety and operability reasons. The remaining requirements in Section 2.3.2, instead, define how the HEMS should behave while performing its main task, i.e., maximization of self-consumption. Such requirements define several needs and thus they may conflict with each other, as well as with system limitations, so an order of priority is defined.

2.3.1. System limitations (hard constraints)

The following constraints must be taken into account to comply with system limitations.

I. Non-shiftable loads cannot be influenced by the HEMS.
II. Non-interruptible loads cannot be paused or turned off once they are started.
III. Self-produced renewable energy is self-consumed whenever possible.
IV. Self-production surplus is stored if possible, otherwise, it is injected into the grid.
V. Power and SOC constraints must be respected.

The first and second constraints come from the load characterization. The third and fourth constraints are usually imposed by the BESS; nonetheless, they do not represent a limitation for the purpose of self-consumption maximization. The fifth constraint takes into account practical limitations, such as the charge and discharge C-rate of the battery (e.g., for safety and longevity), maximum capacity of the battery, local grid capacity, etc.

2.3.2. Desired behavior (soft constraints)

The following constraints, which are listed in decreasing priority order, represent how the HEMS should act, according to the stakeholders.

VI. Shiftable loads must satisfy termination constraints.
VII. Limit power draws up to η kW.
VIII. Feed adjustable loads with the required power.
IX. Recharge the EV battery.
X. Terminate shiftable loads as soon as possible.

Constraint VI. allows users to define their own needs on task termination. Tight termination constraints for shiftable loads (or even imposing an immediate start) reduce the margin for optimization, eventually leading to infeasibility due to power limits if the constraints are hard. In general, planning task termination in advance provides better results. Constraint VII. is introduced to perform peak shaving, and η is a tunable parameter. Constraints VIII. and IX. force the adjustable loads and the wallbox to work whenever possible, i.e., when there is sufficient available power. Finally, constraint X. is a mild incentive to start shiftable loads sooner, i.e. to prefer early termination instead of late termination. Starting shiftable loads immediately, if possible, is also a prudential choice: additional user requests may arrive in the future, making constraint satisfaction more involved.

2.3.3. Objective

The problem can be formulated as a multi-objective optimization problem. First of all, the cost of power draw should be minimized. Variable pricing is allowed, provided that a price prediction is available, thus allowing for economic DR. Also, peak shaving capabilities are of interest to deal with DR. Finally, user satisfaction must be taken into account as well.

3. Load scheduling via mixed integer linear programming

In this Section, we propose a mathematical model for both the system and the requirements that have been introduced in Section 2. First of all, a set of variables is introduced in Section 3.1. Such variables are then exploited to define both the constraints (Section 3.2) and the cost function (Section 3.3).

In the remainder of the article, we denote by $\mathbb{R}$ the set of real numbers, with $\mathbb{N}$ the set of natural numbers including zero, and with $\mathbb{N}_k$ the set of the first k elements of $\mathbb{N}$, i.e., $\{i \in \mathbb{N} : i < k\}$, or, analogously, $\mathbb{N}_k = \{0, 1, \dots, k-1\}$. Moreover, $\mathbb{X}^{a \times b}$ represents a matrix with a rows and b columns over a given field $\mathbb{X}$.

3.1. Variables

3.1.1. Decision variables

We define the following decision variables that are going to be employed to model the system:

- $x = [x_{i,j}] \in \mathbb{N}_2^{n_s \times h_p}$ represents the shiftable load start, i.e., $x_{i,j} = 1$ if the ith shiftable load is started after j time steps, and 0 otherwise. In detail, n_s is the number of shiftable appliances, while h_p is the length of the prediction horizon.
- $w = [w_{i,j}] \in \mathbb{N}_{n_{lev}}^{n_a \times h_p}$ represents the power level provided to the ith adjustable load at each time step j in the control horizon. The power level is assumed to be discretized, as many adjustable loads can only assume a finite set of power levels (e.g., heaters), and n_{lev} is the cardinality of such a set of power levels. Moreover, n_a is the number of adjustable loads.

3.1.2. User inputs

The user inputs that are going to be taken into account in the system model are defined as:

- $u = [u_i] \in \mathbb{N}_2^{1 \times n_s}$ models the user requirement to start a shiftable load, i.e., $u_i = 1$ if the ith shiftable load task is pending (required but not started yet), and 0 otherwise, and n_s is the number of shiftable appliances.
- $v = [v_i] \in \mathbb{N}_{n_{lev}}^{1 \times n_a}$ models the desired power level of the adjustable loads, as set by the user. No information about the future values of v is available, i.e., the user can modify it at his will and the scheduler should comply with the new requirement.
- $u_{wb} \in \mathbb{N}_2$ is a scalar that defines whether the EV is connected to the wallbox for recharging ($u_{wb} = 1$) or not ($u_{wb} = 0$).

3.1.3. Known constants

The following constants, which must be fed to the optimization algorithm to fulfill its task, are assumed to be known in advance:

- $c^0 = [c_j^0] \in \mathbb{R}^{1 \times h_p}$ defines the unitary cost of the power draw at each time step in a day. It allows for economic DR, encouraging load shifting in the most economic time steps.
- $c_{imm}^0 = [c_{imm,j}^0] \in \mathbb{R}^{1 \times h_p}$ is the income for injecting energy back to the grid at each time step in a day.
- $c_{station}^0 = [c_{station,j}^0] \in \mathbb{R}^{1 \times h_p}$ represents the cost of power in a public EV charging station at each time step in a day.
- $d = [d_i] \in \mathbb{N}^{1 \times n_s}$ is the overall duration (in time steps) of ith shiftable load cycle.
- $p^0 = [p_{i,j}^0] \in \mathbb{R}^{n_{tot} \times l_{cycle}}$ is the average power consumption of the ith load at each time step j, and $n_{tot} = n_s + n_a + n_{ns}$ is the total number of appliances. If the ith load is shiftable, it represents the predicted consumption for a single cycle, once it starts. If the ith load is adjustable, it is the average consumption at the lowest power level, and it is constant $\forall j$. If the ith load is non-shiftable, it is the average quarter-hourly consumption in a day, obtained by the historical data.
- $e^0 = [e_j^0] \in \mathbb{R}^{1 \times h_p}$ is the average power produced from renewable sources at each time step in a day.
- $t_s \in \mathbb{R}$ is the duration of each time step.

Please note every appliance may exhibit different behaviors depending on unpredictable factors. For example, the power draw of a washing machine depends on the water temperature, the load, and the type of cycle set by the user. So, the actual production and consumption, which are not available in real applications, are replaced by their predictions. The predictions are based on the average values of historical data, and the behavior of the control system with such predictions is then tested in simulation. The superscript

$(\cdot)^0$ remarks that the costs correspond to the time of the day starting from midnight so such variables are independent of the current time, assuming that the costs c^0, c^0_{imm}, $c^0_{station}$, and e^0 show a daily cyclic nature. This last assumption is not a limitation: such quantities are only employed for calculating the time-varying quantities in the remainder, which can be defined in any alternative way.

The following time-varying quantities, which are necessary to perform the optimization, can be calculated before each run of the optimization algorithm starts:

- $t \in \mathbb{N}_{h_p}$ is the current time step, expressed in terms of time steps from the last midnight. It is employed to perform cyclic permutations of the constants, in order to synchronize them with the current time (e.g., find the next four time-varying constants).
- $c = [c_j] \in \mathbb{R}^{1\times h_p}$ is the unitary cost of the power draw in the next h_p time steps.
- $c_{imm} = [c_{imm,j}] \in \mathbb{R}^{1\times h_p}$ is the income for injecting energy into the grid in the next h_p time steps.
- $c_{station} = [c_{station,j}] \in \mathbb{R}^{1\times h_p}$ represents the cost of power in a public EV charging station in the next h_p time steps.
- $e = [e_j] \in \mathbb{R}^{1\times h_p}$ is the average power produced by renewable sources in the next h_p time steps.
- $e_{wbmax} \in \mathbb{R}$ is the maximum amount of energy that can be stored in the battery, given the current SOC.
- $\Delta t = [\Delta t_i] \in \mathbb{N}^{1\times n_s}$ is the maximum termination time (time step count) declared by the user for the ith shiftable load.
- $\Delta t_{wb} \in \mathbb{N}$ is the maximum termination time for EV battery recharging.
- $fr = [fr_i] \in \mathbb{N}^{1\times n_s}$ is the remaining time until the ith shiftable load finishes its task (zero if not in execution). It is exploited to avoid load interruption for tasks that are already in progress.
- $M_s \in \mathbb{R}$, $M_\mu \in \mathbb{R}$, $M_\sigma \in \mathbb{R}$, $M_u \in \mathbb{R}$, $M_v \in \mathbb{R}$, $M_{wb} \in \mathbb{R}$ are positive constants that penalize some unwanted events.

3.1.4. Auxiliary variables

The following time-varying quantities are defined within the optimization algorithm and depend on the decision variables:

- $p = [p_{i,j}] \in \mathbb{R}^{n_{tot}\times h_p}$ is the prediction of the power consumption of each ith load in the next h_p time steps. It depends on the load scheduling x, the nominal consumption p^0, and the current time t.
- $a = [a_j] \in \mathbb{R}^{1\times h_p}$ is the prediction of self-consumption in the next h_p time steps, according to the consumptions of p and the PV generation forecast e.
- $s = [s_j] \in \mathbb{R}^{1\times h_p}$ quantifies the violation of the peak shaving constraint.
- $\sigma = [\sigma_j] \in \mathbb{N}_2^{1\times n_s}$ quantifies the violation of the termination constraints for shiftable loads.
- $\mu \in \mathbb{N}_3$ is strictly positive if the adjustable loads are severely underpowered.

3.2. Constraints

3.2.1. Hard constraints

In the following, the constraints that have been listed in Section 2.3.1 are modeled.

Constraint I. from Section 2.3.1 is modeled by

$$p_{5,j} = p^0_5, \tag{1}$$

that is, the overall consumption of non-shiftable loads is supposed to be equal to the quarter-hourly estimation, thanks to historical data.

Constraint II. from Section 2.3.1 is formulated as

$$\sum_{j=1}^{h_p} x_{i,j} \le 1, \quad \forall i \in \left[1, \dots, n_s\right] \tag{2}$$

$$x_{i,j} = 0 \quad \forall i \in [1, \dots, n_s], \forall j \in \left[h_p - l_{cycle} + 1, \dots, h_p\right] \tag{3}$$

$$\text{if } x_{i,j} = 1 \text{ then } \quad \left[p_{i,j}, \dots, p_{i,j+l_{cycle}}\right] = \left[p^0_{i,1}, \dots, p^0_{i,1+l_{cycle}}\right], \quad \forall i \in \left[1, \dots, n_s\right], \forall j \in \left[1, \dots, h_p - l_{cycle}\right] \tag{4}$$

$$\text{if } fr_i > 0 \text{ then } \quad p_{i,j} = p^{old}_{i,j+1}, \ \forall i \in [1, \dots, n_s], \forall j \in \left[1, \dots, fr_i\right] \tag{5}$$

Eq. (2) limits the number of starts to 1 for each appliance. Eq. (3) imposes that no load can start if its termination may occur beyond the prediction horizon. Eq. (4) states that whenever a shiftable appliance i starts in the time step j ($x_{i,j} = 1$), then the related energy consumption, from the beginning to the end of the task, is equal to the average of the related historical data. The last Eq. (5) makes it possible to keep the memory, in a receding horizon framework, of non-interruptible tasks that are already in progress.

Remark 1. Logical implications represent non-convex constraints, and they are implemented by making use of the well-known big-M approach.

To enforce constraint III. from Section 2.3.1, the following equality constraint is employed

$$a = \min\left(e, \mathbf{1}^T p\right), \tag{6}$$

where min is the element-wise minimum, and $\mathbf{1}$ is a column vector of ones such that $\mathbf{1}^T p$ is the overall consumption (sum of all appliances) at each time step (i.e., in this case, $\mathbf{1}^T \in \mathbb{R}^{1\times n_{tot}}$). So, the predicted self-consumption is maximized, because all of the self-produced renewable energy is employed to power the scheduled loads.

Finally, the following hard constraints are enforced for real-valued variables:

$$0 \le p_{i,j} \le \eta \quad \forall i = [1,\dots,n_{tot}],\ \forall j = [1,\dots,h_p] \tag{7}$$

$$0 \le a_j \le 2\eta \quad \forall j = [1,\dots,h_p] \tag{8}$$

$$0 \le s_j \le 2\eta \quad \forall j = [1,\dots,h_p] \tag{9}$$

Constraints IV. and V. from Section 2.3.1 are introduced later (Section 4.2), as they directly involve the BESS.

Remark 2. The constraints I.–V. from Section 2.3.1 do not compromise the feasibility of the problem, as long as non-interruptible loads can be powered, which is a reasonable assumption. A feasible (but undesirable, due to severe user dissatisfaction) solution to the set of constraints I.–V. is simply deactivating every shiftable/adjustable load, as well as disconnecting the BESS, while maximizing self-consumption. Thus, the problem is feasible, despite the presence of hard constraints.

3.2.2. Soft constraints

In this Subsection, we deal with the mathematical definition of the soft constraints in Section 2.3.2. Such constraints can potentially generate infeasible problems, thus they are softened to avoid a failure of the optimization algorithm. Constraint VI. from Section 2.3.2 is modeled as follows:

$$1 - \sum_{j=1}^{\Delta t_i - d_i} x_{i,j} = \sigma_i \quad \forall i \in [1,\dots,n_s] : u_i = 1 \tag{10}$$

$$\sigma_i = 0 \quad \forall i \in [1,\dots,n_s] : u_i = 0 \tag{11}$$

where σ_i is equal to 1 if the termination constraint for the ith appliance is not satisfied, $\sigma_i = 0$ otherwise. To encourage satisfaction of the termination restriction, a penalty proportional to $\sum_{i=1}^{n_s} \sigma_i$ is added to the cost function. In other words, if the user asks for the ith appliance to run ($u_i = 1$), then it should end before its predetermined termination time Δt_i, taking into account the duration of the task d_i. As soon as the ith load is started, the variable u_i is reset to 0.

Constraint VII. from Section 2.3.2 is modeled by:

$$\sum_{i=1}^{n_{tot}} p_{i,j} \le e_j + \eta + s_j \quad \forall j \in [1,\dots,h_p]. \tag{12}$$

In other words, for every time step j in the prediction horizon, the sum of the loads must be lower than the peak shaving limit η (if not applicable, η is the maximum power draw), plus the expected renewable energy production e_j. The variable s_j is used to relax the problem and ensure its feasibility.

Constraint VIII. from Section 2.3.2 is represented by:

$$p_{4,j} = p_4^0 w_{4,j} \quad \forall j \in [1,\dots,h_p] \tag{13}$$

$$w_j \le v \quad \forall j \in [1,\dots,h_p] \tag{14}$$

$$w_j \ge v - 1 - \mu \quad \forall j \in [1,\dots,h_p] \tag{15}$$

Eq. (13) means that the ith adjustable load can assume a finite set of power draw values, that are multiples of $p_{i,j}^0$. Please note $p_{i,j}^0 = p_{i,k}^0\ \forall j,k \in [1,\dots,h_p]$ in case of adjustable loads, as explained in Section 3.1.3. Eq. (14) states that the load consumption should not overcome the nominally required power draw. We do not impose exact user satisfaction ($w_j = v$), and we relax such constraint by adding a penalty term (proportional to $v - w_j$) in the cost function. Finally, Eq. (15) is an optional soft constraint, stating that the power load may be lowered only in part.

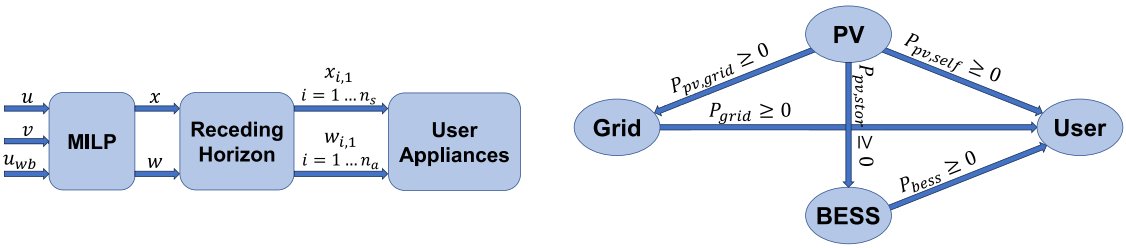
Constraint IX. from Section 2.3.2, models EV battery recharging:

$$e_{wb} \le e_{wbmax} \tag{16}$$

$$\text{if } u_{wb} = 1 \text{ then } \sum_{j=1}^{\Delta t_{wb}} p_{6,j} t_s = e_{wb} \tag{17}$$

where t_s is the sampling time for the MILP solver and e_{wbmax} is the remaining energy to be stored to fully charge the battery. To soften the constraint, we do not impose the actual energy e_{wb} to be equal to the required one for a full charge e_{wbmax} (i.e., full recharging), instead, we penalize the difference $e_{wbmax} - e_{wb}$ in the cost function.

Finally, no additional equations are needed to model constraint X. and it can be directly included in the cost function by adding a penalty, which is proportional to $\sum_{i=1}^{n_s} \sum_{j=1}^{h_p} \left(j\, x_{i,j}\right)$. In other words, the higher j when the ith shiftable load starts, the higher the penalty.



(a) MILP scheduling: receding horizon.

(b) BESS management: flow network.

Fig. 1. Load scheduling and BESS management diagrams.

3.3. Cost function

The cost function takes into account the unitary cost of energy, as well as user satisfaction and constraint satisfaction. In order to fulfill the objective in Section 2.3.3, we introduce the following scalars:

$$J_e = c\left(-a + \mathbf{1}^T p\right)^T - c_{imm}\,(e-a)^T \tag{18}$$

$$J_{ko} = \sum_{j=1}^{h_p} M_s s_j \tag{19}$$

$$J_d = \sum_{i=1}^{n_s}\sum_{j=1}^{h_p}\left(j\,M_u\,x_{i,j}\right) + M_\sigma \sum_{j=1}^{h_p}\sigma_j + M_v \sum_{j=1}^{h_p}(v - w_j) + M_\mu \mu \tag{20}$$

$$J_{wb} = c_{station}(e_{wbmax} - e_{wb})^T + \sum_{j=1}^{h_p}\left(j\,M_{wb}\,p_{6,j}\right), \tag{21}$$

where J_e is the cost of power drawn from the grid, J_{ko} is the cost of peak shaving violation, J_d is the cost of user dissatisfaction (violation of termination constraints, waiting for shiftable loads, reduction of adjustable loads), and J_{wb} is the cost of replenishing the battery level after disconnection (i.e., the non-charged capacity) in a public charging station, plus a small penalty term for delaying the EV charging. Such a multi-objective optimization problem is reduced to a conventional single-objective minimization problem through linear scalarization. The scalar cost function to be minimized is

$$J = J_e + J_{ko} + J_d + J_{wb}, \tag{22}$$

where J_e and J_{wb} are homogeneous (both represent a price), while the scaling factors for J_{ko} and J_d are included in the definition of M_s and M_v.

3.4. Solver

The resulting optimization problem is a MILP, thus it can be solved online in a reasonable time (each run of the MILP scheduler usually takes around 3 s in the simulated scenario, see Section 5 for further details). The whole optimization process is performed in MATLAB [26]: the problem is modeled using YALMIP [27] and then solved using MOSEK [28]. The solver operates within a receding horizon framework (refer to Fig. 1(a)). The optimization problem on the prediction horizon $[1, h_p]$ is solved. Subsequently, only the initial commands $x_{i,1}$ for $i = 1, \dots, n_s$ and $w_{i,1}$ for $i = 1, \dots, n_a$ of the optimal sequence are transmitted to the appliances. After a time step of t_s, the optimization process is repeated over the prediction horizon $[2, h_p + 1]$, etc.

4. BESS management via RL

The task scheduling shown in Section 3 cannot be implemented in practice unless some kind of flexibility is introduced in the system. The MILP solver relies on consumption and production forecasting, which are estimated thanks to historical data: hence, the expected and the actual consumption may be substantially different. For example, the power load of each appliance may depend on several unknown factors that cannot be taken into account. A meaningful example of such factors is given by user settings: although they could be queried by the MILP solver in the theory, and then exploited to refine future load estimation, many practical limitations arise: need for exhaustive data to perform estimations, need for additional APIs to communicate with the appliances, increased complexity of the optimization problem, etc.

The required robustness to uncertainties is given by the BESS, which acts as an energy buffer to be employed whenever necessary. There is no trivial method to exploit the BESS. A greedy policy, such as drawing energy from the BESS whenever possible, leads to an optimal self-consumption; on the other hand, the battery is often depleted, thus it lacks robustness concerning any unpredictable increase of the energy demand. In contrast, if a conservative policy is adopted and the BESS is discharged only when needed to

counteract a demand peak, then the battery is often fully charged: in this case, there is no room for storing the surplus energy given by PV, and hence self-consumption is poor.

The objective of the RL agent is twofold: to maximize self-consumption, as well as to make the MILP scheduling viable despite the presence of uncertainties, ie to perform the tasks without violating the peak shaving constraint. Also, the BESS should manage the power peaks between samples that the MILP scheduler (which necessarily runs at low speed due to computational reasons) cannot detect.

The RL agent must learn how to employ the energy stored in the BESS to supply the loads while maximizing self-consumption. Practically speaking, the action of the RL agent consists of defining how much energy should be drawn from the battery to feed the loads.

4.1. Environment model

The environment for RL is represented by the directed graph in Fig. 1(b), where P_{grid} is the power draw from the grid to the appliances, P_{bess} is the power flow from the BESS to the appliances, $P_{pv,grid}$ is the power injection from the PV plant back into the grid, $P_{pv,stor}$ is the power generated by the PV to be stored in the BESS, $P_{pv,self}$ is the power generated by the PV and fed to the appliances (i.e., self-consumption), and P_{loads} is the overall power required by the appliances. Please note that the graph in Fig. 1(b) is not fully connected. Each variable is non-negative, which means that the power can flow only in the direction of the arrow. Also, note that we have neglected reactive power in this diagram. In practice, the influence of reactive power on household appliances is still limited to low-power devices. In the case of energy-intensive and/or major appliances with inherently low power factor, it is addressed using dedicated power factor correction devices to ensure compliance with regulations on power quality. Also, recharging the BESS directly from the grid is neglected, because the objective is to maximize self-consumption, so loading the battery from the grid is unprofitable. Finally, recharging the BESS from the grid increases battery wear and tear and it should be avoided, and power flow from the BESS to the grid is neglected for the same reason.

4.2. Environment constraints

In this Subsection, the constraints in Section 2.3.1 are introduced in the BESS optimization. First of all, the power balance between the supply and the demand side in Fig. 2 is modeled by

$$P_{grid}(\tau) + P_{pv,self}(\tau) + P_{bess}(\tau) = P_{loads}(\tau) \quad \forall \tau. \tag{23}$$

The reinforcement learning algorithm must strictly satisfy this condition. Please note that the actual $P_{loads}(\tau)$ is influenced by the task scheduling coming from the MILP optimization, as well as by random variations of load consumption. Analogously, the actual $P_{pv,self}(\tau)$ cannot be predicted precisely, and its estimation (i.e., the variable a in the MILP optimization) is different from the actual value. The modeling of such uncertainty is detailed in Section 5.

To enforce constraints III. and IV. from Section 2.3.1, the following equality constraints are employed

$$P_{pv,self}(\tau) = \min\left(P_{pv}(\tau), P_{loads}(\tau)\right) \tag{24}$$

$$P_{pv,stor}(\tau) = \min\left(P_{pv}(\tau) - P_{pv,self}(\tau), (E_{batt,max}(\tau) - E_{batt}(\tau))/\delta\tau\right) \tag{25}$$

$$P_{pv,grid}(\tau) = P_{pv}(\tau) - P_{pv,self}(\tau) - P_{pv,stor}(\tau). \tag{26}$$

Eq. (24) maximizes instantaneous self-consumption (constraint III.), given the current PV energy production $P_{pv}(\tau)$ and the current sum of loads $P_{loads}(\tau)$. Eq. (25) forces the power surplus $P_{pv}(\tau) - P_{pv,self}(\tau)$ to be stored in the BESS, provided that sufficient capacity is available, where $E_{batt,max}(\tau)$ is the full charge capacity (eventually taking into account battery wear level at time τ), $E_{batt}(\tau)$ is the current remaining capacity, and $\delta\tau$ is the algorithm sampling time (which can be different from MILP scheduling sampling time, i.e., far way faster than the latter). In particular, $\delta\tau$ is chosen such that all of the aforementioned variables are essentially constant in that time interval. Eq. (26) defines the remaining power to be injected into the grid. It is also possible to limit power injection into the grid, if necessary, adding the constraint $P_{pv,grid} \leq P_{pv,grid,max}$, where $P_{pv,grid,max}$ is the upper power limit. Finally, constraint V. from Section 2.3.1 is modeled by:

$$P_{pv,stor}(\tau) \leq P_{pv,stor,max} \tag{27}$$

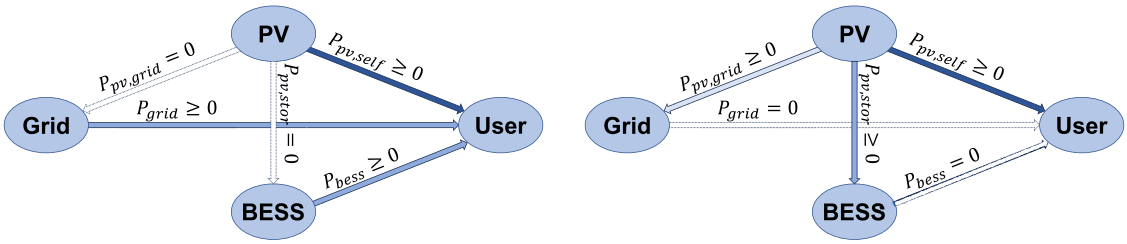
$$P_{bess}(\tau) \leq P_{bess,max} \tag{28}$$

$$E_{batt,min}(\tau) \leq E_{batt}(\tau) \leq E_{batt,max}(\tau) \tag{29}$$

where custom power and SOC limitations can be enforced. According to the model and the constraints, in the remainder we detail the two possible scenarios: renewables deficit (Fig. 2(b)) and renewables surplus (Fig. 2(a)).

4.2.1. Renewables deficit

When the PV power is not sufficient to feed the loads ($P_{pv}(\tau) < P_{loads}(\tau)$, see Fig. 2(a)), the PV production is fully employed to feed the loads, and no surplus can be stored. The PV power is insufficient to feed the loads, so additional power is needed. The task of the RL agent is to decide how much power should come from the BESS and, consequently, how much power is taken from the grid. In this scenario, a flat battery in the BESS is undesired, because more energy must be drawn from the grid, which is the most expensive source. Moreover, peak shaving constraints may be violated to feed the loads.



(a) Energy flow network when $P_{pv}(\tau) < P_{loads}(\tau)$.

(b) Energy flow network when $P_{pv}(\tau) \geq P_{loads}(\tau)$.

Fig. 2. Energy flow networks. Darker lines highlight higher priority in the flow.

4.2.2. Renewables surplus

When the PV power is sufficient to feed the loads ($P_{pv}(\tau) \geq P_{loads}(\tau)$, see Fig. 2(b)), the PV production is strictly split into self-consumption, storage, and grid injection, according to (24)–(26). Moreover, as the PV power is sufficient to feed the loads, there is no need to take power from the BESS nor from the grid. Hence, there is no degree of freedom, and the RL agent cannot operate any choice. In this scenario, having a fully replenished battery is undesired because no more PV energy can be accumulated: the surplus energy is injected into the grid, which is less convenient than storing it, and overall self-consumption is reduced because delayed self-consumption decreases.

4.3. State dynamics

The most relevant state to be modeled is the battery charge $E_{batt}(\tau)$. It depends on the action of the RL agent, the power demand, and the PV energy production. The following discrete-time system is employed to describe the battery charge dynamics:

$$E_{batt}(\tau + \delta\tau) = E_{batt}(\tau) + \rho P_{pv,stor}(\tau)\delta\tau - P_{bess}(\tau)\delta\tau/\rho, \tag{30}$$

where ρ is the BESS efficiency (which is assumed, without loss of generality, to be equal in the charge and discharge phase). Another relevant information is time (time of the day, season, etc.). Its dynamics are trivial, so modeling is omitted.

Remark 3. Additional information could be made available to the RL agent, such as PV energy forecast, prediction of consumption according to scheduled loads, etc. However, the larger the state and the observation space, the more involved is the training of the RL agent. Because of a deployment, a small observation space entails faster learning; on the other hand, the performances are expected to increase if additional information is provided to the agent.

4.4. Observations

The RL agent is trained using three observations: the battery charge status (i_1), the time of the day (i_2), and the month (i_3). In particular, to ease the training of the RL agent, input quantization is performed. The first input $i_1 \in \mathbb{N}_5$, which represents the battery charge status, is chosen as

$$i_1(\tau) = \left\lfloor 4 \frac{E_{batt}(\tau) - E_{batt,min}(\tau)}{E_{batt,max}(\tau) - E_{batt,min}(\tau)} \right\rceil, \tag{31}$$

where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer. The second input $i_2 \in \mathbb{N}_{24}$ is the current hour of the day. The third input i_3 is the month. For the sake of brevity, we are going to perform training and testing in a single month, i.e., July, thus we neglect i_3 in the remainder. The input i_3 can be included to generalize the algorithm to the entire year; alternatively, 12 separate RL agents can be trained to cover the entire year and i_3 can be omitted, as month switches very slow with respect to the battery charge and discharge cycle. The size of the observation space, which is defined by the cartesian product of $\mathbb{N}_5$ and $\mathbb{N}_{24}$, is 120. This size increases by a factor of 12 if every month is considered.

4.5. Action

The RL agent must decide how much power $P_{bess}(\tau)$ should be drawn from the battery to feed the current loads and, consequently, how much power should be drawn from the grid instead. In order to greatly simplify the training while guaranteeing the satisfaction of BESS constraints, we set the actual power as a convex combination of a greedy policy $P_{bess}^{greedy}(\tau)$ and a conservative policy $P_{bess}^{cons}(\tau)$, i.e.,

$$P_{bess}(\tau) = (1 - \alpha(\tau))P_{bess}^{cons}(\tau) + \alpha(\tau)P_{bess}^{greedy}(\tau), \tag{32}$$

where $\alpha(\tau)$ is a value between 0 and 1.

In the following, we suppose that the quantities in (24)–(26) can be exactly calculated by the agent (i.e., without uncertainty), according to the current load power demand and the current PV production. This assumption is not restrictive as long as $\delta\tau$ can be arbitrarily reduced, so we can assume that the current power is constant throughout the time interval $\delta\tau$.

The most conservative policy is to draw energy from the battery only to satisfy the peak shaving constraint:

$$P_{bess}^{cons}(\tau) = \min\left(\max\left(0, P_{loads}(\tau) - P_{pv,self}(\tau) - \eta(\tau)\right), P_{bessmax}(\tau), \rho(E_{batt}(\tau) - E_{batt,min})/\delta\tau\right) \tag{33}$$

In other words, $P_{bess}^{cons}(\tau)$ is set equal to $P_{loads}(\tau) - P_{pv,self}(\tau) - \eta(\tau)$, then lower and upper limits are enforced due to the constraints. Please note that $P_{bess}^{cons}(\tau) = 0$ holds as long as the grid can supply the loads without overcoming the peak shaving constraint. If $P_{bess}(\tau) = P_{bess}^{cons}(\tau)$ is chosen, then the battery is discharged only if the power from the grid is not sufficient to feed the scheduled loads.

On the contrary, the greedy policy is given by

$$P_{bess}^{greedy}(\tau) = \min\left(P_{loads}(\tau) - P_{pv,self}(\tau), P_{bessmax}, \rho(E_{batt}(\tau) - E_{batt,min})/\delta\tau\right). \tag{34}$$

If $P_{bess}(\tau) = P_{bess}^{greedy}(\tau)$ is chosen, then self-consumption is maximized and the power drawn from the grid is minimized.

Consider (32). If $\alpha(\tau) = 0$, then $P_{bess}(\tau) = P_{bess}^{cons}(\tau)$, so no power is taken from the battery whenever possible. In contrast, if $\alpha(\tau) = 1$, the battery is exploited as much as possible, i.e., by covering the entire load demand $P_{loads}(\tau) - P_{pv,self}(\tau)$ while satisfying the constraints of the battery power and the residual capacity of the battery. Please note that once $P_{bess}(\tau)$ is set, the amount of energy from the grid $P_{grid}(\tau)$ is uniquely defined to satisfy the power balance constraint in (23).

In short, the RL agent must set $\alpha(\tau)$ at each time step, based on available observations, to define whether a greedy discharge or a conservative discharge should be performed. As done for the observations, action quantization is performed as well to ease the RL agent training; hence, the action is restricted to $\alpha(\tau) \in \begin{bmatrix}0 & 0.25 & 0.5 & 0.75 & 1\end{bmatrix}$. To explore the state space, we employ a conventional ϵ-greedy strategy. The parameter ϵ is set to decrease exponentially during training, promoting more exploration in the early stages.

4.6. Reward function

Two objectives must be taken into account by the RL agent. First of all, the self-consumption must be maximized, i.e.,

$$R_{self}(\tau) = (\rho\, c(\tau) - c_{imm}(\tau))P_{bess}(\tau) \tag{35}$$

where $(\rho\, c(\tau) - c_{imm}(\tau))$ represents the cost saving when the energy is self-consumed (instead of grid injection). The term $R_{self}(\tau)$ is a positive reward to encourage self-consumption. Moreover, the peak shaving limit should be respected, so we define

$$R_{peak}(\tau) = \begin{cases} -M_{rl}(P_{grid}(\tau) - \eta) & \text{if } P_{grid}(\tau) - \eta > 0 \\ 0 & \text{otherwise} \end{cases} \tag{36}$$

where $-M_{rl}(P_{grid}(\tau) - \eta)$ is a penalty to discourage peak shaving violations (i.e., the penalty for drawing too more power from the grid than allowed), and M_{rl} is a weighting factor to perform linear scalarization between the two objectives. Hence, the overall reward function is:

$$R(\tau) = R_{self}(\tau) + R_{peak}(\tau) \tag{37}$$

$$G(\tau) = \sum_{k=0}^{\infty} \gamma_{rl}^{k} R(\tau + k + 1), \tag{38}$$

where $G(\tau)$ is the future reward, which is of interest for the RL training according to [29], and γ_{rl} is the discount rate.

5. Simulation results

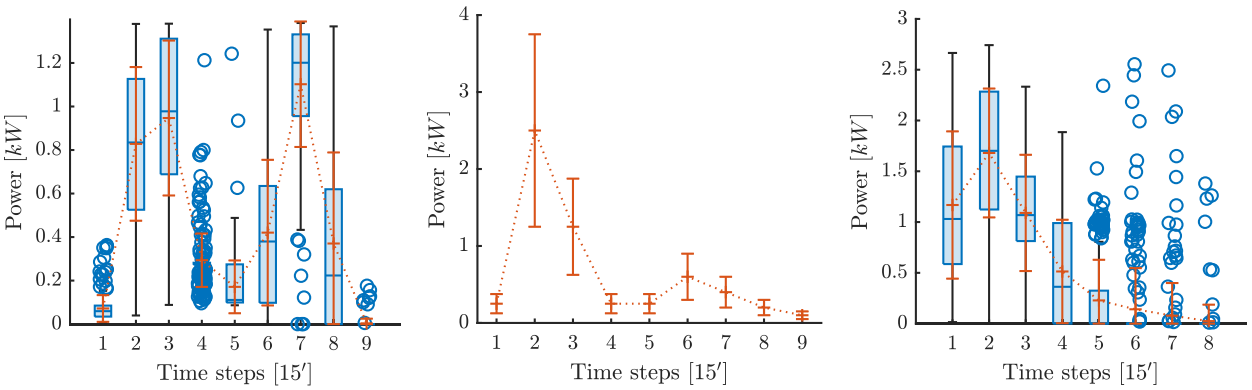
In this Section, we show the simulation results. First, the simulation scenario is described in Section 5.1. The appliances under consideration are introduced, and the user termination constraints are detailed. Then, the simulation results for the cooperative MILP solver and RL agent are shown. In detail, the results of the scheduler are reported in Section 5.2. Such results do not take into account uncertainties and the scheduling algorithm is deterministic. In Section 5.3, instead, we show how the RL agent deals with uncertainty to satisfy the scheduled tasks while maximizing self-consumption.

5.1. Simulation scenario

We consider a HEMS with three shiftable appliances, a lumped non-shiftable load, an adjustable load for air conditioning, and a single wallbox for an EV. The energy consumption of shiftable appliances has been modeled making use of the historical data in [22]. In particular, for each appliance, the mean quarter-hourly consumption is calculated, as well as the quarter-hourly standard deviation of power consumption. Please note that the mean power consumption varies greatly in time, depending on the peculiar work cycle of each appliance. The same analysis is performed for the non-shiftable loads, which are calculated as the difference between the overall power draw and the appliances under consideration, and for the PV power generation. We suppose adjustable loads can assume a finite set of possible consumption values. Finally, we suppose the EV wallbox can be precisely commanded to draw any exact amount of power. Please note this last assumption can be removed by introducing additional discrete variables, at the price of a marginal amount of computational complexity.

Table 2
Mean and standard deviation of shiftable loads power consumption (in kW). Each column represents a quarter-hourly sampling of the working cycles.

	1	2	3	4	5	6	7	8	9
Dishwasher avg	0.0719	0.8282	0.9471	0.2937	0.1712	0.4204	1.1023	0.3704	0.0036
Dishwasher std	0.0613	0.3530	0.3560	0.1227	0.1213	0.3346	0.2883	0.4184	0.0215
Washer avg	0.2500	2.5000	1.2500	0.2500	0.2500	0.6000	0.4000	0.2000	0.1000
Washer std	0.1250	1.2500	0.6250	0.1250	0.1250	0.3000	0.2000	0.1000	0.0500
Dryer avg	1.1679	1.6802	1.0900	0.5134	0.2284	0.1410	0.0783	0.0250	0
Dryer std	0.7250	0.6346	0.5725	0.5095	0.4002	0.4053	0.3213	0.1617	0



(a) Dishwasher. (b) Washing machine. (c) Dryer.

Fig. 3. Shiftable loads: consumption by sample (real data in blue/black, model in red).

5.1.1. Dishwasher

The first appliance ($i = 1$) is a dishwasher. To model possible users' needs, each time the appliance is required to start, we assume the user needs clean dishes before the next meal (before breakfast at 7 am, before lunch at 1 pm, before dinner at 8 pm). The termination time Δt_1 is calculated accordingly. To model possible user behaviors, the user requires the appliance to start once per day, setting $u_1 = 1$ at a random instant (i.e., uniformly distributed) between 6 am and 12 pm. The actual consumption from [22] is shown in Fig. 3(a), by the black and blue box plot. Each blue rectangle shows the first quartile, the median, and the third quartile. The black whiskers show the range of samples in the interquartile range. The blue circles represent possible outliers, i.e., values that are not included between the black whiskers. The error bar in red, instead, represents the Gaussian approximation used in this simulation. In particular, the error bar in Fig. 3(a) is centered on the average consumption, and the total bar length is twice the standard deviation. The average and the standard deviation are also resumed in Table 2.

5.1.2. Washing machine

The second appliance ($i = 2$) is a washing machine. We suppose the user requires the washing machine to end before a fixed time of the day (e.g., 1 pm), and its termination Δt_2 is derived accordingly. User behavior is randomized as in the case of the dishwasher, that is, once per day.

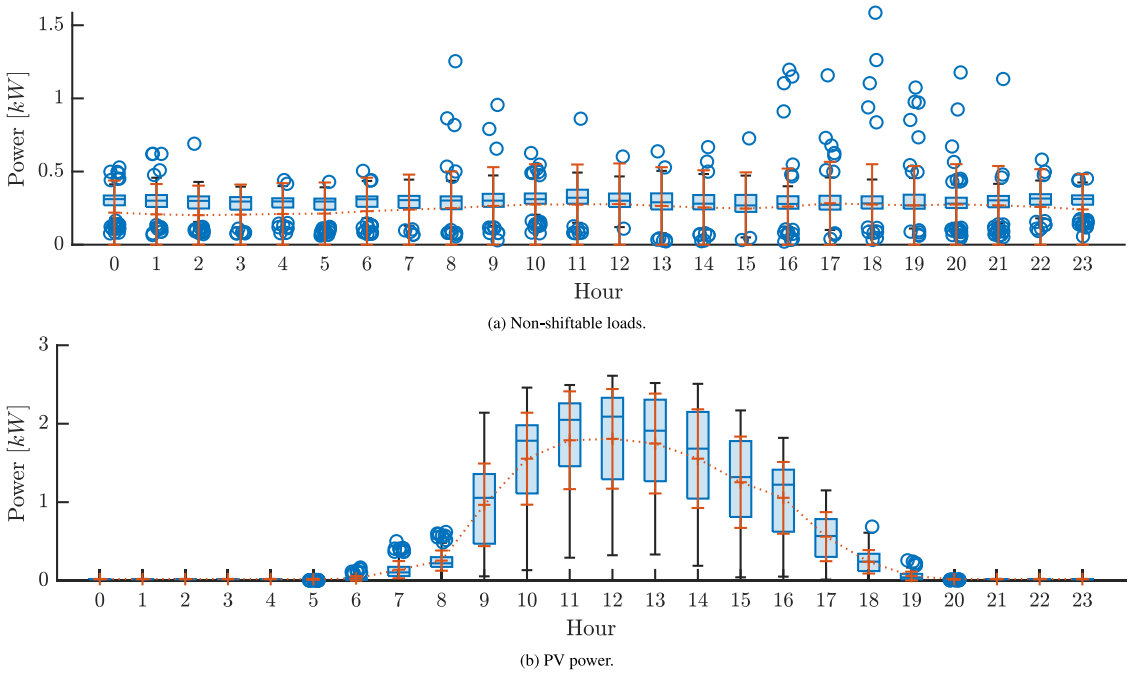
The actual consumption for the washing machine is not available in [22] (apparently, the appliance has never been switched on). Thus, the average values shown in [30] have been employed. The standard deviation has been arbitrarily set to one-half of the average consumption, as shown by the error bars in Fig. 3(b) and by Table 2.

5.1.3. Dryer

The third appliance ($i = 3$) is a dryer. We assume the user wants the dryer to end its task in a fixed amount of hours (e.g., 12 h), hence $\Delta t_3 = 4 \cdot 12$, and the coefficient 4 converts the time from an hourly to a quarter-hourly basis. Again, the user behavior is randomized as in the case of both the dishwasher and the washing machine. The actual consumptions, taken from the dataset in [22], are shown in Fig. 3(c) (black and blue box-plot), and the standard deviations employed for the Gaussian model in the simulations are given by the error bars in red. Again, average and standard deviation are reported in Table 2.

5.1.4. Air conditioner

The fourth appliance ($i = 4$) is an air conditioner. We recall that the user can modify its power level at will, and no prediction of the user behavior is available, thus we suppose the current power level lasts indefinitely in time until another user input occurs. To model user inputs, we assume that the air conditioner is switched on once every day, at a random instant between 6 am and 12 pm. The air conditioner is switched off by the user after a random duration (at least 2 h, up to 24 h). The power level is also randomized, from level 1 (500 W) to level 3 (1500 W). Without loss of generality, the power levels are arbitrarily decided, as the appliance is adjustable.



(a) Non-shiftable loads.

(b) PV power.

Fig. 4. Non-shiftable loads and PV power: hourly consumption or production (real data in blue/black, model in red) in July.

Table 3
Mean and standard deviation of non-shiftable loads power consumption (in kW) and PV power generation (in kW). Each column represents a different hour of the day.

	1	2	3	4	5	6	7	8	9	10	11	12
Loads avg (am)	0.2937	0.2819	0.2718	0.2798	0.2625	0.2894	0.2872	0.2948	0.2996	0.3126	0.3173	0.3018
Loads avg (pm)	0.2857	0.2796	0.2727	0.2967	0.2952	0.3086	0.3063	0.2884	0.2874	0.3073	0.2957	0.2963
Loads std (am)	0.0979	0.0870	0.0781	0.0683	0.0829	0.0803	0.0704	0.1424	0.1171	0.0888	0.0995	0.0827
Loads std (pm)	0.1035	0.0991	0.0963	0.1717	0.1344	0.1969	0.1530	0.1386	0.1159	0.0788	0.0736	0.0879
PV avg (am)	0.0196	0.0198	0.0199	0.0199	0.0169	0.0288	0.1395	0.2552	0.9661	1.5542	1.7889	1.8071
PV avg (pm)	1.7476	1.5545	1.2542	1.0556	0.5602	0.2399	0.0584	0.0178	0.0196	0.0196	0.0192	0.0195
PV std (am)	0.0011	0.0011	0.0011	0.0010	0.0053	0.0301	0.1101	0.1286	0.5265	0.5861	0.6232	0.6349
PV std (pm)	0.6365	0.6288	0.5820	0.4572	0.3127	0.1492	0.0558	0.0041	0.0012	0.0011	0.0013	0.0011

5.1.5. Non-shiftable loads

The fifth appliance ($i = 5$) represents the lumped consumption of non-shiftable and non-interruptible loads. Consumption is randomized according to the dataset in [22], and no user behavior is modeled indeed. The non-shiftable loads from in [22] have been scaled down (by a factor of 3) to fit with the typical Italian domestic contract limitations (3 kW), which is by far lower than the power availability in the original database. Original data (although scaled down) is shown in Fig. 4(a), where the mean and standard deviation of the Gaussian distribution for the simulations are shown in red. Mean and standard deviation are listed in Table 3.

5.1.6. EV wallbox

The sixth appliance ($i = 6$) is the EV wallbox. We assume the charging process must terminate every day at 8 am according to the user's needs, hence the termination time Δt_{wb} is calculated consequently. Even in this case, the scheduler does not have any prior knowledge of the charge starting time. To simulate the user behavior, the EV is connected to the wallbox every day, at a random instant ranging from 12 am to 12 pm. As for the air conditioning, the power levels are arbitrarily decided, because the appliance is adjustable. In particular, we suppose that the wallbox power can assume any positive real value (i.e., without quantization).

5.1.7. PV plant

We consider a small residential PV plant for the simulation. The PV power is taken from the dataset in [22] as well. In particular, a single 400 W PV array in [22] is scaled up by a factor of 6, to have approximately 2.5 kW of maximum power from renewables. The original (and scaled-up) data is shown in Fig. 4(b). Mean and average values (red bars in Fig. 4(b), values in Table 3) are taken into account to perform the simulations.

Table 4
Main parameter values in the simulated scenario.

t_s	h_p	n_s	n_a	n_{ns}	n_{lev}	n_{tot}	l_{cycle}	d	M	M_σ	M_μ	M_u	M_{wb}	M_v
0.25 h	96	3	1	1	4	6	10	$[9 \quad 9 \quad 8]$	20	20	5	10^{-3}	$5 \cdot 10^{-5}$	2
$c_j^0, j \in [33, 76]$		$c_j^0, j \notin [33, 76]$		c_{imm}^0		$c_{station,j}^0, \forall j$	$p_{i,j}^0, i \in [1,3], \forall j$		$p_{4,j}^0, \forall j$		$p_{5,j}^0, \forall j$		e^0	
0.0461975		0.0409725		$c^0/2$		0.1125	See avg in Table 2		0.5		See avg in Table 3		See avg in Table 3	

5.2. MILP simulation

In the following, we present the simulation results for the load scheduling. The simulation scenario is generated under the conditions shown in Section 5.1. The main simulation parameters are reported in Table 4. Please note that all the prices c^0 and c_{imm}^0 refer to the price of 1 kW for a single sample time of length t_s, i.e., 15 min. In other words, the actual price per kWh is four times higher. Similarly, p^0 and e^0 are expressed in kW. In contrast, $c_{station}^0$ is a commercial price per kWh in public charging stations, and acts as a penalty term in the case the battery is not fully recharged.

5.2.1. Single step behavior

To show how the MILP solver behaves, we report an example where the scheduler is stressed by several inputs in the same time sample, while a small maximum power is imposed (the peak shaving limit η is set to 2.5 kW). In particular, we suppose the user requests to activate all of the shiftable loads (dishwasher, washer, dryer) while switching on the air conditioner (minimum power) and also plugging in the EV for recharging. In this example, the EV battery SOC is set to 50% (i.e., $e_{wb} = 28$ kWh), and the current time is 12:15 am. According to the termination constraints in Section 5.1, the shiftable loads (dishwasher, washer, dryer) must terminate at 8 pm of the current day, 12:15 am of the following day, and 00:15 am of the following day, respectively. The EV is going to be unplugged at 08:15 am. Such termination constraints are shown in magenta in Fig. 5.

We note that the shiftable loads are scheduled in a feasible way, meaning that the termination constraints are satisfied: the scheduled start of each appliance is shown in green, and the expected consumption (in blue) is greater than zero in the activation interval accordingly. Please note that the shiftable appliances are delayed by the scheduler to minimize the cost function (22). In particular, the dryer is started immediately, because there is a large amount of PV power to be self-consumed. The washer and the dishwasher are slightly delayed, in order to exploit available PV power while respecting the peak shaving constraint. The required air conditioning power is satisfied as well, and non-shiftable loads cannot be controlled in any way. The power of the wallbox is modulated by the MILP scheduler to comply with the peak shaving constraint. This can be noticed by looking at the overall power consumption, as the power drawn from the grid is often equal to the peak shaving limit η: the power draw from the grid (orange bars in the overall power consumption plot) is initially kept to the maximum value, thus there is no room for additional power draw until 01:45 pm. Also, note that the whole PV energy is self-consumed until 08:00 am the next day. Self-consumption decreases in the following due to the lack of required appliances. The grid power price is also reported. Please note that economic DR is encouraged by the slightly lower unitary prices during the night (calculated according to the Italian authority's current practices): the EV charging is stopped at 5 pm (when the whole PV energy is self-consumed by non-shiftable loads and air conditioning) and it is resumed at 7 pm, to exploit the reduced fee in the late evening.

5.2.2. One month simulation

In the following, we illustrate the results of the MILP scheduler for one month of simulation (i.e., July) where the peak shaving limit η is set to 3 kW. The simulation consists of 2977 executions of the scheduler in a receding horizon framework. The results are compared to the case where load scheduling is not performed. In detail, we define the scheduling-free baseline scenario by modifying the original optimization problem as follows. The variables σ and μ are forced to zero (no tolerance for delays, no tolerance for the adjustable loads), and M_σ and M_μ are eliminated accordingly. M_s is set to zero (no penalty for peak shaving violation). No delay is allowed for shiftable loads, thus any appliance is started as soon as the user requires it (i.e., $u_i = 1 \rightarrow x_{i,1} = 1$). The peak shaving violation s_j is set to $\max(0, \sum_{i=1}^{n_{tot}} p_{i,j} - e_j - \eta)\ \forall j \in [1, \dots, h_p]$ and it is not minimized. The adjustable load power w_j is set rigidly to v. A greedy recharging of the wallbox is performed (i.e., the EV is charged as soon as possible), and no economic DR is performed; we suppose that the wallbox is aware of the available power, thus we have added the constraint

$$p_{6,j} \leq \max(0, e_j + \eta - \sum_{i=1}^{5} p_{i,j})\ \ \forall j = 1, \dots, \Delta t_6 \tag{39}$$

in order to draw as much power as possible.

Performances throughout the month are summarized in Table 5. Thanks to the scheduler, the self-consumed energy is 11.4% higher with respect to the absence of a scheduler. In fact, shiftable loads are delayed to take advantage of PV power and EV vehicle is recharged maximizing self-consumption. Consequently, the energy drawn from the grid is 4.51% lower. Not only less energy is taken from the grid, but also a larger share is taken when the energy price is lower: 88.57% in the presence of the MILP scheduler instead of 78.307%. The scheduler also avoids overloads: the total amount of energy taken overcoming the peak shaving threshold

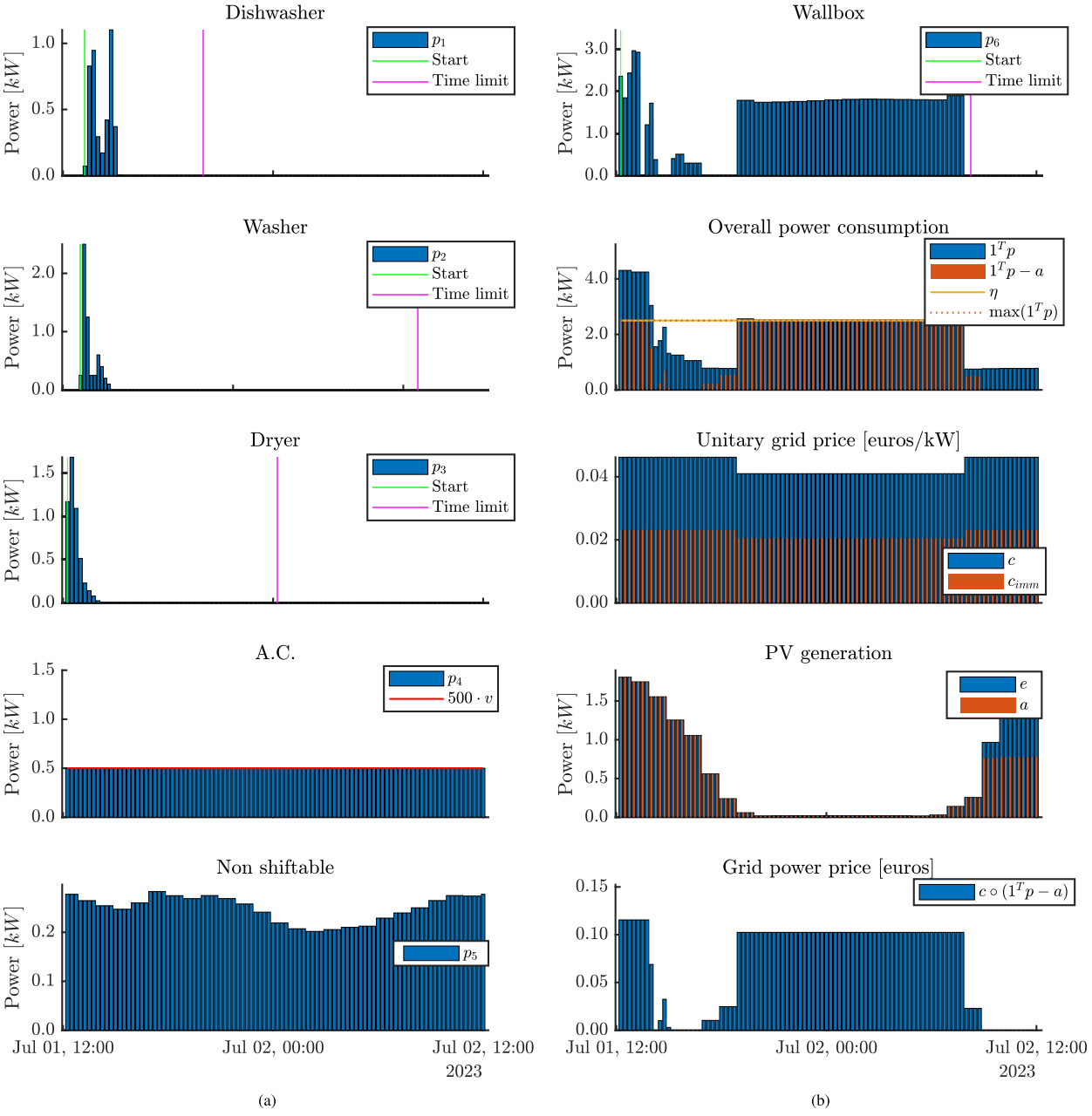


Fig. 5. Detail of a MILP solution.

η is approximately 100 times lower when the scheduler is used. The amount of energy to the EV battery is marginally smaller (4.27%). In fact, the battery is not fully recharged in 5 times out of 30, because the charging time is too short. In this case, when the scheduler is enabled, the delayed shiftable loads can limit the available power once EV is connected; instead, it does not occur when the scheduler is disabled because the shiftable loads start immediately. As the price of energy shows a very small hourly variation, the bill saving, thanks to the scheduler, mainly consists of increased self-consumption. The MILP scheduler entails a larger delay for shiftable loads, i.e., 2.853 additional hours on average; on the other hand, no scheduling failures (i.e., violation of termination constraints or violation of hard constraints) occur. Finally, to avoid overcharges, the adjustable loads are temporarily underpowered for a total of 3 hours (12 samples), in particular, the air conditioning power level is lowered by 1 (i.e., −500 W) for 2.75 h and lowered by 2 (i.e., −1000 W) for 0.25 h. Such behavior occurs when the air conditioning is set to the maximum power (1500 W) and other loads must be powered while respecting the peak shaving constraint.

Table 5
Performances: MILP scheduling compared to no scheduling.

	MILP scheduler	No scheduler
Self-consumed energy [kWh]	267.258	239.867
Energy drawn from the grid [kWh]	728.705	763.101
Energy drawn in the cheapest time slot [%]	88.571	78.307
Energy injected into the grid [kWh]	142.013	169.404
Energy drawn violating peak shaving [kWh]	0.0632	6.6084
Energy to the EV [kWh]	369.431	374.811
Overall bill [€]	108.047	112.872
Average shiftable loads start delay [hours]	3.103	0.250
Adjustable loads marginally diminished [hours]	3.000	0.000
Adjustable loads critically diminished [hours]	0.250	0.000
Scheduling failures [count]	0	0

5.3. RL training and testing

The simulation in Section 5.2 does not take into account any uncertainty in both energy consumption and prediction: the MILP scheduler is deterministic, and each consumption in Fig. 5 corresponds to the mean values shown in Section 5.1. On the other hand, we have already shown that the actual energy consumption is stochastic and its standard deviation is relevant, so the scheduling in Section 5.2 practically becomes infeasible due to the violation of the peak shaving constraint.

We define *scheduling failure* as the undesired event in which scheduled tasks cannot be performed because the energy demand is too high and the BESS cannot compensate for it. More practically, when a scheduling failure occurs, we have to choose between violating the peak shaving constraint and performing load shedding to reduce the required energy.

Thus, we propose to employ a RL agent to manage a BESS to reduce the number of scheduling failures, as well as to enhance self-consumption by adding a delayed self-consumption. We focus on Q-learning, a model-free off-policy RL algorithm based on temporal differences [29], as it resulted to be more effective and easier to train than deep reinforcement learning in such a small-sized problem. Several RL agents, based on Q-learning, have been trained successfully, each one with different hyperparameters. In particular, we have performed a grid search, exploring four Learning Rates (LRs) (10^{-2}, 10^{-3}, 10^{-4}, and 10^{-5}), two penalties M_{rl} in (36) (10^{7} and 10^{12}), and three discount factors γ_{rl} in (38) (1, $e^{-1/96}$, and $e^{-1/48}$, i.e., no discount, 24 h half-life, and 12 h half-life, respectively). As for the other system parameters, the nominal BESS capacity is set to 6 kWh, the maximum charge ($P_{bess,max}$) and discharge rate ($P_{pv,stor,max}$) are set to 3 kW (i.e., 0.5 C), and the battery efficiency ρ is set to 95%. The selection of the battery's capacity is based on the optimal capacity outlined in [31]. Please note that $P_{pv,stor,max}$ is never reached because the PV peak power is less than $P_{pv,stor,max}$. Without loss of generality, the minimum and maximum battery charges are set to $E_{batt,min} = 0$ kWh and $E_{batt,max} = 6$ kWh for the sake of clarity in the results. The environment, the states, the action, the observations, and the reward function are implemented according to Section 4.

The training is performed using a 310 days dataset using a MILP scheduler, so the dataset is analogous to the one in Section 5.2.2. The RL agent runs every 15 min in the simulation (i.e., at the same frequency of the MILP scheduler, because $t_s = \delta\tau = 15$ min), but in the real deployment it can run at high frequency: the evaluation consists in choosing the action according to a small look-up table (i.e., according to the Q-table) and the online learning (i.e., Q-table update) is straightforward as well. The entire dataset, where actual and predicted consumption and PV production have been calculated in advance, represents a single episode in the training of the RL agent. The RL training is performed on 600 episodes, and it is executed on a 16 core i9-11900k CPU running at 5.3 GHz. The training is performed in MATLAB, using the reinforcement learning toolbox. Each training takes approximately 3.5 h. Please note that we set a fixed number of episodes to perform the training, and each episode consists of the same number of steps: as penalties are unavoidable in such an uncertain scenario, we never stop the episode even in case of negative rewards. Stopping the episode, as soon as a penalty occurs, has been investigated, but no agent has been successfully trained under such restrictive conditions.

The performances of the trained agents are resumed in Table 6, where the 24 hyperparameter combinations are considered to train the same amount of RL agents. Each agent has been then tested with a different dataset of 310 days, and the testing results are resumed in Table 6 as well. In detail, in Table 6 we report the energy reward (i.e., the sum of R_{self}, defined in (35)) and the peak shaving penalty (i.e., the sum of R_{peak}, defined in (36)) separately, for the sake of clarity. Moreover, we report the daily scheduling failure rate, i.e., the average number of peak shaving violations per day.

The results in Table 6 are graphically represented in Fig. 6, where the training performances are represented by an empty diamond and the testing performances are represented by a filled triangle. In the x axis, the daily energy saving is shown (the higher, the better). In the y axis, instead, the daily failure rate is reported (the lower, the better). We do not report the penalty (36) in the y axis because the hyperparameter M_{rl} varies, so the quantities are incomparable. Moreover, the objective is to avoid scheduling failures, so the number of failures is more descriptive than the amount of the penalty. Training and testing results using the same agent are connected by a dotted line in Fig. 6: we note that training and testing results are consistent. This is crucial as it means that the RL agents have learned a policy that can be used in real scenarios, which is different from the training dataset. We also note a slight increase in the savings in the test data. Please, remember that both the training set and the test set have been randomly generated according to Section 5.2.2, so this change is justified by slightly more favorable conditions in the testing scenario.

We highlight that the most important hyperparameter is the LR: in Fig. 6, we use different colors for each LR to highlight the presence of clusters. Higher LRs (10^{-2} and 10^{-3}) lead to agents that are sparse in Fig. 6 as their performances are fairly different.

Table 6
RL agents versus trivial and ideal algorithms: hyperparameters, rewards, penalties, and daily failure rate.

Agent			Energy reward		Penalty cost		Daily failure rate	
LR	γ_{rl}	M_{rl}	Train	Test	Train	Test	Train	Test
10^{-2}	1.000	10^{7}	90.829	94.984	$-1.73 \cdot 10^{9}$	$-1.33 \cdot 10^{9}$	3.681	3.174
10^{-2}	0.990	10^{7}	92.763	97.111	$-2.25 \cdot 10^{9}$	$-2.05 \cdot 10^{9}$	5.132	5.261
10^{-2}	0.979	10^{7}	88.313	91.226	$-1.30 \cdot 10^{9}$	$-8.80 \cdot 10^{8}$	2.987	2.203
10^{-2}	1.000	10^{12}	91.816	96.937	$-2.59 \cdot 10^{14}$	$-2.25 \cdot 10^{14}$	5.916	5.816
10^{-2}	0.990	10^{12}	94.379	97.901	$-2.46 \cdot 10^{14}$	$-2.13 \cdot 10^{14}$	5.742	5.671
10^{-2}	0.979	10^{12}	91.567	95.406	$-2.05 \cdot 10^{14}$	$-1.68 \cdot 10^{14}$	4.526	4.174
10^{-3}	1.000	10^{7}	84.581	88.240	$-8.21 \cdot 10^{8}$	$-4.01 \cdot 10^{8}$	1.568	0.945
10^{-3}	0.990	10^{7}	87.268	91.328	$-1.44 \cdot 10^{9}$	$-1.21 \cdot 10^{9}$	3.445	3.123
10^{-3}	0.979	10^{7}	95.291	98.736	$-2.69 \cdot 10^{9}$	$-2.33 \cdot 10^{9}$	6.536	6.432
10^{-3}	1.000	10^{12}	84.439	88.231	$-4.73 \cdot 10^{13}$	$-3.71 \cdot 10^{13}$	1.003	0.906
10^{-3}	0.990	10^{12}	86.765	90.840	$-1.38 \cdot 10^{14}$	$-1.15 \cdot 10^{14}$	3.423	2.826
10^{-3}	0.979	10^{12}	94.539	98.873	$-2.50 \cdot 10^{14}$	$-2.45 \cdot 10^{14}$	6.116	6.358
10^{-4}	1.000	10^{7}	89.147	92.430	$-1.15 \cdot 10^{9}$	$-8.47 \cdot 10^{8}$	2.561	2.052
10^{-4}	0.990	10^{7}	89.056	92.037	$-9.48 \cdot 10^{8}$	$-6.47 \cdot 10^{8}$	1.952	1.471
10^{-4}	0.979	10^{7}	86.663	89.367	$-7.96 \cdot 10^{8}$	$-4.43 \cdot 10^{8}$	1.771	1.058
10^{-4}	1.000	10^{12}	88.016	91.367	$-1.01 \cdot 10^{14}$	$-4.69 \cdot 10^{13}$	1.955	0.987
10^{-4}	0.990	10^{12}	89.028	91.773	$-8.82 \cdot 10^{13}$	$-5.24 \cdot 10^{13}$	1.813	1.155
10^{-4}	0.979	10^{12}	88.271	91.009	$-8.66 \cdot 10^{13}$	$-6.23 \cdot 10^{13}$	1.790	1.371
10^{-5}	1.000	10^{7}	91.332	94.573	$-1.51 \cdot 10^{9}$	$-1.03 \cdot 10^{9}$	2.978	2.281
10^{-5}	0.990	10^{7}	93.870	97.745	$-1.95 \cdot 10^{9}$	$-1.55 \cdot 10^{9}$	3.907	3.374
10^{-5}	0.979	10^{7}	91.972	94.634	$-1.67 \cdot 10^{9}$	$-1.19 \cdot 10^{9}$	3.484	2.755
10^{-5}	1.000	10^{12}	90.289	94.070	$-1.66 \cdot 10^{14}$	$-1.41 \cdot 10^{14}$	3.468	3.323
10^{-5}	0.990	10^{12}	92.831	96.696	$-2.00 \cdot 10^{14}$	$-1.63 \cdot 10^{14}$	4.268	3.894
10^{-5}	0.979	10^{12}	92.878	96.253	$-1.98 \cdot 10^{14}$	$-1.55 \cdot 10^{14}$	3.932	3.378
Conservative		10^{7}	11.476	11.442	$+0.00 \cdot 10^{0}$	$-1.62 \cdot 10^{7}$	0.000	0.003
Greedy		10^{7}	100.867	104.352	$-5.00 \cdot 10^{9}$	$-4.99 \cdot 10^{9}$	11.478	12.136
Ideal		10^{7}	103.735	106.811	$+0.00 \cdot 10^{0}$	$-1.62 \cdot 10^{7}$	0.000	0.003

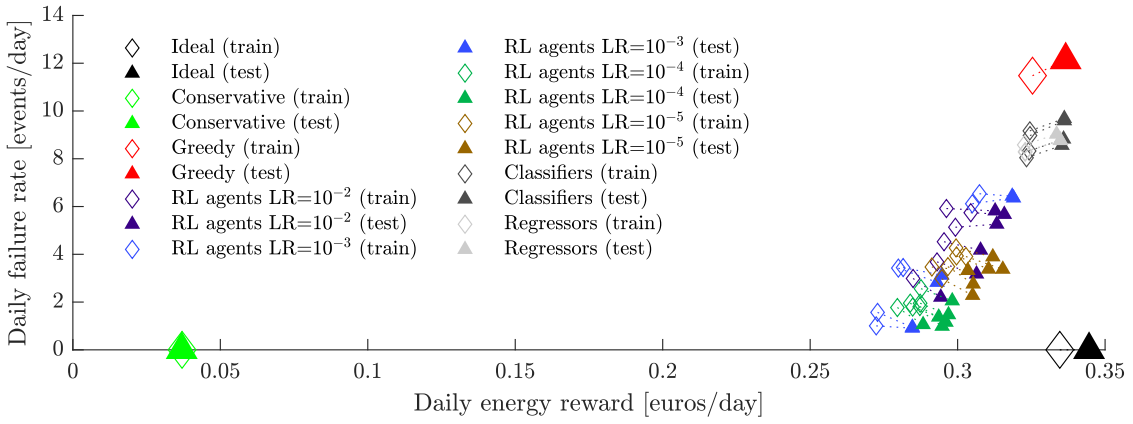


Fig. 6. Reinforcement learning agent performances in terms of daily savings and satisfaction of energy demand.

Conversely, agents with a smaller LR (10^{-4} and 10^{-5}) form two clusters in Fig. 6 because their performances are more consistent. In fact, high LR is not recommended in stochastic settings, as the convergence to a good policy may not happen at all. On the other hand, smaller LRs (10^{-4} and 10^{-5}) may take more time to converge to a good policy: this happens with the smallest LR (10^{-5}), where the scalarized reward (37) is worse than the one given by the LR 10^{-4}, which shows the best reward on average. Thus, the LR 10^{-4} is the most favorable one, in terms of performance and consistency of the training results.

The remaining hyperparameters M_{rl} and γ_{rl}, instead, do not show a significant impact on the final RL agent performances, while we expected a lower failure rate for larger M_{rl}.

The results of the RL agents are compared with a fully conservative agent and a fully greedy agent, i.e., agents that always act as in Eqs. (33) and (34), respectively. The greedy agent is shown in red in Fig. 6, while the conservative one is shown in green. The greedy agent causes the highest daily failure rate, but its energy reward is higher than any other RL agent, as expected. On the contrary, the conservative agent shows the lowest daily failure rate, but its energy reward is much lower than the RL agents we have trained. The trained RL agents show a trade-off between income and scheduling failure rate, which is obtained by mixing the two simple behaviors, i.e., the greedy and the conservative one, according to the learned policy. Such agents show a marginal

reduction of energy reward with respect to the greedy agent, but they drastically reduce the daily failure rate, see for example the dark green agents in Fig. 6 (LR 10^{-4}).

The results of the RL agents are also compared with a non-causal optimizer (*ideal* for brevity), i.e., with an algorithm that knows in advance the actual consumption and user behaviors in the future. Please note that such an ideal optimizer is clearly infeasible in reality, and merely represents an upper bound for the performances of any other strategy.

When complete information about the future is available, the *ideal* algorithm estimates the maximum reward in the case of optimal BESS management. We remark that such information is not available in real applications, as detailed in Section 5.1, because user behavior is random, and consumption and PV power are not deterministic.

The *ideal* algorithm obtains the highest energy reward, outperforming the fully greedy agent (33) as well. This happens because the *ideal* algorithm has full knowledge of the future, so it also takes advantage of the hourly price of electricity. The reward (35) also depends on the current price of the energy $c(\tau)$. On the other hand, the amount of total power delivered by the *ideal* algorithm and the greedy agent is the same.

The *ideal* algorithm shows very low failure rates, as it exploits the exact knowledge of future consumption. In the test set, a single scheduling failure occurs anyway, due to a sequence of peaks in power demand. We note that the fully conservative agent (33) shows the same failure rate of the *ideal* algorithm, but the energy reward is severely affected.

We note that many RL agents, especially the dark green ones in Fig. 6, represent a feasible approximation of the *ideal* behavior. By the way, we remark that the performances of the ideal agent cannot be reached in any way because of the lack of information. The RL agents only know the current SOC, the current time, and the current amount of power required by the user, while the *ideal* algorithm makes use of future consumption predictions with an infinite prediction horizon, and those predictions are not corrupted by uncertainty. Moreover, the information provided to the RL agents is quantized, thus it lacks resolution. On average, training the agents with a LR 10^{-4} gives consistent and repeatable results, while all of the discount factors and penalty terms have led to minor differences.

Fig. 6 also reports the results using different AI-based methods instead of RL agents. We train four classifiers (tree, support vector machine, shallow artificial neural network, and k-nearest neighbors) and three regressors (tree, support vector machine, and shallow artificial neural network). The training involves attempting to replicate the optimal decision made by the *ideal* optimizer using the available inputs, namely current SOC and hour. In particular, the classifiers minimize the misclassification cost function and they are trained to replicate a discrete $\alpha = \{0, 0.25, 0.5, 0.75, 1\}$, as done by the RL agents. Instead, the regressors minimize the mean squared error and return a continuous $\alpha \in [0, 1]$. The classifiers and the regressors are reported in dark and light gray in Fig. 6, respectively. Even employing different algorithms and utilizing various cost functions, they all yield a greedy strategy with a high failure rate.

6. Conclusion

In this study, we have proposed a novel Home Energy Management System by combining Mixed Integer Linear Programming and Reinforcement Learning. Specifically, Reinforcement Learning is used to manage the charge and discharge of a Battery Energy Storage System, while Mixed Integer Linear Programming is employed for load scheduling. The focus of Reinforcement Learning is specifically on Battery Energy Storage System management, as it is a load/generator with stochastic behavior primarily influenced by factors such as photovoltaic production and variations in consumption due to changes in user behavior. The addition of Mixed Integer Linear Programming to Reinforcement Learning has been motivated by the challenge of developing a Home Energy Management System using Reinforcement Learning in scenarios where handling a large state space for system modeling is required. By employing both techniques, we have achieved a dimension reduction of the state space, facilitating Reinforcement Learning training, albeit at the expense of problem optimality due to problem decoupling. The results show that the Reinforcement Learning agent can achieve optimal performance in terms of daily savings and meeting energy demand. The Reinforcement Learning agent can be configured as either a greedy agent or a conservative agent through the setting of the reward function. Depending on the objective of the Home Energy Management System, the agent can maximize self-consumption, minimize energy peaks, or find an optimal trade-off. Since the proposed Home Energy Management System software has been deployed in the cloud, future work involves testing the software in a real environment. Additionally, we plan to scale the software to manage energy aggregation across multiple dwellings in future works.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the Marche region in the implementation of the POR MARCHE FESR 2014–2020, project “Miracle” (Marche Innovation and Research fAcilities for Connected and sustainable Living Environments), CUP B28I19000330007.

References

[1] Mohammadi Mohammad, Noorollahi Younes, Mohammadi-ivatloo Behnam, Hosseinzadeh Mehdi, Yousefi Hossein, Khorasani Sasan Torabzadeh. Optimal management of energy hubs and smart energy hubs–a review. Renew Sustain Energy Rev 2018;89:33–50.
[2] Qureshi Kashif Naseer, Alhudhaif Adi, Hussain Adil, Iqbal Saleem, Jeon Gwanggil. Trust aware energy management system for smart homes appliances. Comput Electr Eng 2022;97:107641.
[3] Baboli P Teimourzadeh, Damavandi M Yazdani, Moghaddam M Parsa, Haghifam MR. A mixed integer modeling of micro energy-hub system. In: 2015 IEEE power & energy society general meeting. IEEE; 2015, p. 1–5.
[4] Setlhaolo Ditiro, Sichilalu Sam, Zhang Jiangfeng. Residential load management in an energy hub with heat pump water heater. Appl Energy 2017;208:551–60.
[5] Sayed Khairy, Gabbar Hossam A. Building energy management systems (BEMS). In: Energy conservation in residential, commercial, and industrial facilities. John Wiley & Sons NJ; 2018, p. 15–81.
[6] Kiran Sharma Vidhu, Mohapatra Srikanta Kumar, Shitharth S, Yonbawi Saud, Yafoz Ayman, Alahmari Sultan. An optimization-based machine learning technique for smart home security using 5G. Comput Electr Eng 2022;104:108434.
[7] Yu Liang, Qin Shuqi, Zhang Meng, Shen Chao, Jiang Tao, Guan Xiaohong. A review of deep reinforcement learning for smart building energy management. IEEE Internet Things J 2021;8(15):12046–63.
[8] Ahrarinouri Mehdi, Rastegar Mohammad, Seifi Ali Reza. Multiagent reinforcement learning for energy management in residential buildings. IEEE Trans Ind Inf 2021;17(1):659–66.
[9] Alfaverh Fayiz, Denaï M, Sun Yichuang. Demand response strategy based on reinforcement learning and fuzzy reasoning for home energy management. IEEE Access 2020;8:39310–21.
[10] Lu Renzhi, Hong Seung Ho, Yu Mengmeng. Demand response for home energy management using reinforcement learning and artificial neural network. IEEE Trans Smart Grid 2019;10(6):6629–39.
[11] Wei Qinglai, Liao Zehua, Shi Guang. Generalized actor-critic learning optimal control in smart home energy management. IEEE Trans Ind Inf 2021;17(10):6614–23.
[12] Tao Yuechuan, Qiu Jing, Lai Shuying, Zhang Xian, Wang Yunqi, Wang Guibin. A human-machine reinforcement learning method for cooperative energy management. IEEE Trans Ind Inf 2022;18(5):2974–85.
[13] Hosseini Seyed Mohsen, Carli Raffaele, Dotoli Mariagrazia. Robust optimal energy management of a residential microgrid under uncertainties on demand and renewable power generation. IEEE Trans Autom Sci Eng 2021;18(2):618–37.
[14] Nie Zelin, Gao Feng, Yan Chao-Bo, Guan Xiaohong. Multi-timescale decision and optimization for HVAC control systems with consistency goals. IEEE Trans Autom Sci Eng 2020;17(1):296–309.
[15] Mason Karl, Grijalva Santiago. A review of reinforcement learning for autonomous building energy management. Comput Electr Eng 2019;78:300–12.
[16] Lissa Paulo, Deane Conor, Schukat Michael, Seri Federico, Keane Marcus, Barrett Enda. Deep reinforcement learning for home energy management system control. Energy AI 2021;3:100043.
[17] Yu Liang, Sun Yi, Xu Zhanbo, Shen Chao, Yue Dong, Jiang Tao, Guan Xiaohong. Multi-agent deep reinforcement learning for HVAC control in commercial buildings. IEEE Trans Smart Grid 2021;12(1):407–19.
[18] Xu Xu, Jia Youwei, Xu Yan, Xu Zhao, Chai Songjian, Lai Chun Sing. A multi-agent reinforcement learning-based data-driven method for home energy management. IEEE Trans Smart Grid 2020;11(4):3201–11.
[19] Lu Renzhi, Bai Ruichang, Luo Zhe, Jiang Junhui, Sun Mingyang, Zhang Hai-Tao. Deep reinforcement learning-based demand response for smart facilities energy management. IEEE Trans Ind Electron 2022;69(8):8554–65.
[20] Ojand Kianoosh, Dagdougui Hanane. Q-learning-based model predictive control for energy management in residential aggregator. IEEE Trans Autom Sci Eng 2022;19(1):70–81.
[21] Alhamed Khalid M, Iwendi Celestine, Dutta Ashit Kumar, Almutairi Badr, Alsaghier Hisham, Almotairi Sultan. Building construction based on video surveillance and deep reinforcement learning using smart grid power system. Comput Electr Eng 2022;103:108273.
[22] Barker Sean, Mishra Aditya, Irwin David, Cecchet Emmanuel, Shenoy Prashant, Albrecht Jeannie, et al. Smart*: An open data set and tools for enabling research in sustainable homes. SustKDD, August 2012;111(112):108.
[23] Longe Omowunmi Mary, Ouahada Khmaies, Rimer Suvendi, Harutyunyan Ashot N, Ferreira Hendrik C. Distributed demand side management with battery storage for smart home energy scheduling. Sustainability 2017;9(1):120.
[24] Baghdadi Issam, Briat Olivier, Delétage Jean-Yves, Gyan Philippe, Vinassa Jean-Michel. Lithium battery aging model based on dakin's degradation approach. J Power Sources 2016;325:273–85.
[25] Cardoso Gonçalo, Brouhard Thomas, DeForest Nicholas, Wang Dai, Heleno Miguel, Kotzur Leander. Battery aging in multi-energy microgrid design using mixed integer linear programming. Appl Energy 2018;231:1059–69.
[26] The MathWorks Inc. MATLAB version: 9.10.0.1602886 (R2021a). 2022.
[27] Löfberg J. YALMIP : A toolbox for modeling and optimization in MATLAB. In: Proceedings of the CACSD conference. Taipei, Taiwan; 2004, p. 284–9.
[28] MOSEK ApS. The optimizer for mixed-integer problems. 2019.
[29] Sutton Richard S, Barto Andrew G. Reinforcement learning: An introduction. MIT Press; 2018.
[30] Lezama Fernando, Soares Joao, Canizes Bruno, Vale Zita. Flexibility management model of home appliances to support DSO requests in smart grids. Sustainable Cities Soc 2020;55:102048.
[31] Koko Sandile Phillip. Optimal battery sizing for a grid-tied solar photovoltaic system supplying a residential load: A case study under South African solar irradiance. Energy Rep 2022;8:410–8.

Riccardo Felicetti received his M.Sc. cum laude in Computer and Automation Engineering in 2016 and the Ph.D. cum laude in Information Engineering in 2021 from Università Politecnica delle Marche, Italy, where he is currently a research fellow. His main research interests are fault detection and diagnosis, fault tolerant control, and optimization with applications to vehicles and energy management systems.

Francesco Ferracuti received his Ph.D. in automation, information and management engineering from Università Politecnica delle Marche, Ancona, Italy, in 2014. He is a researcher at Università Politecnica delle Marche. His research interests include model-based and data-driven fault diagnosis, system identification, signal processing, statistical pattern recognition, and their applications in industry.

Sabrina Iarlori is a researcher at the Information Engineering Department of Polytechnic University of Marche, Italy, where she graduated in biomedical engineering (with honors) and obtained her Ph.D. in computer engineering, management and automation. She obtained a M.Sc. in industry 4.0 design at University of Pisa. Her research interests concern human–robot interaction and collaboration in assistive robotics.

Andrea Monteriù received his M.Sc. cum laude in electronic engineering and a Ph.D. from Università Politecnica delle Marche, Italy, in 2003 and 2006, respectively. Currently, he is an associate professor at Università Politecnica delle Marche. His research interests mainly focus on the areas of fault diagnosis and fault tolerant control applied on robotic, and unmanned and artificial intelligent systems.