Knowledge-Based Systems 280 (2023) 111027

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Solving an energy resource management problem with a novel multi-objective evolutionary reinforcement learning method

G.M.C. Leite [a,b,*], S. Jiménez-Fernández [b,*], S. Salcedo-Sanz [b], C.G. Marcelino [c], C.E. Pedreira [a]

[a] *Systems and Computing Department, Federal University of Rio de Janeiro (COPPE - UFRJ), Rio de Janeiro, 21941-909, Brazil*
[b] *Department of Signal Processing and Communications, Universidad de Alcalá (UAH), Alcalá de Henares, 28801, Spain*
[c] *Institute of Computing, Federal University of Rio de Janeiro (UFRJ), Rio de Janeiro, 21941-901, Brazil*

ARTICLE INFO

Keywords:
Multi-objective reinforcement learning
Policy search
Neuroevolution
Microgrids
Energy management

ABSTRACT

Microgrids have become popular candidates for integrating diverse energy sources into the power grid as means of reducing fossil fuel usage. Energy Resource Management (ERM) is a type of Unit Commitment problem, where a player operates a microgrid with diverse renewable generators integrated with an external supplier. Calculating the economic dispatch of each committed unit on a planning horizon is an NP-hard problem, and therefore, finding an exact solution is difficult. This paper presents a multi-objective solution to the ERM problem from the perspective of battery operation and external supplier dispatch. First, a novel multi-objective decision problem modeling is proposed that considers three objectives: cost, greenhouse gas emissions, and battery degradation. This framework involves a learning agent that controls the depth of discharge of a Lithium-Ion battery. To address the proposed problem, a new multi-objective algorithm called Multi-Objective Evolutionary Policy Search (MEPS) is introduced. The proposed algorithm uses NeuroEvolution of Augmenting Topologies structure to evolve artificial neural networks for estimating action-preference values considering multi-objective rewards. The MEPS performance is evaluated on both standard and newly-proposed benchmark problems, using the hypervolume as the evaluation metric. When compared to standard deep reinforcement learning, results showed that MEPS provides cost-effective, environmentally friendly, and efficient energy storage management solutions. Furthermore, MEPS effectively solves the proposed ERM problem by finding neural networks with a small number of nodes and connections, which are suitable for use in embedded control systems. Overall, MEPS proved to be a promising multi-objective approach in the transition to clean energy resources.

1. Introduction

The rise in atmospheric greenhouse gas (GHG) concentrations and the impacts of climate change have made the use of renewable energy sources (RESs) a global priority for both humanity and industry [1–3]. During the 2010–2020 decade, solar and wind technologies have faced a cost reduction that increased their deployment in energy generation to the detriment of fuel-based energy generation [4]. Furthermore, the decarbonization of transport and mobility services is also considered a key alternative for reducing GHG emissions [5]. Accordingly, electric vehicles are being promoted for helping to achieve a transition towards renewable energies in this sector and for reducing the corresponding environmental impacts of vehicles based on fossil fuels [6].

Microgrids (MG) have become popular candidates for integrating distributed energy resources (DERs) into the power grid aiming the goal of zero carbon emissions. The microgrid concept comprises a low-voltage distribution system that integrates distributed energy resources, such as microturbines, photovoltaic panels, and electric vehicles, with storage and flexible loads. Microgrid systems can operate either autonomously or non-autonomously by interconnecting to the public grid [7,8].

A very important task in operating an MG involves determining the optimal Unit Commitment (UC), taking into account technical and economic constraints over a long planning horizon, of up to one year. UC refers to the problem of determining the schedule of generating units within a power system, with the goal of minimizing costs while satisfying system constraints [9]. The UC module of an MG controls not only the committed generators and power imported from the public grid, but also the power exported to/imported from ESS units [10].

* Corresponding authors at: Department of Signal Processing and Communications, Universidad de Alcalá (UAH), Alcalá de Henares, 28801, Spain.
E-mail addresses: gmatos@cos.ufrj.br, gabriel.matos@uah.es (G.M.C. Leite), silvia.jimenez@uah.es (S. Jiménez-Fernández), sancho.salcedo@uah.es (S. Salcedo-Sanz), carolina@ic.ufrj.br (C.G. Marcelino), pedreira@cos.ufrj.br (C.E. Pedreira).

https://doi.org/10.1016/j.knosys.2023.111027
Received 11 July 2023; Received in revised form 6 September 2023; Accepted 21 September 2023
Available online 26 September 2023
0950-7051/© 2023 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Therefore, UC in MGs with ESS units is a complex constrained optimization problem, due to the climate-dependent nature of renewable resources such as solar and wind, and the fluctuating market price. In this way, the Energy Resource Management (ERM) of an MG can be considered as a type of UC problem, where a player operates an MG with diverse renewable generators in integration with an external supplier [11]. An efficient ERM solution leads to not only a profitable but also a sustainable and reliable operation of the MG [7]. Thus, controlling the dispatch of generation units is one of the most important optimization problems for daily operation scheduling and planning. Moreover, according to the International Electrotechnical Commission in the standard IEC 61970, the computer system that assures the effective management operation of a microgrid is called Energy Management System (EMS) [12]. Consequently, the EMS of a microgrid encompasses both supply-side and demand-side management, while ensuring that system constraints are respected.

EMSs are usually divided into centralized and decentralized according to their mode of operation. In centralized mode, the management system is located in a central station and connected to the distributed energy resources (DERs) via communication lines for control and data exchange. On the other hand, in the decentralized mode, each DER operates independently and manages itself using a local controller, eliminating the need for communication. In this paper, the ERM control of a centralized energy management system is modeled as a sequential decision problem and treated from a multi-objective perspective regarding operating cost, greenhouse gas emissions, and battery degradation.

Many classical and heuristic optimization algorithms have been proposed to handle energy management problems [13–16]. However, the ERM problem can also be interpreted as a sequential decision problem with Markov decision process (MDP) properties [17]. In such modeling, reinforcement learning (RL) approaches have been proposed to solve the resulting ERM problems. For instance, Mannion et al. [18] employed a multi-agent RL modeling using Q-Learning for controlling the generating units, considering both cost and pollutant emissions. In this model, different functions that combine cost, emission values, and a constraint violation penalty into a single reward signal are evaluated. In [19], a near-optimal ESS operation strategy using state–action–reward–state–action (SARSA) algorithm is presented. The operation strategy is then employed to manage uncertainties in forecasting wind power generation. Wind power uncertainty is also tackled in [20], in which a two-fold solution is proposed. First, a long-short term memory (LSTM) model is used to predict wind power. Then, an ERM problem for wind power control is modeled and solved as a sequential decision problem using deep Q-learning. In the proposed modeling, the learning agent is responsible for finding the optimal charge/discharge decision-making strategy for the ESS present in the MG.

Regarding the life span and maintenance of ESSs, Wu et al. [21] proposed a deep RL method that combines Monte Carlo Tree Search (MCTS) and state–action estimation using deep neural networks for preventive maintenance in a set of batteries. A combination of Q-learning and MCTS is also presented in [22], in which the learning agent handles the dispatching of the ESS in the microgrid using a multiperiod stochastic model and considering the battery's degradation cost. The management of a community battery ESS (CBESS) through Q-Learning is presented in [23]. The presented solution employs the cost of charging and discharging the CBESS plus a penalty for exceeding maximum and minimum SoC limits.

Within an EMS, some recent works target the problem of demand response (DR), in which the energy consumption can be shifted over time, and generated energy is stored to be used in high-demand periods, providing more flexibility to the grid. For instance, in [24], a Q-Learning algorithm is employed to control a service provider's energy retail price. To achieve the objective of maximizing the profit obtained by the service provider while minimizing the cost for customers, the reward is a function of the energy demand prior to the new retail price and the actual energy consumption after setting the price. In [25], a Deep Q-Learning-based solution is employed in a home energy management system (HEMS) to provide not only control but also demand response by shifting load. The goal is to minimize cost and penalties for daily actions and battery usage while maximizing user's satisfaction and consumption of PV energy. A Deep Q-Learning-based solution for a HEMS is also presented in [26], in which the objective is to minimize both the expected energy cost and the difference between the daily consumption and PV generation.

The role of an EMS also includes guaranteeing security by defending against cyber-attacks, and avoiding peaks of load demand by employing demand side management (DMS). A priority deep Q-learning (PDQN) algorithm is employed by [27] to perform DMS and minimize costs and load peaks in attending load demand. To achieve these objectives, the reward for reducing load peak is decomposed into four components that explore properties of the differences in load height between time steps. Regarding security, a three-module EMS is proposed by [28]. The first module detects anomalies by classifying suppliers and predicts the real energy supply, the second module is responsible for the system operation, and the third is composed of a market call-auction system. Moreover, the second module employs a Q-Learning algorithm that is responsible for acting towards maximizing the ratio between the reserved energy and the total energy produced. These studies, summarized along with their limitations in Table 1, have provided an important reference for this work.

According to the above literature review and data summarized in Table 1, the following research gaps have been detected:

1. Based on the presented studies, the problem of ERM control has not yet been approached from a multi-objective perspective;
2. Despite the work presented in [18], there has been no particular focus on greenhouse gas emissions in the majority of these studies that deal with RES;
3. Not all studies have considered both wind and solar generation;
4. Although battery degradation cost is considered in some works, it is not explicitly calculated and considered as an objective;
5. If a weighted sum of objectives is utilized to solve a multi-objective problem, it is not possible to discover multiple trade-offs among objectives in one run, and the derived solutions are constrained to the convex regions of the Pareto front [29];

In order to provide multiple trade-offs among objectives, multi-policy multi-objective reinforcement learning (MORL) algorithms have emerged as a promising alternative to finding high-quality trade-off solutions. For example, in [30], an MORL algorithm, that couples a short term-planning approach of parameterizing a reward function estimator with a long-term planning based on actor-critic. Song et al. [31] presented a multi-objective version of proximal policy optimization (PPO) for a multi-objective trajectory control and task offloading (TCTO) problem. This problem consists of flying an unmanned aerial vehicle along a planned trajectory to collect computation tasks from smart devices. In [32], an algorithm that combines Fuzzy Coding and Actor-Critic is proposed to solve a multi-objective robotic visual control problem.

Despite the existence of MORL proposals based on policy search in the state-of-the-art, they often rely on policy gradient updates, and they are rarely employed in EMS solutions. Furthermore, the use of deep learning models is frequent in ERM, RL [33], and time-series processing [34]. However, deep learning model performances are highly dependent on the architecture/topology and, therefore, require a lot of domain knowledge and human intervention [35]. Motivated by this and based on the research gaps mentioned above, our proposal aims at exploiting not only the strong generalization ability of RL and the advantages of gradient-free optimization but also the search performance of Evolutionary Algorithms. Therefore, the main contributions of this work are:

Table 1
Investigation of RL techniques applied to ERM problems.

Refs.	Generation sources	RL Method	Objectives	Limitations
[18]	Fossil-fueled Generators	Q-Learning	Minimize the operational cost and total pollutant emissions	Only considers fossil-fueled generators and the two objectives are converted into one objective that represents the weighted sum.
[19]	Wind	SARSA	Minimize the difference between the error in forecasted Wind Power and ESS energy	Does not consider photovoltaic panels and only one objective is handled
[20]	Wind	Deep Q-Learning	Minimize the costs of buying energy and ESS operation, penalties for ESS constraints, penalties for optimal SoC, and discarded wind energy	Does not consider photovoltaic panels and the five objectives are converted into one objective that represents the weighted sum.
[21]	Fleet of batteries	MCTS and Deep Learning (ResNet)	Maximize the reliability of the entire battery system over time	Does not consider neither wind nor photovoltaic generation. Only one objective is considered.
[22]	Wind and Solar	MCTS and Q-Learning	Minimize the degradation cost of ESS, the penalty cost for exceeding the maximum power sent back to public grid, and maximize the revenue generated from power sent back to public grid	The reward is a scalar corresponding to the weighted sum of the three objectives.
[23]	Wind and Solar	Double Deep Q-Learning	Minimize the costs of ESS operation and penalties for operating over the SoC limits	The reward is a scalar value.
[24]	Public grid	Q-Learning	Maximize the profit obtained by the service provider and minimize the cost for customers	The reward is a scalar obtained by a convex combination between both objectives. Neither wind nor solar energy generation is considered. Tabular Q-learning suffers from the curse of dimensionality.
[25]	Solar	Deep Q-Learning and Double Deep Q-Learning	Minimize cost and penalties for daily actions and battery usage, and maximize user's satisfaction and consumption of PV energy	The reward is a scalar obtained by the sum of 5 different objectives. Wind energy is not considered.
[26]	Solar	Deep Q-Learning	Minimize cost of energy and the difference between the daily consumption and PV generation	The reward is sum of two objectives. Solar energy is not considered.
[27]	Wind and Solar	Priority Deep Q-Learning	Minimize cost of energy and load peaks	The reward is a scalar corresponding to the sum of 5 objectives.
[28]	Wind and Solar	Q-Learning	Maximize the ratio between the reserved energy and the total energy produced	The reward is a scalar value. Tabular Q-learning suffers from the curse of dimensionality.

1. A novel multi-objective neuroevolutionary reinforcement learning algorithm for deterministic multi-policy search, that does not depend on policy gradient updates, is proposed.
2. A diversity selection mechanism that uses a heavy tail distribution to restrict the number of individuals selected at each iteration is proposed. This mechanism applied to evolving neural networks, allows some newly introduced topologies to have time to improve.
3. A new MORL benchmark environment based on the ZDT 3 multi-objective optimization benchmark function is proposed.
4. An extensive comparison and statistical analysis between the proposed method and other MORL algorithms present in the literature is presented.
5. A new multi-objective energy management problem that accounts for the amount of CO_2 emissions, integrates a non-linear degradation model for lithium-ion batteries, and uses real-world load time-series data is presented.

The remainder of the paper is as follows: Section 2 provides the necessary definitions regarding RL, and Section 3 presents the proposed ERM control problem modeling. After, Section 4 describes the proposed algorithm. The benchmark analysis and experimental results are presented in Section 5. The conclusions are presented in Section 6.

2. Reinforcement learning background

In reinforcement learning (RL), the learning agent deals with the problem of making decisions in unknown, possibly dynamic environments. In the standard single-objective case, the overall target of the agent is to learn a sequence of decisions that maximizes the expected value of a scalar feedback signal. Essentially, these decisions relate to action selection in certain environmental states. A typical formalization of a reinforcement learning environment by means of Markov decision process (MDP) is a tuple (S, A, T, R) [36], in which:

- $S = \{s_1, \dots, s_N\}$ denotes the state space,
- $A = \{a_1, \dots, a_r\}$ denotes the finite set of available actions,
- $T(s'|s, a) \in [0, 1]$ is a transition function that specifies, for each state, action, and next state, the probability of that next state occurring given an action at the current state, and
- $R(s, a) : S \times A \to \mathbb{R}$ is a reward function that specifies, for each pair state–action, the expected immediate reward.

The goal of an agent is to learn a policy π that maps each state to an action so that the expected return received in the long run is maximized [37]. In this work, only deterministic transitions are considered, hence $T(s'|s, a) = 1$. The state-dependent value function of a policy π in a state s is defined as

$$V^\pi(s) = \mathbf{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s \right], \tag{1}$$

in which r_t is the reward obtained at time t and $\gamma \in [0, 1]$ is the discount factor. In a finite horizon model, the state value function becomes

$$V^\pi(s) = \mathbf{E}_\pi \left[\sum_{k=0}^{h} r_{t+k} | s_t = s \right], \tag{2}$$

in which h denotes the length of the horizon. The expected return from starting at state s, taking action a, and following policy π is given by a $Q^\pi(s, a)$-value. For an infinite horizon model, it is expressed as:

$$Q^\pi(s, a) = \mathbf{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, a_t = a \right]. \tag{3}$$

The optimal Q^*-values are defined as

$$Q^*(s, a) = R(s, a) + \mathbf{E} \left[\gamma \max_{a'} Q^*(s', a') \right]. \tag{4}$$

The Q-Learning algorithm presented in [38] provides a way to iteratively approximate Q^*. In Q-Learning, each state–action pair is stored in a Q-table and, with learning rate $\alpha \in (0,1]$, updated incrementally based on feedback and Temporal Difference learning (TD) [39] according to the rule

$$\hat{Q}(s,a) = \hat{Q}(s,a) + \alpha_t \left(R(s,a) + \gamma \max_{a'} \hat{Q}(s',a') - \hat{Q}(s,a) \right). \quad (5)$$

Assuming that all state–action pairs are visited and updated under Q-Learning, the $\hat{Q}$ estimates converge to the optimal values Q^* in either deterministic or non-deterministic MDPs [40,41].

By selecting the first action according to the policy π, the Q-function is equivalent to the value function and can be written as

$$V^{\pi}(s) = \mathbf{E}_{\pi} \left[Q^{\pi}(s_t, \pi(s_t)) | s_t = s \right]. \quad (6)$$

Finding a policy that maximizes Eq. (6) requires searching over a function space, which is generally an intractable problem. Therefore, policies are parameterized by some $\theta \in \Theta \subset \mathbb{R}^d$ so that search is performed in a Euclidean space of finite dimension. Consequently, the resulting problem becomes

$$\max_{\theta} \mathbf{E}_{\pi_\theta} \left[Q^{\pi_\theta}(s_t, \pi_\theta(s_t)) | s_t = s \right] = J(\theta). \quad (7)$$

In this work, policies are parameterized using neural networks.

2.1. Multi-objective reinforcement learning

In multi-objective reinforcement learning (MORL), the reward objective space consists of two or more dimensions that must be optimized simultaneously [42,43]. Hence, scalar reward values in MDPs are translated into reward vectors $\mathbf{R}(s,a) \in \mathbb{R}^m$. The variable m stands for the number of objectives and the ith component of the reward vector denotes the reward obtained for the ith objective. Since the agent will optimize several objectives simultaneously, there will be different optimal policies with respect to distinct goals. Therefore, the optimality criteria used is, in general, the concept of Pareto dominance [44].

Generally, to establish an ordering among solutions in MORL problems, two solutions are compared according to the Pareto dominance relation [45].

Definition 1. Given two policies $\pi_1, \pi_2 \in \Pi$, it is said that policy π_1 strictly dominates policy π_2, denoted by $\pi_1 \succ \pi_2$, if

$$\forall i \in \{1,\dots,m\}, \mathbf{V}_i^{\pi_1}(s) \le \mathbf{V}_i^{\pi_2}(s) \land \exists j \in \{1,\dots,m\}, \mathbf{V}_j^{\pi_1}(s) < \mathbf{V}_j^{\pi_2}(s) \quad (8)$$

$\mathbf{V}^{\pi_1}(s)$ strictly improves $\mathbf{V}^{\pi_2}(s)$ in at least one objective and $\mathbf{V}^{\pi_1}(s)$ is at least equal to $\mathbf{V}^{\pi_2}(s)$ in all other objectives.

Definition 2. Given two policies $\pi_1, \pi_2 \in \Pi$, it is said that policy π_1 is incomparable to policy π_2 if

$$\exists i \in \{1,\dots,m\}, \mathbf{V}_i^{\pi_1}(s) < \mathbf{V}_i^{\pi_2}(s) \land \exists j \in \{1,\dots,m\}, \mathbf{V}_j^{\pi_2}(s) < \mathbf{V}_j^{\pi_1}(s). \quad (9)$$

$\mathbf{V}^{\pi_1}(s)$ strictly improves $\mathbf{V}^{\pi_2}(s)$ in at least one objective and $\mathbf{V}^{\pi_2}(s)$ strictly improves $\mathbf{V}^{\pi_1}(s)$ in at least one objective. Therefore, both policies are incomparable.

Definition 3. A policy $\pi_i^* \in \Pi$ is said to be Pareto optimal iff it is non-dominated by any other policy π_j:

$$\pi_i^* \in \Pi \leftrightarrow \nexists \pi_j \in \Pi : \pi_j \succ \pi_i^*. \quad (10)$$

The set $\Pi^* \subset \Pi$ with all Pareto optimal policies is named Pareto front [44].

3. ERM problem formulation

In this section, we formally define the Energy Resource Management problem for a planning horizon of one year. First, we present the microgrid configuration followed by the objective functions and constraints definitions. Then, the ERM control problem is modeled as a Markov Decision Process (MDP) and treated from a multi-objective perspective regarding operating cost, greenhouse gas emissions, and battery degradation. In this resulting control problem, the learning agent plays the role of an EMS that is responsible for managing the maximum allowed monthly ESS' depth of discharge (DoD), and the energy imported from the public grid.

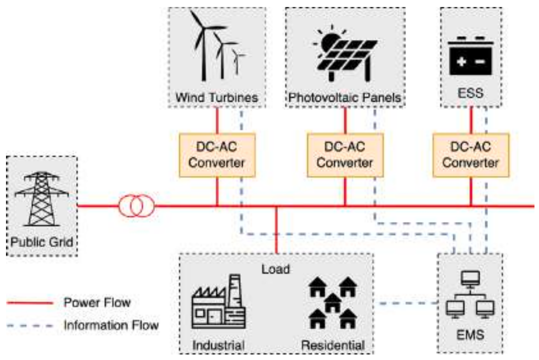


Fig. 1. Solar wind power microgrid system structure.
Source: Based on [20].

Table 2
General information about the microgrid.

	Unit	INV	PV	WT	LTO battery
Life time (years)	Years	15	24	24	17.5
Efficiency (%)	%	96	20.4	95	90
Rated power	kW	–	0.45	100	–
Capacity	kW	–	–	–	1000
Cycles	un	–	–	–	8000
Initial cost	$	–	500.00	1800.00	–
Cost	$/kW	700.00	–	–	1143.00
Operational cost	$/kW	–	18	0.36	–

3.1. Microgrid system model

The structure of the simulated solar wind power microgrid system is based on [46], which also details the mathematical modeling concerning the components functioning. The simulated microgrid is composed of six Norvento nED 100-22 wind turbines,[1] 5000 HiKu 450W-CS3W-450MS photovoltaic panels,[2] an ESS, DC/AC converter, electrical load that comprises both residences and industrial buildings, main grid connection (the main grid price mechanism employed is real-time pricing (RTP)), and EMS. Fig. 1 presents the system structure. Moreover, the ESS consists in a 1000 kW capacity spinel lithium titanate ($Li4Ti5O_{12}$ (LTO) [47]) battery. Table 2 details the configuration values for the MG project with a lifetime of 24 years.

With respect to the data used, a generic region from Cadiz, in Southern Spain, was used as case a study of this work. In this regard, an annual load profile (measured in 8640 h) comprising both a residential community and an industrial consumption is analyzed. Besides, hourly data for solar radiation, wind speed, ambient temperature, and dynamic energy price from 2021 serve as input to the MG system.

Next, the objective functions and the different constraints involved in the problem are introduced.

3.2. Operational costs objective and ESS constraints

The charge/discharge control of the ESS represents the use or storage of energy from the ESS. At each time step t, the ESS can be either

[1] https://www.norvento.com/productos/aerogeneradores-de-media-potencia/.
[2] https://www.csisolar.com/au/hiku/.

charging or discharging. The minimum state-of-charge (SoC) value is defined as the $1 - DoD(t)$ of ESS. Furthermore, the SoC at time step t is given by [20]:

$$SoC(t) = \begin{cases} SoC(t-1) + \frac{E_{bat}(t)\cdot\eta_c}{E_{rated}}, & E_{bat}(t) \geq 0 \\ SoC(t-1) + \frac{E_{bat}(t)}{E_{rated}\cdot\eta_d}, & E_{bat}(t) < 0, \end{cases} \tag{11}$$

in which $\eta_c = 0.90$ and $\eta_d = 1.0$ are the charging and discharging efficiencies, respectively. E_{rated} denotes the rated capacity of the ESS. The amount of used or stored energy from the ESS, $E_{bat}(t)$, is calculated as

$$E_{bat}(t) = \begin{cases} E_{dch}(t), & \text{if discharging} \\ E_{ch}(t), & \text{if charging,} \end{cases} \tag{12}$$

in which $E_{dch}(t)$ and $E_{ch}(t)$ are the discharging and charging requested energy amount given by [46]:

$$E_{dch}(t) = \begin{cases} \max\{(1 - DoD(t) - SoC(t-1)) & \\ \quad \cdot E_{rated}, E_{dch}(t)\}, & SoC(t-1) > 1 - DoD(t) \\ 0, & SoC(t-1) \leq 1 - DoD(t), \end{cases} \tag{13}$$

$$E_{ch}(t) = \min\{(SoC_{max} - SoC(t-1)) \cdot E_{rated}, E_{ch}(t)\}. \tag{14}$$

In addition, the use and degradation costs of ESS are considered in the operation. The ESS use cost coefficient is given by [46,48]:

$$c_d(t) = \frac{C_i}{L_c \cdot E_{rated} \cdot DoD(t)}, \tag{15}$$

in which L_c is the available cycle lifetime and C_i is the initial investment of ESS. Therefore, in this modeling the total cost of an operation is obtained by [48]:

$$C_{total}(t_{start}, t_{end}) = IC + PW_p + PW_{np} + \sum_{t=t_{start}}^{t_{end}} c_d(t), \tag{16}$$

in which the value $C_{total}(t_{start}, t_{end})$ represents the sum of the system costs of operating from time t_{start} to t_{end}. The initial cost (IC) refers to the 20% cost for operation & maintenance, 6% discount rate, 1.4% inflation rate, personnel cost, installation, and connections. There is also included both the periodic costs PW_p of components maintenance, such as PV panels and wind generators, and the non-recurrent costs PW_{np} of components replacement, such as ESS [48]. Moreover, the charging/discharging power constraints are given by

$$\begin{cases} 0 \leq E_{ch} \leq E_{ch}^{max}, \\ 0 \leq E_{dch} \leq E_{dch}^{max}, \\ E_{ch} \cdot E_{dch} = 0, \end{cases} \tag{17}$$

in which E_{ch}^{max} and E_{dch}^{max} are the maximum charge and discharge energy, respectively. Additionally, the ESS' SoC should be maintained at a suitable range at each time step

$$SoC_{min}(t) \leq SoC(t) \leq SoC_{max}, \tag{18}$$

with $SoC_{min}(t)$ denoting the minimum SoC value at time step t and SoC_{max} as the maximum allowed SoC. Finally, the resulting objective function for cost minimization, modeled by [46], is

$$\min Cost(t_{start}, t_{end}) = C_{total}(t_{start}, t_{end}) + \sum_{t=t_{start}}^{t_{end}} P_t^{buy} \cdot Pr_t, \tag{19}$$

where the operational cost is combined with the cost of the energy bought from the public grid P_t^{buy} at price Pr_t to supply the insufficient microgrid power.

3.3. Microgrid CO_2 emissions objective

In order to take into consideration an estimation of the amount of CO_2 emissions for energy bought from the public grid and renewable generation, we have used the greenhouse gas emission values as an average of the minimum and maximum values from [49] in grams of CO_2eq./KWh for each energy source. Hence, the following values have been used:

- Solar Photovoltaic: 44.15 g CO_2eq./KWh;
- Wind Power: 11.90 g CO_2eq./KWh;
- Nuclear: 5.75 g CO_2eq./KWh;
- Hydro: 76.50 g CO_2eq./KWh;
- Cogeneration and Combined cycle: 156.00 g CO_2eq./KWh;

Moreover, with respect to energy bought from the public grid, the CO_2eq./KWh quantity is calculated using the dispatchable energy composition in Spain as reported by [50]:

- Solar Photovoltaic: 9%;
- Wind Power: 26%;
- Nuclear: 24%;
- Hydro: 13%;
- Cogeneration: 17%:
- Combined cycle: 11%;

Thus, the total emissions in CO_2eq./KWh [16] for wind/solar power generation and energy imported from the public grid are each:

$$Emission_{WT}(t_{start}, t_{end}) = 11.90 \cdot \sum_{t=t_{start}}^{t_{end}} P_t^{WT}, \tag{20}$$

$$Emission_{PV}(t_{start}, t_{end}) = 44.15 \cdot \sum_{t=t_{start}}^{t_{end}} P_t^{PV}, \tag{21}$$

$$\begin{aligned} Emission_{Buy}(t_{start}, t_{end}) = (44.15 \cdot 0.09 + 11.90 \cdot 0.26 + 5.75 \cdot 0.24 + \\ 75.60 \cdot 0.13 + 156.00 \cdot 0.28) \cdot \sum_{t=t_{start}}^{t_{end}} P_t^{Buy}, \end{aligned} \tag{22}$$

in which P_t^{WT} and P_t^{PV} stand for the energy generated from wind turbines and photovoltaic panels at time t, respectively. From that, Leite et al. [16] modeled an objective function for minimizing the amount of CO_2 emissions, according to:

$$\begin{aligned} \min Emission(t_{start}, t_{end}) = Emission_{WT}(t_{start}, t_{end}) + \\ Emission_{PV}(t_{start}, t_{end}) + Emission_{Buy}(t_{start}, t_{end}). \end{aligned} \tag{23}$$

3.4. ESS degradation objective

In our modeling approach, we have included a quantification of the Lithium-Ion battery (LIB) capacity degradation as a combination of calendar and cycle aging, given by [51]:

$$\begin{aligned} \Delta C = C_0 \cdot \Bigg(0.75\tau \cdot \sum_t \alpha_{cap_t} \cdot d^{-0.25} + \\ \frac{1}{\sqrt{EFC}} \left(\sum_{w=1}^{W} \beta_{cap}(DoD_w) \cdot DoD_w + \frac{1}{2} \sum_{h=1}^{H} \beta_{cap}(DoD_h) \cdot DoD_h \right) \Bigg), \end{aligned} \tag{24}$$

in which C_0 is the initial battery capacity, d are the total days, t is the model period in hours. W and H are the number of whole and half equivalent full cycles (EFC), obtained after applying rainflow cycle counting algorithm to the ESS state of charge's (SoC) profile [52]. DoD_w and DoD_h are the depths of discharge (DoD) associated with each whole and half EFCs, respectively. The term α_{cap} denotes the calendar aging factor, given by [51]

$$\alpha_{cap} = (7.543V - 23.75) \cdot 10^6 \cdot e^{6976/T}, \tag{25}$$

in which V and T are the battery's cell voltage and temperature (in K), respectively. It is assumed that the cell voltage is constant and equals to 3.7 [53] and T is equal to the ambient temperature T_{amb} converted from Celsius to Kelvin. In addition, the term β_{cap} describes the battery aging factor in terms of equivalent full cycles [51],

$$\beta_{cap}(DoD) = 7.348 \cdot 10^{-3} \cdot (\bar{V} - 3.667)^2 + 7.6 \cdot 10^{-4} + 4.081 \cdot 10^{-3} \cdot DoD. \quad (26)$$

The term $\bar{V}$ is the average cell voltage, which, in this work, is equal to the cell voltage $V = 3.7$. Therefore, the objective function for minimizing the accumulated ESS' degradation according to [51] is

$$\min Degradation(t_{start}, t_{end}) = \sum_{t=t_{start}}^{t_{end}} \Delta C(DoD(t)). \quad (27)$$

3.5. ERM Markov decision process

In the presented ERM problem, the energy management is performed through both the monthly control of the ESS' depth of discharge (DoD) [20], and the imported energy from the public grid. By setting a DoD value, the manager restricts the amount of energy that is available to be used from the ESS. The insufficient microgrid power is supplied by importing from the public grid. Thus, the system needs information from the environment to set different DoD values under different states. As a result, the system dynamics of the ERM can be formalized as a Markov decision process (MDP) characterized by a state space S, an action space A, and a reward R evaluated every month over a finite time horizon of a year. The state space S [20] is characterized by a $\mathbb{R}^3$ vector that contains the state information at each month m:

$$s_m \in S = \left\{ SoC(t), \frac{T_m^{buy}}{T_m^{load}}, \frac{T_m^{dch}}{T_m^{load}} \right\}, \quad (28)$$

in which T_m^{buy} denotes the total energy bought in month m, T_m^{dch} denotes the total energy discharged from the ESS in month m, and T_m^{load} is the accumulated load demand over the month m. Note that, since data time steps are defined in hours, each month consists of a batch of 720 h.

At each month, the agent decides the DoD that will be used. Thus, the action space contains eight available actions and is defined as

$$a_m \in A = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}. \quad (29)$$

After an action is performed, the ESS dynamics are updated as follows:

$$DoD(m) = a_m, \quad (30)$$

in which $DoD(m)$ indicates the DoD used for every hour corresponding to month m.

Finally, the multi-objective reward function proposed in this work is a $\mathbb{R}^3$ vector given by

$$\mathbf{R}(s_m, a_m) = \left[Cost(t_{start}^m, t_{end}^m), Emission(t_{start}^m, t_{end}^m), Degradation(t_{start}^m, t_{end}^m) \right], \quad (31)$$

in which $t_{start}^m = 720 \cdot (m-1)$ and $t_{end}^m = 720 \cdot m$. The components of the reward vector indicate a summation of the $Cost$, CO_2 $Emission$, and ESS $Degradation$ over the hours within the current month m.

Once the MDP is defined, reinforcement learning is used for solving the resulting control problem. The learning agent dynamics of the proposed control problem can be summarized as follows:

(1) **Choose** an action a_m.
(2) **Run** the microgrid for 720 h (30 days).
(3) **Update** the current state s_m.
(4) **Update** Costs.
(5) **Update** Emission.
(6) **Update** Degradation.
(7) **Update** battery capacity based on its degradation from the previous month.
(8) **Repeat** steps (1)–(7).

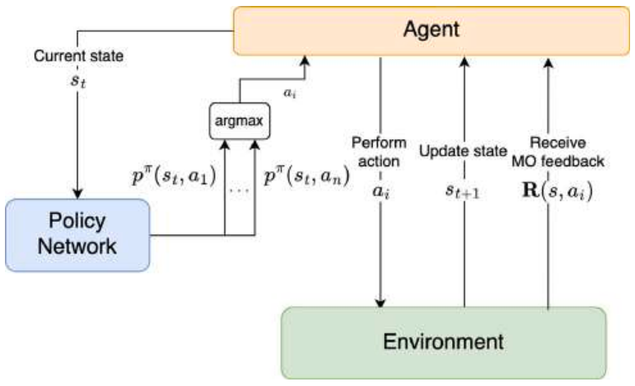


Fig. 2. Example of the agent's interaction with the environment, transitioning from state s_t to state s_{t+1} after selecting the action associated with the highest preference value $p(s, a)$.

4. The multi-objective evolutionary policy search algorithm

A model-free method that estimates action-preference values in MORL problems, the Multi-objective Evolutionary Police Search (MEPS) algorithm, is proposed in this work. MEPS falls in the RL "actor-only" family of algorithms and inherits NEAT structure to evolve artificial neural networks (ANNs) that implement deterministic policies in MORL environments. First, an initial random population P_t (for time $t = 0$) of n_p ANNs with one output node for each possible action is created. At each generation, individuals are evaluated according to a multi-objective reward function over h episodes, with $\mathbf{r}_h$ indicating the accumulated reward of each individual. Thereafter, the accumulated reward is used to sort the networks in population P_t by means of non-dominated sorting and a density measure.

Specifically, the ANNs employed in MEPS are designed to output action-preference values $p(s, a)$ for each available action a, given a state s as input. Moreover, to ensure that the agent deterministically follows the policy, the actions are selected in a greedy manner. Thus, Fig. 2 shows an example of an agent at state s_t selecting an action, and transitioning to state s_{t+1}.

The density measures considered are crowding distance (CD) [54] and hypervolume contribution (HVC) [55]. Crowding distance is defined as infinity for extremal solutions, and as the sum of side lengths of the cuboid that touches adjacent solutions in the case of a non-extremal solution on the Pareto front. It is meant to distribute solution points uniformly on the Pareto front. In contrast to this, the hypervolume contribution measure assigns a value to each solution according to its contribution to the hypervolume of the Pareto front. Consequently, it is meant to distribute them in a way that maximizes the covered hypervolume, focusing on knee-points without losing extremal points of the Pareto front.

Afterwards, a K_t set of n_p parents are randomly selected from the population P_t using a binary crowded tournament selection [54]. Note that if the density measure used is the hypervolume contribution, the density value of an individual x_i used in the tournament selection is $1/HVC(x_i)$. Thereafter, an offspring population Λ_t is generated by cloning selected parents and applying two types of mutation: structural and parametrical mutations. Structural mutations occur with a predefined probability and comprise (1) adding a new connection to previously unconnected nodes, and (2) adding a new hidden node.

It is important to note that MEPS provides feedforward ANNs, hence, no recurrent connections are allowed. Parametrical mutation encompasses updating connection weights and biases by adding a Gaussian noise with zero mean and standard deviation given by a parameter σ. Despite the performance of the crossover operator in the ablation studies presented in [56], it is not guaranteed to generate

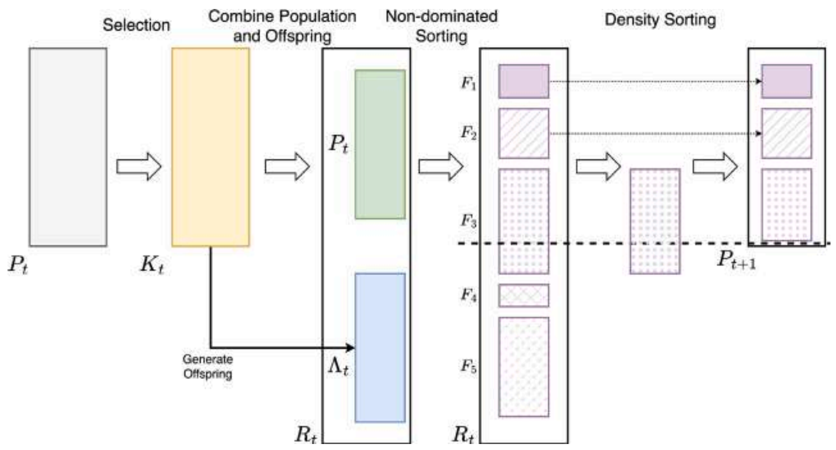


Fig. 3. Standard MEPS flowchart of the evolution from generation t to generation $t+1$.

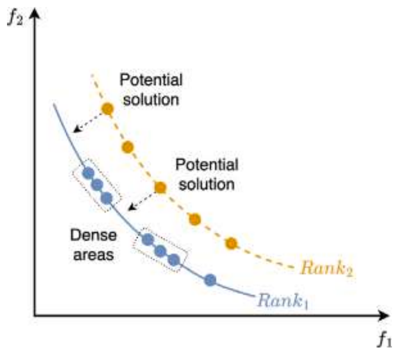


Fig. 4. Discarded potential solutions due to dense regions of non-dominated solutions. *Source:* Based on [58].

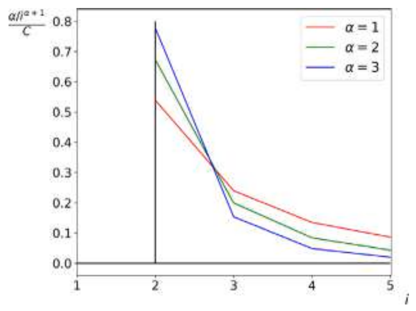


Fig. 5. Pareto distribution illustration for different values of α and four non-dominated fronts. The X-axis and Y-axis denote the non-dominated fronts and the proportion of the population that will be selected from each front, respectively.

a chromosome that preserves the good characteristics of the parents regarding the quality of the solution. As a result, the crossover operator disturbs NEAT's search ability, as attested in [57]. Therefore, the crossover operator has not been employed in MEPS.

The next generation population P_{t+1} is composed of the survivors selected from population $R_t = P_t \cup \Lambda_t$ of size $2n_p$. Differently from NEAT original proposal, MEPS does not utilize any speciation mechanism. Therefore, there are two possible approaches to perform survival selection. The first approach is similar to the NSGA-II [54] survival selection mechanism, in which the population R_t is sorted by means of non-dominated sorting in fronts or ranks, and then selected to the next generation iteratively by front. If the size of a front is bigger than the available slots in the next generation, the front is sorted in descending order according to the selected density measure, and the individuals from less dense regions are selected. Fig. 3 illustrates this approach.

The second approach comprises adaptively restricting the number of selected survivors from each front. From a multi-objective optimization perspective, in early generations, some undesired non-dominated individuals are selected over individuals from other fronts [58]. Fig. 4 shows a situation in which some ignored dominated solutions are potential candidates to improve the Pareto front. Likewise, from a neuroevolutionary perspective, discarded potential solutions may indicate newly introduced topologies, which are prematurely extincted. Thus, this work proposes a novel selection method that allows some individuals from higher ranks to survive. Specifically, a heavy-tailed Pareto distribution [59] is used for this purpose. The maximum number of survivors for each ith front is given by

$$n_{F_i} = \begin{cases} \frac{\alpha/i^{\alpha+1}}{C} \cdot \left(n_p - \lceil n_p \cdot ratio \rceil\right), & i > 1 \\ \lceil n_p \cdot ratio \rceil, & i = 1, \end{cases} \tag{32}$$

in which $C = \sum_{i>1}^{K} \alpha/i^{\alpha+1}$ for K non-dominated fronts, and $ratio$ denotes the fraction of individuals selected from the first front. The parameter α determines how heavy the distribution's tail is, as presented in Fig. 5. To avoid both premature convergence to an incomplete Pareto front and extinction of potential topology innovations, the $ratio$ is set to increase with the number of generations in a relationship defined by Eq. (33)

$$ratio = \begin{cases} 1, & t_{max} > t > t_r \\ \psi + t_r \cdot \frac{(1-\psi)}{t}, & t \leq t_r, \end{cases} \tag{33}$$

where $\psi \in (0,1)$ stands for the initial fraction of non-dominated individuals selected, which is increased over generations. This definition of $ratio$ allows the algorithm to gradually decrease the exploration of solutions from all non-dominated fronts. Moreover, after the pre-defined t_r generations, the heavy tail selection mechanism is replaced by the approach shown in Fig. 3.

Despite the fact that Eq. (32) denotes the maximum number n_{F_i} of allowed survivors in each front i, there may not exist all the n_{F_i} individuals in the ith front. To handle this problem, the procedure is started from the first front, and, if $|front_i| < n_{F_i}$, the remaining $n_{F_i} - |front_i|$ slots are added to the $n_{f_{i+1}}$ allowed survivors of front $i+1$. This procedure is repeated until all fronts are processed. Although unlikely, there could be situations in which there are still some slots left at the end. In such cases, the procedure is repeated from the beginning with the remaining individuals from each front until filling the remaining slots.

MEPS uses a memory of the same size as the population to store the best individuals ever found throughout generations. The memory is updated after the next generation population P_{t+1} selection. Let M be the memory population, the memory update mechanism, inspired by [60], can be summarized as follows:

(1) **Generate** a temporary population $M_{temp} = M \cup P_{t+1}$.
(2) **Clear** memory M.
(3) **Sort** the temporary population M_{temp} into non-dominated fronts.
(4) **If** the first front size is bigger than n_p, sort based on the density measure employed.
(5) **Assign** individuals from the sorted front to M until memory is full.

Algorithm 1: Memory update mechanism

```
Input: Population ($P_{t+1}$), Memory ($M$), density measure $S$
Output: Updated memory $M$
1  $M_{temp} \leftarrow M \cup P_{t+1}$;
2  $M \leftarrow \emptyset$;
3  Sort population in non-dominated fronts $F_1, \dots, F_k$ according to Pareto
     Dominance;
4  if $|F_1| > n_p$ then
5      Sort $F_1$ based on the density measure $S$;
6  end
7  $i \leftarrow 0$;
8  while $|M| < n_p$ do
9      Assign the $i$-th individual from $F_1$ to $M$;
10     $i \leftarrow i + 1$;
11 end
12 return $M$
```

Finally, MEPS avoids performing gradient updates and computing value function or Q-value function estimators as it updates network policies in an evolutionary manner. Additionally, MEPS is classified as a multi-policy algorithm, as it leverages population-based techniques and produces a set of Pareto-optimal policies [61]. Algorithm 2 shows the pseudocode for the proposed method. In addition, Table 3 summarizes a description of the parameters used.

In order to provide a clear distinction among MEPS versions, the different configurations of MEPS are identified based on both the survivor selection method and the density measure used. The survivor selection method can be either based on heavy tail selection (H1) or based on non-dominance sorting (H0). The density measure can be crowding distance (S0) or hypervolume contribution (S1). In this way, $4(2 \cdot 2)$ versions of MEPS are proposed and analyzed. As an example, one possible setting of MEPS is H1/S1, in which the proposed heavy tail survivor selection mechanism is used along with hypervolume contribution as the density measure.

4.1. Computational complexity

Subsequently, we evaluate the time complexity of MEPS as shown in Algorithm 2 according to the density measure employed. When using crowding distance, the binary crowded tournament selection operator in Step (5) involves running non-dominated sorting for all fronts and sorting based on the density measure. So, it is executed with time complexity $O(m \cdot n_p^2)$, in which m stands for the number of objectives. In Step (6), the offspring generation is executed with time complexity $O(n_p \cdot (TC+TN))$, in which TC and TN are the total maximum number of connections and nodes, respectively. The fitness evaluation present in Steps (3) and (7) is run with time complexity $O(h \cdot |A|)$. Sorting is executed in Step (9) with time complexity $O(m \cdot (2n_p)^2)$. In Steps (10)–(29) and (31)–(42), the worst-case scenario comprises sorting only one front with size $2n_p$. Hence, the time complexity is $O(m \cdot (2n_p)\log(2n_p))$.

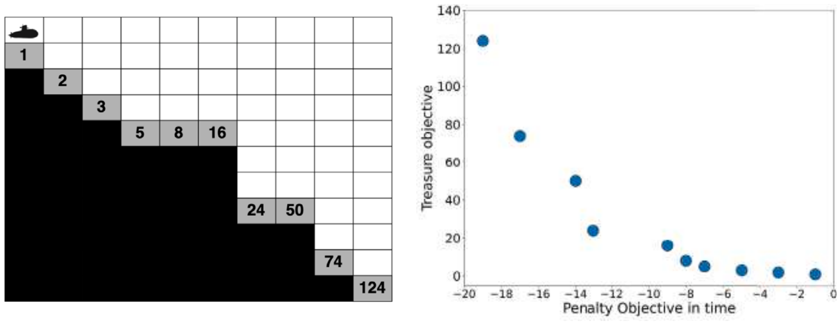
Afterwards in Step (44), time complexity for the memory update mechanism presented in Algorithm 1 depends on whether the number of objectives in the problem is $m = 2$ or $m = 3$. For $m = 2$ and $m = 3$, this memory update is executed with time complexity $O(2n_p \cdot \log(2n_p))$ and $O(2n_p \cdot \log^2(2n_p))$, respectively [62]. Finally, after omitting low order terms, the MEPS version that employs crowding distance is governed by the non-dominated sorting component of the algorithm and executed with time complexity $O(t_{max} \cdot m \cdot n_p^2)$. The MEPS counterpart that employs hypervolume contribution is governed by the calculation of the hypervolume, which is executed with time complexity $O(n_p^3 \cdot m^2)$ [55].

Table 3
Parameters description.

Parameter	Description
n_p	Population size
S	Density measure function
$\phi(x)$	Activation function
ψ	Initial fraction selected from first front
t_{max}	Total generations
t_r	Final generation of the heavy tail survivor selection
α	Heavy tail selection parameter
n_i	Number of input nodes
n_h	Number of initial hidden nodes
n_o	Number of output nodes
p_{ac}	"Add connection" mutation probability
p_{an}	"Add node" mutation probability
σ	Parametrical mutation standard deviation
HT	Indicate the use or not of the heavy tail survivor selection
h	Length of the episode to evaluate the agent

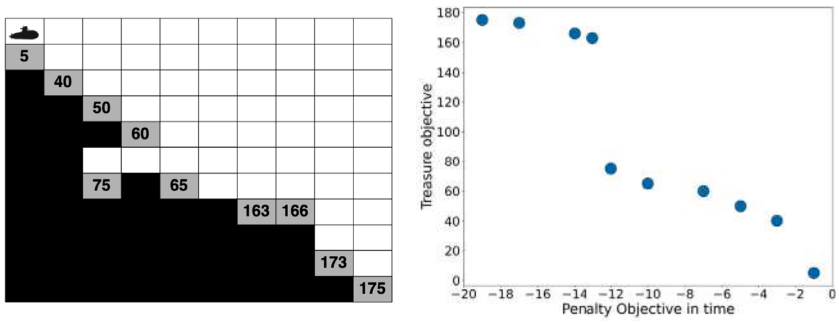
Algorithm 2: The Multi-Objective Evolutionary Policy Search algorithm (MEPS)

```
Input: $n_p$, $\psi$, $t_{max}$, $t_r$, $\alpha$, $\phi(x)$, $\sigma$, $p_{ac}$, $p_{an}$, $n_i$, $n_o$, $n_h$, $S$, $HT$, $h$
Output: Memory $M$
1  $t \leftarrow 0$;
2  Initialize population $P_t$ with fully connected ANNs containing $n_i$ input nodes, $n_h$
     hidden nodes and $n_o$ output nodes;
3  Evaluate each individual of $P_t$ for an episode of length $h$;
4  while $t < t_{max}$ do
5      Generate a population $K_t$ of $n_p$ parents selected from the population $P_t$ using
         binary crowded tournament selection and the selected density measure $S$;
6      Generate $\Lambda_t$ offspring population
7  end
8  ;
9  Evaluate each individual of $\Lambda_t$ for an episode of length $h$;
10 $R_t \leftarrow P_t \cup \Lambda_t$;
11 Sort population $R_t$ in non-dominated fronts $F_1, \dots, F_k$;
12 $P_{t+1} \leftarrow \emptyset$;
13 if $t \leq t_r$ OR $HT > 0$ then
       /* runs heavy tail selection */
14     Calculate $ratio$ according to Eq. (33);
15     Calculate maximum number of survivors $n_{F_1}, \dots, n_{F_k}$ for each front using
         parameters $\alpha$ and $\psi$;
16     $remaining \leftarrow 0$;
17     For each Front $F_i$ do
18         $n_{F_i} \leftarrow n_{F_i} + remaining$;
19         if $n_{F_i} < |F_i|$ OR $\min(|F_i|, n_{F_i}) > n_p - |P_{t+1}|$ then
20             Sort front based on density measure function $S$;
21             Add the $\min(|F_i|, n_{F_i}, n_p - |P_{t+1}|)$ fittest individuals from $F_i$ to $P_{t+1}$;
22         end
23         else
24             $P_{t+1} \leftarrow P_{t+1} \cup F_i$;
25         end
26         if $n_p = |P_{t+1}|$ then
27             break;
28         end
29         $remaining \leftarrow \max(n_{F_i} - \min(|F_i|, n_p - |P_{t+1}|), 0)$;
30     end
31 end
32 else
33     For each Front $F_i$ do
34         if $|F_i| > n_p - |P_{t+1}|$ then
35             Sort front based on density measure function $S$;
36             Add the $n_p - |P_{t+1}|$ fittest individuals from $F_i$ to $P_{t+1}$;
37         end
38         else
39             $P_{t+1} \leftarrow P_{t+1} \cup F_i$;
40         end
41         if $n_p = |P_{t+1}|$ then
42             break;
43         end
44     end
45 end
46 Update memory $M$ following memory update procedure;
47 $t \leftarrow t + 1$;
```



(a) Environment (b) Pareto Front

Fig. 6. Deep Sea Treasure (DST) MORL benchmark.



(a) Environment (b) Pareto Front

Fig. 7. Modified Bountiful Sea Treasure (MBST) MORL benchmark.

As a result, both MEPS versions are polynomial in the population size n_p. A complete MEPS code version, based on [63], is available at https://github.com/gabrielmatos26/MEPS-Paper.

5. Experiments

In order to evaluate the performance and quality of our proposal, we conducted a two-fold experimental analysis. First, we have selected traditional MORL benchmark environments with 2 and 3 objectives. Then, we proposed a new 2-objective MORL benchmark environment as a variation of the Deep Sea Treasure [64], Discontinuous Deep Sea Treasure. Finally, we evaluate and analyze the performance of MEPS as an EMS controlling the depth of discharge of an ESS in a microgrid.

From the several metrics available to evaluate the performance of a MORL algorithm, [64], we have employed the hypervolume indicator (HV) to evaluate the performance of each algorithm. The HV value is calculated using the multiple policies' accumulated rewards obtained by the learning agents at the end of the pre-defined number of generations [64]. The HV measure is selected because it provides a single value to compare different algorithms and it does not require knowledge about the true Pareto front or its approximation. Moreover, All the computational simulations were conducted using an Intel(R) Core(TM) i9-10900X CPU@3.70 GHz and 64 GB RAM, with Windows 10 Pro. The simulation code is implemented in Python 3.10.

5.1. Comparison with benchmark MORL methods

The proposed versions of MEPS are compared with three state-of-the-art MORL algorithms, Pareto Q-Learning (PQL) [65], Q-Managed (QM) [66] and Multi-Policy Soft Actor-Critic (MPSAC) [67]. For a fair comparison, as Q-Managed was initially proposed only for 2-objective problems, we did not analyze its performance on environments with 3 objectives. Five MORL benchmark environments with 2 and 3 objectives have been selected to evaluate the proposal for an MORL algorithm, namely Deep Sea Treasure [64], Pressurized Bountiful Sea Treasure [65], Modified Bountiful Sea Treasure [66], Space Exploration [68], and Bonus World [68]. Besides, a sixth novel MORL environment is proposed and used to validate MEPS proposal.

Deep Sea Treasure (DST) is an episodic deterministic environment in which an agent controls a submarine searching for an undersea treasure. The environment is a rectangular grid with 11 rows and 10 columns (Fig. 6(a)), containing 10 treasures of varying values at different locations. Each episode starts with the submarine at the top left corner and ends if either a treasure is found or h actions were taken. The treasures are arranged to have the lowest treasure value at the closest location to the starting point and the highest value at the furthest location, which means the treasure value is inversely proportional to its distance from the source. The agent can move around the environment by performing four available actions, representing the four cardinal directions - (1) right, (2) left, (3) down, or (4) up. Each action taken by the agent incurs a time penalty,[3] which is a -1 decrease applied to the time penalty objective. Note that, this time penalty objective is not a time unit, but is the sum of penalties the agent receives for the time it takes in interacting with the environment.

Additionally, the agent's goals are to minimize the time penalty received in finding a treasure and maximize the treasure value. The reward vector $\mathbf{r}_h \in \mathbb{R}^2$ consists of the time penalty as the first element and the treasure value as the second element. The true Pareto front (Fig. 6(b)) is globally concave with some local concavities in the second, fourth, and sixth points from left to right.

A variation of the DST is the Modified Bountiful Sea Treasure (MBST) environment. This is a 2-objective problem in which not only the treasure values but also their location are modified (Fig. 7(a)). To find all the Pareto optimal policies, the agent must learn how to

[3] In order to maintain the same description for penalty objective in time from [66], it is presented as a negative time penalty in the Pareto fronts.

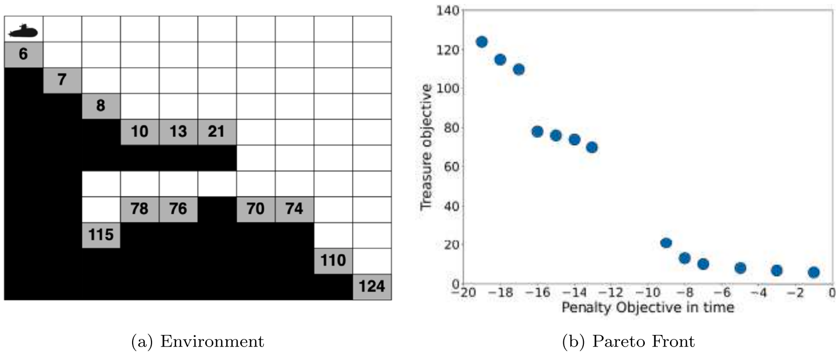


Fig. 8. Discontinuous Deep Sea Treasure (DDST) MORL benchmark.

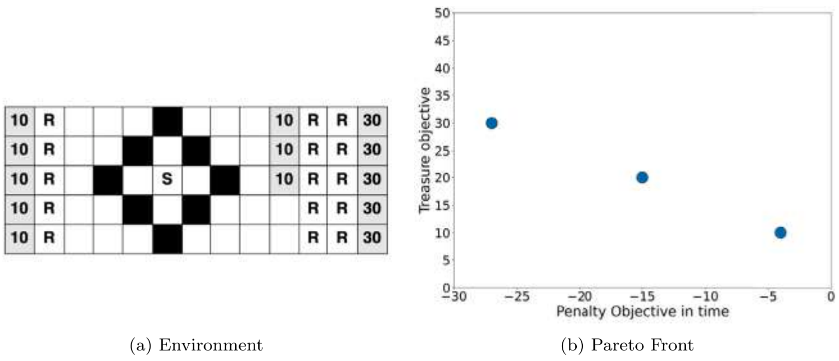


Fig. 9. Space Exploration (SE) MORL benchmark.

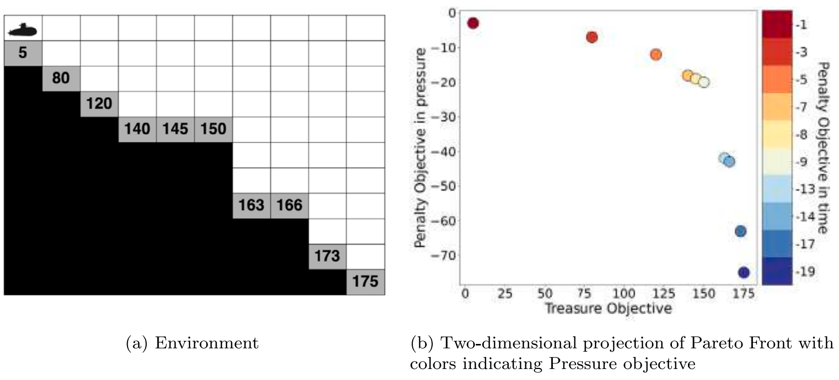


Fig. 10. Pressurized Bountiful Sea Treasure (PBST) MORL benchmark.

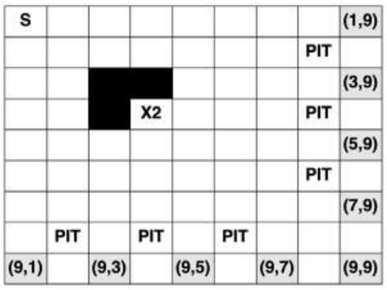
enter in a tunnel between two locations, which makes learning more challenging. With this change, the Pareto front (Fig. 7(b)) is divided into one convex part and one non-convex part.

Afterwards, a new variation of the DST is proposed, dubbed Discontinuous DST (DDST). Inspired by the ZDT set of MOO benchmark functions [69], specifically ZDT3, both treasure values and treasure locations are modified to create two discontinuities in the Pareto front. Furthermore, to increase difficulty in learning, DDST includes a tunnel containing three treasures (Fig. 8(a)). To discover all the Pareto optimal policies, the agent needs to decide not collecting the closest reward to the tunnel entrance but, instead, to enter the tunnel. The true Pareto front (Fig. 8(b)) is also divided into three parts, mixing convex and non-convex parts.

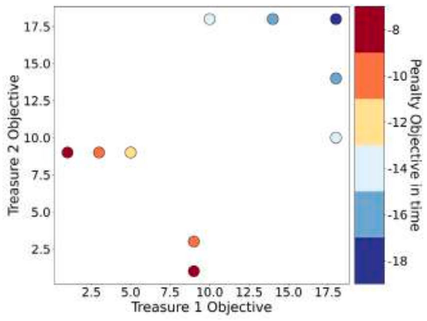
Another 2-objective benchmark environment is the Space Exploration (SE) (Fig. 9(a)), in which the agent controls a spaceship that starts the episode in the location marked ‘S’ with the goal of discovering a habitable planet while minimizing the radiation it is exposed during the search. The first objective is the radiation penalty. After every action, the radiation objective is penalized by -1 or -11 if the next state is marked ‘R’. The second objective is the mission level of success, which denotes the habitability of the planet, or -100 if the agent moves to a black cell, representing an asteroid.

The episode ends whether the agent performs h actions, reaches a planet, or collides with an asteroid. Unlike the previous environments, in SE, the agent is also allowed to move diagonally, totaling eight directions. Additionally, a movement that would lead the agent out of the grid, takes it to the opposite edge of the grid. For example, if the agent moves down from the bottom row of the grid, the next state will be the top row of the grid, maintaining the column. Thus, in this environment, every action leads to a state change. The Pareto front is illustrated in Fig. 9(b).

Pressurized Bountiful Sea Treasure (PBST) Environment (Fig. 10(a)) is a variation of DST in which a third objective representing the pressure penalty is included. Similarly to penalty in time, this objective denotes the pressure penalty received by the agent for staying underwater at the depth indicated by the row, with an initial value



(a) Environment



(b) Two-dimensional projection of Pareto Front with colors indicating Time objective

Fig. 11. Bonus World (BW) MORL benchmark.

equal to −1. For example, if the agent moves down from the starting point, it receives a −2 pressure penalty for row 2 and, therefore the total pressure penalty is −3. The first and second objectives remain the same as in the DST environment. Moreover, the treasure values are modified to create a Pareto front that is globally convex. The set of Pareto optimal policies is presented in Fig. 10(b).

Bonus world (Fig. 11(a)) is a 3-objective environment similar to DST, in which the agent starts at cell marked ‘S’ and is allowed to move in the four cardinal directions with black cells indicating walls the agent cannot pass through. At every step, a time penalty of −1 is applied to the first objective. The gray cells correspond to terminal states that reward the agent with values corresponding to treasure 1 and treasure 2. If the agent reaches a cell marked ‘X2’, a bonus is activated and the values for treasure 1 and treasure 2 are doubled. Contrarily, if the agent enters a cell marked ‘PIT’, it not only loses the bonus but is also moved to the start state. Although not all the Pareto optimal policies require the agent to activate the bonus, some Pareto optimal policies are only reachable with the bonus activated. Therefore, this is a difficult environment because the agent must learn to both avoid penalties and activate the bonus before moving to some terminal states. The set of Pareto optimal policies is shown in Fig. 11(b).

In this analysis, the obtained HVs are compared to the HV of the true Pareto front in each benchmark environment. For the Bonus World and Space Exploration environments, the choice of reference point was based on choosing a point that is worse than the nadir point. In the Deep Sea Treasure and its variations, the reference points used are the same as in [65]. Moreover, as a measure of network complexity, MEPS and MPSAC are compared in terms of their average number of network nodes and connections.

For each benchmark environment, 20 independent executions were performed. Since the main goal of this experiment is to validate the proposed approach, no fine-tuning of the parameters has been done. Therefore, PQL and QM were run with the hyperparameter values found in the literature. Also, MPSAC execution was divided into two stages. The hyperparameters used throughout executions are presented in Table 4 and the initial topology used in MEPS is presented in Fig. 12. Regarding the output layer for both MEPS and MPSAC networks, the number of nodes is equal to the number of available actions in the environment. As a result, the output layer contains four nodes in the DST, MBST, DDST, PBST, and BW environments, and 8 nodes for the SE environment.

Detailed benchmark results based on the hypervolume indicator are shown in Table 5 and the algorithm with superior performance is indicated in bold for each problem. The normalized average hypervolume obtained by each algorithm on each benchmark environment is shown in Fig. 13. In both DST (Fig. 13(a)) and PBST (Fig. 13(b)) problems, all the four versions of MEPS are able to find the entire Pareto front in less than half the number of maximum generations with similar topologies. The final networks evolved using MEPS are composed of an average of 25 connections and 10 (1 input, 5 hidden, and 4 output) nodes.

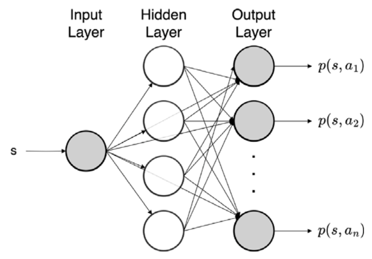


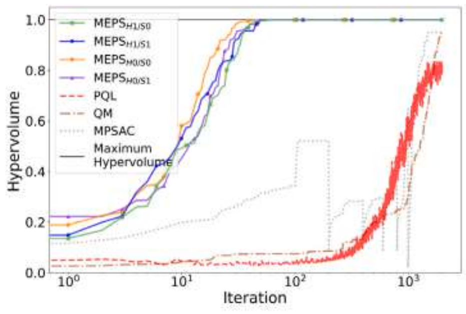
Fig. 12. MEPS initial topology configuration for benchmark tests.

Table 4
Parameter initialization values used by the algorithms in benchmark tests.

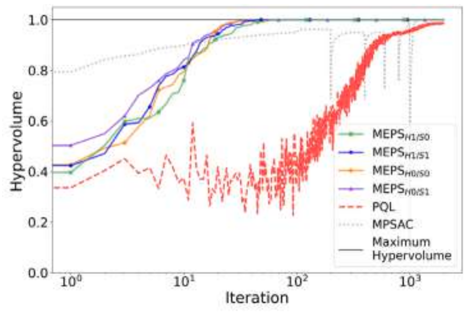
Description	MEPS	PQL	QM	MPSAC
Population size	50	–	50	50
Activation function	relu	–	–	relu
Initial fraction selected from first front	0.5	–	–	–
Final generation of the heavy tail survivor selection	1000	–	–	–
Heavy tail selection parameter	1.0	–	–	–
Number of input nodes	1	–	–	1
Number of initial hidden nodes	4	–	–	64
Number of output nodes	4 or 8	–	–	4 or 8
“Add connection” mutation probability	0.2	–	–	–
“Add node” mutation probability	0.2	–	–	–
Parametrical mutation standard deviation	0.5	–	–	1.0
Learning rate	–	–	–	0.001
Gamma	–	–	–	0.99
Generations	2000	2000	2000	1000 + 1000
Episode length			20	

PQL, QM, and MPSAC are not able to consistently find all the non-dominated policies in DST, only achieving the maximum hypervolume value in some executions. However, MPSAC is able to find all the non-dominated policies in all the 20 runs in PBST. It is worth noting that, due to the simpler Pareto front of these benchmarks compared to the other benchmarks, the use of the proposed heavy tail survivor selection operator (H1/S0 and H1/S1) results in a delay in the search. Hence, both H0/S0 and H0/S1 achieve the maximum hypervolume in fewer generations.

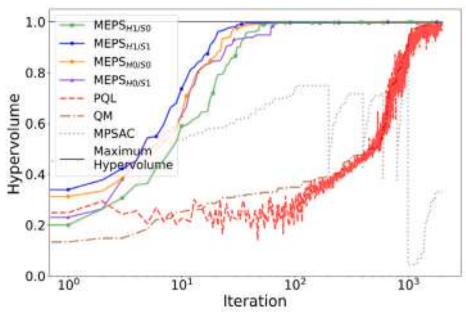
In the MBST problem (Fig. 13(c)), with the exception of MPSAC, all the algorithms obtained all the Pareto optimal solutions in at least one execution. The effectiveness of the proposed survivor selection method is demonstrated with H1/S0 and H1/S1 achieving the highest mean hypervolume values. Specifically, H1/S0 can find all the Pareto optimal solutions in all the executions.



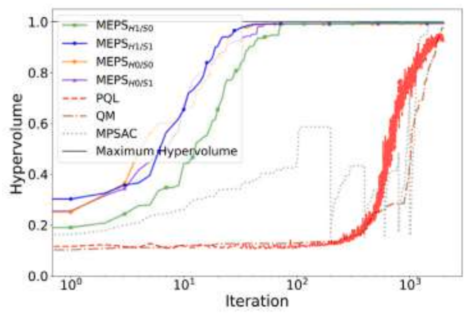
(a) DST



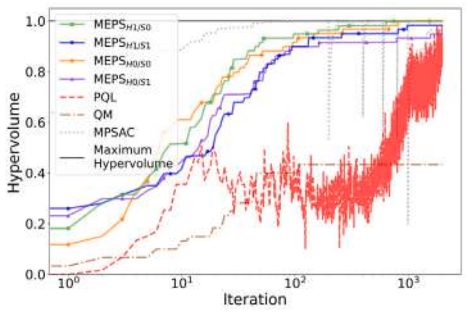
(b) PBST



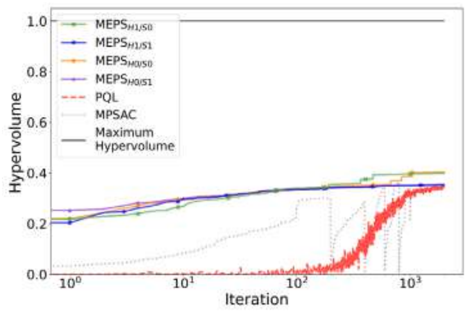
(c) MBST



(d) DDST



(e) SE



(f) BW

Fig. 13. Average normalized hypervolumes obtained by PQL, QM, MPSAC, and MEPS versions in all the MORL benchmark environments.

Regarding the topology of the final networks, H1/S0 networks provide an average of 26 connections and 10 (1 input, 5 hidden, and 4 output) nodes. By increasing the difficulty, as in the DDST problem (Fig. 13(d)), the MEPS stands out as an advantageous algorithm, as PQL, QM, and MPSAC cannot find all the Pareto optimal solutions in any execution. Among the MEPS configurations, H1/S0 achieves the highest mean hypervolume with less complex networks, containing an average of 28 connections and 12 nodes. Therefore, it can be said that H1/S0 achieved the highest mean hypervolumes in the DST problem and its variations, providing a population of solutions comprising less complex networks.

Furthermore, in both SE (Fig. 13(e)) and BW (Fig. 13(f)) problems, the MEPS obtained the highest average hypervolumes, which indicates a better approximation of the true Pareto front. Similarly to the DST and PBST results, the use of the heavy tail survivor selection operator in a problem with a Pareto front without any discontinuities, such as SE, worsened the results. The configuration H0/S0 consistently found the entire Pareto front, while the use of hypervolume contribution as the density measure achieved the worst results among MEPS versions. Moreover, MEPS versions using crowding distance, H0/S0 and H1/S0, obtained networks with a smaller average number of connections and nodes, 52 and 18, respectively.

It is worth mentioning that, despite the simple Pareto front (only three solutions) in the SE problem, the increased action space poses a more difficult problem in searching for optimal policies. This is indicated by results showing that six out of seven algorithms struggled to obtain all the Pareto optimal solutions across runs. Unlike previous benchmark results, BW stands as the most difficult benchmark problem, in which none of the algorithms were able to approximate the Pareto front. H1/S0 nonetheless achieved the highest average hypervolume among the algorithms, with an average of only 7 hidden nodes (12 nodes in total) and 28 connections.

The mean and standard deviation values are preliminary measures that are often not sufficient to provide an effective analysis of the obtained results. Hence, a Kruskal–Wallis test is performed, aiming to find possible differences between the mean hypervolumes. Using a significance value of 5%, a p-value below 0.05 is found indicating that there is a difference among the means. Accordingly, a Wilcoxon signed-rank test is carried out to perform a pairwise analysis to identify the differences between the samples analyzed. Table 6 provides the results of the statistical test. The test results indicated that the results of algorithms under the column with a (+) differ with 95% confidence from those under the column with a (−).

The obtained results indicate that MEPS versions obtain competitive results in all the benchmark problems evaluated. In both DST and PBST, MEPS was as efficient as MPSAC. Moreover, in MPSAC each individual was composed of five networks with 320 $(64+64+(64\cdot 7))$ connections and 69 (1 input, 64 hidden, 4 output) nodes each, whereas MEPS

Table 5
Hypervolume analysis for each MORL benchmark environment (labeled as Env). The hypervolume for the true Pareto front (PF) is calculated with reference points given in parenthesis.

Env	PF	Algorithm	Mean	Std	Min	Median	Max
		MEPS$_{H1/S0}$	**1155.00**	**0.00**	**1155.00**	**1155.00**	**1155.00**
		MEPS$_{H1/S1}$	**1155.00**	**0.00**	**1155.00**	**1155.00**	**1155.00**
		MEPS$_{H0/S0}$	**1155.00**	**0.00**	**1155.00**	**1155.00**	**1155.00**
DST	1155.00 (25, 0)	**MEPS$_{H0/S1}$**	**1155.00**	**0.00**	**1155.00**	**1155.00**	**1155.00**
		PQL	947.40	223.66	663.00	1005.00	1155.00
		QM	1091.80	120.22	759.00	1155.00	1155.00
		MPSAC	1098.95	250.66	34.00	1155.00	1155.00
		MEPS$_{H1/S0}$	**358 636.00**	**0.00**	**358 636.00**	**358 636.00**	**358 636.00**
		MEPS$_{H1/S1}$	**358 636.00**	**0.00**	**358 636.00**	**358 636.00**	**358 636.00**
		MEPS$_{H0/S0}$	**358 636.00**	**0.00**	**358 636.00**	**358 636.00**	**358 636.00**
PBST	358 636.00 (25, 0)	**MEPS$_{H0/S1}$**	**358 636.00**	**0.00**	**358 636.00**	**358 636.00**	**358 636.00**
		PQL	353 983.20	1250.58	351 582.00	353 790.00	356 175.00
		QM	–	–	–	–	–
		MPSAC	**358 636.00**	0.00	**358 636.00**	**358 636.00**	**358 636.00**
		MEPS$_{H1/S0}$	**2632.00**	**0.00**	**2632.00**	**2632.00**	**2632.00**
		MEPS$_{H1/S1}$	2630.50	3.66	2622.00	2632.00	2632.00
		MEPS$_{H0/S0}$	2626.00	5.03	2622.00	2632.00	2632.00
MBST	2632.00 (25, 0)	MEPS$_{H0/S1}$	2623.00	3.08	2622.00	2622.00	2632.00
		PQL	2608.20	26.97	2564.00	2620.00	2632.00
		QM	2629.50	11.18	2582.00	2632.00	2632.00
		MPSAC	870.60	1176.35	120.00	120.00	2622.00
		MEPS$_{H1/S0}$	**1412.15**	**2.08**	**1411.00**	**1411.00**	**1416.00**
		MEPS$_{H1/S1}$	1409.95	2.82	1405.00	1411.00	1416.00
		MEPS$_{H0/S0}$	1408.95	3.05	1405.00	1409.00	1416.00
DDST	1416.00 (25, 0)	MEPS$_{H0/S1}$	1407.55	3.42	1405.00	1405.00	1416.00
		PQL	1310.15	114.26	1033.00	1330.00	1411.00
		QM	1383.40	38.57	1299.00	1405.00	1405.00
		MPSAC	1406.60	2.01	1405.00	1405.00	1409.00
		MEPS$_{H1/S0}$	11 539.00	4.47	11 520.00	11 540.00	11 540.00
		MEPS$_{H1/S1}$	11 347.00	832.95	7810.00	11 540.00	11 540.00
		MEPS$_{H0/S0}$	**11 540.00**	**0.00**	**11 540.00**	**11 540.00**	**11 540.00**
SE	11 540.00 (400, 0)	MEPS$_{H0/S1}$	10 956.50	1388.37	7700.00	11 540.00	11 540.00
		PQL	10 740.00	1954.72	3960.00	11 420.00	11 540.00
		QM	5006.00	3313.47	0.00	7570.00	7790.00
		MPSAC	11 131.00	1139.12	7810.00	11 530.00	11 540.00
		MEPS$_{H1/S0}$	**799.12**	**166.09**	**690.46**	**724.86**	**1210.58**
		MEPS$_{H1/S1}$	709.32	6.46	693.90	709.12	720.66
		MEPS$_{H0/S0}$	772.10	127.18	708.68	722.19	1121.11
BW	2038.00 (20, 0, 0)	MEPS$_{H0/S1}$	711.55	9.35	691.83	711.53	728.89
		PQL	704.00	29.42	608.00	708.00	732.00
		QM	–	–	–	–	–
		MPSAC	684.40	20.18	660.00	692.00	716.00

Table 6
Wilcoxon signed-rank test using hypervolume analysis of each MORL problem. Algorithms in the (+) column are statistically significant compared to algorithms in the column (−).

Env	Statistically Significant	
	(+)	(−)
DST	MEPS (all versions), MPSAC	PQL, QM
PBST	MEPS (all versions), MPSAC	PQL
MBST	MEPS$_{H1/S0}$, MEPS$_{H1/S1}$, QM	MEPS$_{H0/S0}$, MEPS$_{H0/S1}$ PQL, MPSAC
DDST	MEPS$_{H1/S0}$	MEPS$_{H1/S1}$, MEPS$_{H0/S0}$, MEPS$_{H0/S1}$ PQL, QM, MPSAC
SE	MEPS (all versions)	PQL, QM, MPSAC
BW	MEPS$_{H1/S0}$, MEPS$_{H0/S0}$	MEPS$_{H1/S1}$, MEPS$_{H0/S1}$ PQL, QM, MPSAC

outputs one network per individual, with an average of 25 connections and 10 nodes. Despite obtaining similar results as the QM algorithm in the MBST problem, MEPS versions were the best algorithms in DDST, SE, and BW. Besides, H1/S0 was the best algorithm for the DDST problem, indicating that the proposal for a survivor selection mechanism, that attempts to search through all the non-dominated fronts, leads to competitive results in a problem with discontinuities and non-convex regions in the Pareto front.

5.2. Ablation study

After evaluating the different versions of MEPS in multiple benchmark environments, we performed an ablation study to investigate the effects of both parametrical and structural mutation operations. Ablations can significantly harm performance. Accordingly, we selected the DDST environment for the ablation study. This environment is complex enough even to unablated MEPS versions. Thus, we believe that it is suitable to compare ablated MEPS to its unablated counterpart.

According to the previously presented results, the MEPS version using crowding distance and the heavy tail survivor selection operator, namely H1/S0, obtained the most competitive performance among the comparisons. Thus, we selected H1/S0 as MEPS unablated version and, consequently, ablations were performed in this MEPS version.

In order to assess the hypothesis that H1/S0 using both structural mutations in combination with parametrical mutation is the most competitive option for the DDST problem, we performed six ablations as follows:

- Ablation 1: allowed only structural mutation of adding a new connection to previously unconnected nodes;
- Ablation 2: allowed only structural mutation of adding a new hidden node;
- Ablation 3: allowed both structural mutations;
- Ablation 4: allowed only parametrical mutation

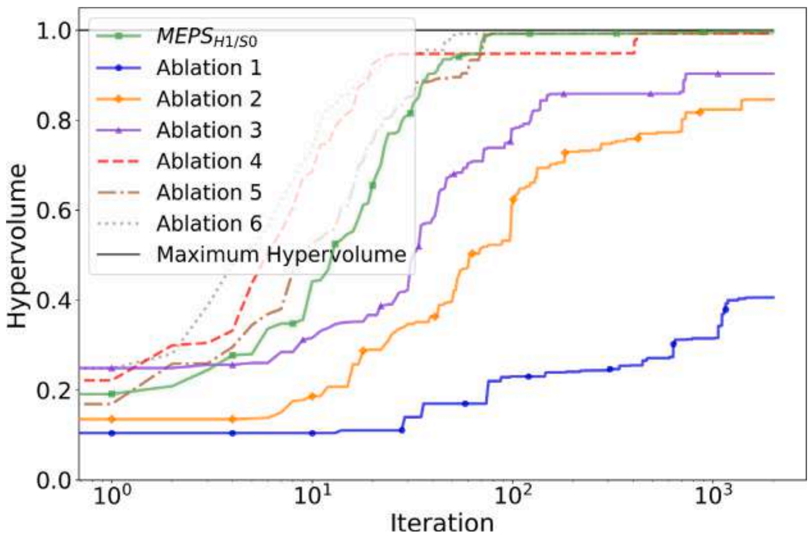


Fig. 14. Average normalized hypervolumes obtained by the unablated H1/S0 version and its six ablated versions in the DDST MORL benchmark environment.

- Ablation 5: allowed both structural mutation of adding a new connection to previously unconnected nodes and parametrical mutation;
- Ablation 6: allowed both structural mutation of adding a new hidden node and parametrical mutation;

Each ablated version was initialized using the parameters presented in Table 4, zeroing each respective restricted mutation probability. For example, Ablation 1 was initialized with zero probability for the "add connection" mutation and zero standard deviation for parametrical mutation. With respect to the initial topology, ablated versions 2, 3, 4, and 6 were initialized as presented in Fig. 12. However, if ablated versions 1 and 5 were initialized containing all the possible connections, the "add connection" mutation would never be executed and, therefore, we would not be able to assess its effect on the final performance. As a result, networks in ablated versions 1 and 5 were initialized by randomly connecting 50% of the possible connections.

Subsequently, 20 independent executions were performed for each ablated version. Fig. 14 shows the normalized average hypervolume obtained by the unablated version and each of the ablations. Detailed results based on the hypervolume indicator are presented in Table 7. Ablations 1 and 2, which constrained the mutations to either add a new hidden node or a new connection, presented the worst average hypervolume values. Although Ablation 3, which employs both structural mutations, improved its performance when compared to Ablations 1 and 2, its performance is still far from those that employ parametrical mutation. This highlights the importance of the parametrical mutation in MEPS.

Among the ablated versions that employed parametrical mutation, Ablation 4, which does not use any structural mutation, achieved the highest mean hypervolume value. This indicates that the combination of parametrical mutation with either "add connection" (Ablation 5) or "add node" (Ablation 6) mutations is still worse than employing all operators together as in the unablated version. Finally, we can see that only the unablated version of MEPS was able to find all the Pareto optimal solutions.

5.3. Solving the ERM problem

In this section, we analyze the results of MEPS in the proposed ERM problem of controlling the DoD of an ESS in a microgrid containing both solar and wind generation. Moreover, two standard MORL algorithms are used for comparison, Multi-Policy Soft Actor Critic and Multi-Objective Deep Q Networks (MODQN). The former is based on ANNs and multi-objective CMA-ES [67], while the latter is based on the MORL framework for Deep RL proposed in [70].

With respect to the configurations, MODQN used two 64-neuron fully connected layers. MPSAC used ANNs with one hidden layer of size 64, and MEPS initialized the ANN population without any hidden layers. ReLU function activation [71] was used in neurons for all the algorithms. In addition, the output layer configuration was the same for all three algorithms and contained 8 neurons for the 8 actions as stated in Eq. (29). The remaining parameters used are listed in Table 8.

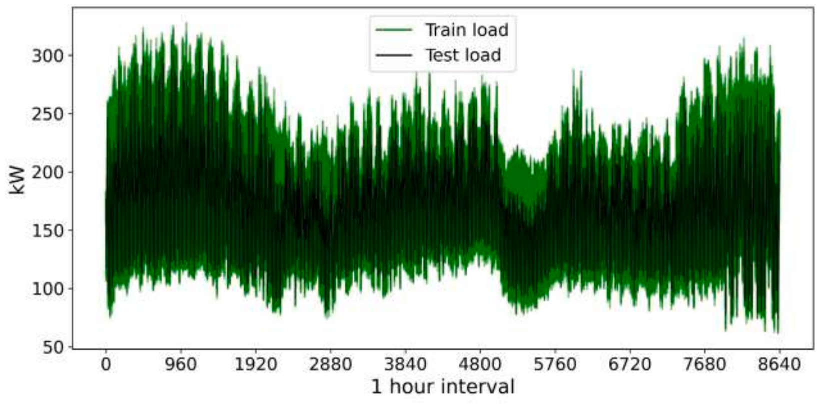
In order to assess the generalization ability of each algorithm, different load scenarios were used in training and testing. The test scenario is depicted in Fig. 15(a) as a black line, and the green area shows the range of the noise added to the test scenario at each hour

Table 7
Hypervolume analysis of H1/S0 unablated version and its six ablated versions in the DDST MORL benchmark environment. The hypervolume for the true Pareto front (PF) is calculated with reference points given in parenthesis.

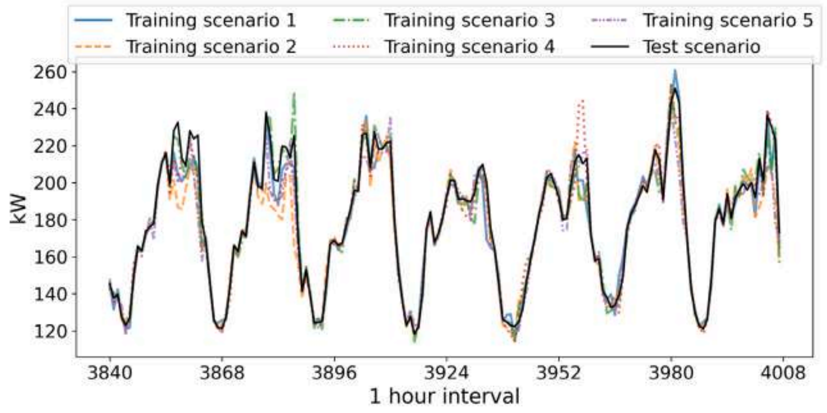
PF	Algorithm	Mean	Std	Min	Median	Max
1416.00 (25, 0)	**$MEPS_{H1/S0}$**	**1412.15**	**2.08**	**1411.00**	**1411.00**	**1416.00**
	Ablation 1	574.30	602.43	144.00	144.00	1405.00
	Ablation 2	1198.05	461.82	144.00	1405.00	1411.00
	Ablation 3	1279.20	388.23	144.00	1405.00	1411.00
	Ablation 4	1407.60	2.76	1405.00	1407.00	1411.00
	Ablation 5	1405.90	2.20	1405.00	1405.00	1411.00
	Ablation 6	1406.20	2.46	1405.00	1405.00	1411.00

Table 8
Parameter initialization values used by the algorithms in the microgrid environment.

Description	MEPS	MPSAC	MODQN
Population size		20	
Initial fraction selected from first front	0.5	–	–
Final generation of the heavy tail survivor selection	250	–	–
Heavy tail selection parameter	1.0	–	–
"Add connection" mutation probability	0.2	–	–
"Add node" mutation probability	0.2	–	–
Parametrical mutation standard deviation	0.5	1.0	–
Learning rate	–	0.001	0.001
Gamma	–	0.99	0.99
Initial epsilon	–	–	0.1
Epsilon decay	–	–	$3.7 \cdot 10^{-5}$
Generations	500	250 + 250	500
Episode length		12	



(a) Load demand scenarios. The black line represents the test scenario, and the green area represents the range of the noise added to the test scenario when generating training scenarios.



(b) Example of a week in autumn from five sampled training scenarios.

Fig. 15. Train and test load scenarios generation.

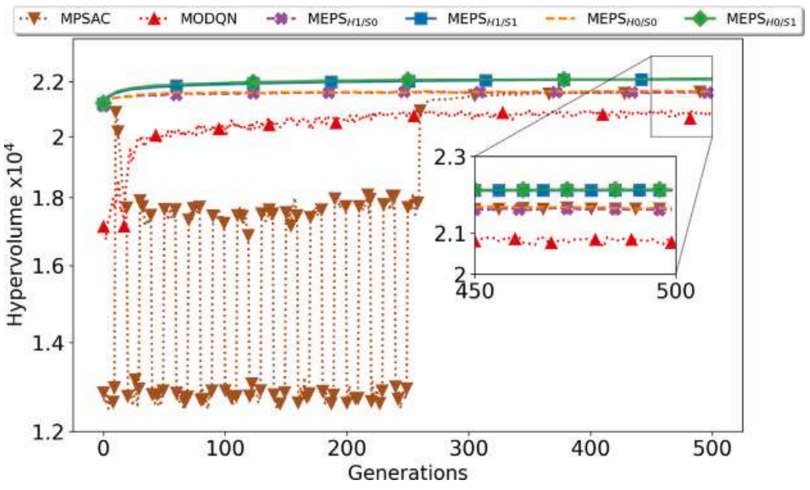


Fig. 16. Average hypervolume values of 20 executions of each algorithm during training.

when generating training scenarios. During training, five different load scenarios are randomly sampled from the green area as shown in Fig. 15(b). The training rewards were the average of the rewards obtained from each one of the five scenarios. Moreover, each algorithm was run 20 times with the same initialization parameters as presented in Table 8 and started with ESS at 20% SoC, in state $s_1 = \{0.2, 0.0, 0.0\}$.

The average hypervolume for the 20 runs of each algorithm's training rewards is presented in Fig. 16. During training, HV values from MPSAC oscillated in the first 250 generations and improved in the last 250, when the MO-CMA-ES counterpart started. On the other hand, although the multi-objective DQNs increased across generations, it only presented a HV value higher than MPSAC in the first 250 generations. After 250 generations, MODQN performance was below the performances of all the other algorithms. MEPS shows the best performance regarding HV, with H1/S1 and H0/S1 achieving the highest HV values at the end of training.

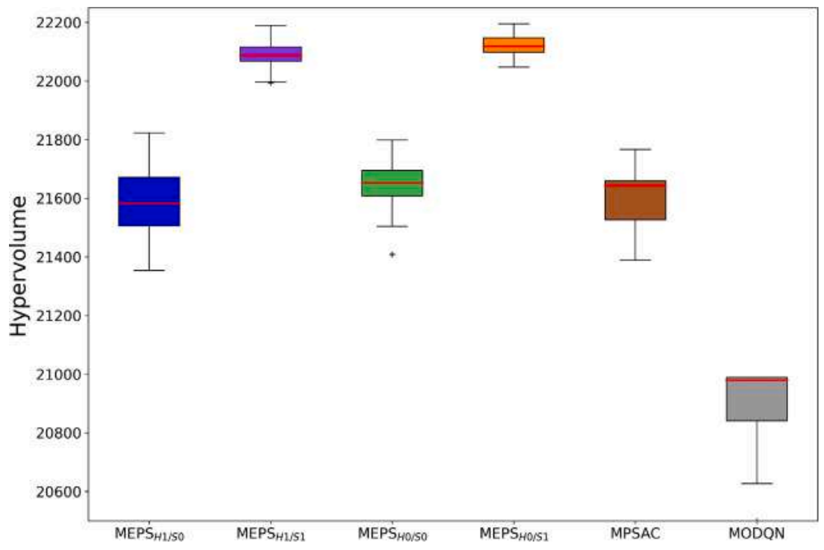


Fig. 17. Boxplot results of the average hypervolumes on test scenario for 20 executions of each algorithm.

Table 9
Performance of each algorithm regarding hypervolume when evaluated in the test scenario.

	Mean	Std.	Worst	Median	Best
MEPS$_{H1/S0}$	21 585.09	126.33	21 353.65	21 582.21	21 823.05
MEPS$_{H1/S1}$	22 090.50	49.18	21 993.69	22 086.85	22 189.52
MEPS$_{H0/S0}$	21 645.97	97.39	21 408.43	21 652.99	21 798.68
MEPS$_{H0/S1}$	**22 120.26**	**44.87**	**22 047.51**	**22 118.48**	**22 195.82**
MPSAC	21 604.24	102.89	21 389.04	21 643.24	21 767.06
MODQN	20 899.15	141.02	20 499.26	20 978.54	20 989.08

Table 10
Ranking of algorithms based on Wilcoxon signed-rank test results using mean hypervolumes in the test scenario.

Rank		
1	2	3
MEPS$_{H1/S1}$	–	–
MEPS$_{H0/S1}$	–	–
–	MEPS$_{H1/S0}$	–
–	MEPS$_{H0/S0}$	–
–	MPSAC	–
–	–	MODQN

Afterwards, each algorithm was evaluated on the test scenario to assess its generalization ability on a scenario never seen before. Table 9 details the test results obtained. It can be seen that, although able to generalize to an unseen scenario, MODQN mean results are not only worse than the results from MEPS and MPSAC, but also present a higher standard deviation. Comparing MPSAC to MEPS versions, MPSAC's results are very similar to H1/S0 results and very close to H0/S0 results. Yet, with respect to H1/S1 and H0/S1, MPSAC's performance is worse in terms of not only mean HV but also standard deviation. Moreover, MPSAC's best HV value is lower than the worst HV values in both H1/S1 and H1/S0. Among MEPS versions, H1/S1 and H1/S0 show very similar performances and present the best values with the smallest variation when compared to MODQN, MPSAC, and other MEPS versions.

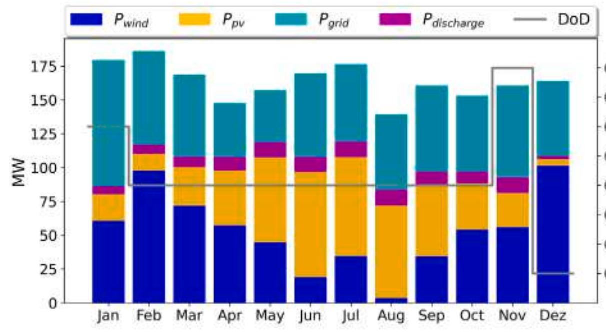
Nevertheless, to provide a robust evaluation of the results obtained, we performed two non-parametric tests. Firstly, we analyze the boxplot behavior, then, similar to Section 5.1, we conducted a two-fold analysis using a Kruskal–Wallis test, followed by a Wilcoxon signed-rank posthoc test with the Holm–Bonferroni correction. Boxplots are a useful tool to analyze the range and distribution of the data, and sometimes, obtain information about the true difference among the means. If the notches in the boxplots do not overlap, it can be concluded, with 95% confidence, that the true means do differ [72]. With this in mind and analyzing Fig. 17, it is possible to conclude that there are differences among the means of the algorithms.

To statically determine the difference in the performance of the evaluated algorithms, a Kruskal–Wallis test with 1% significance level was applied. Thereafter, a Wilcoxon signed-rank test with the Holm-Bonferroni correction was applied to find out, by pairwise comparisons, which specific group's means are different. The Kruskal–Wallis test results corroborated the boxplot analysis attesting that, with a higher confidence level of 99%, there are differences among the mean hypervolume values. After that, from posthoc test results, it is possible to provide a ranking among algorithms as presented in Table 10.

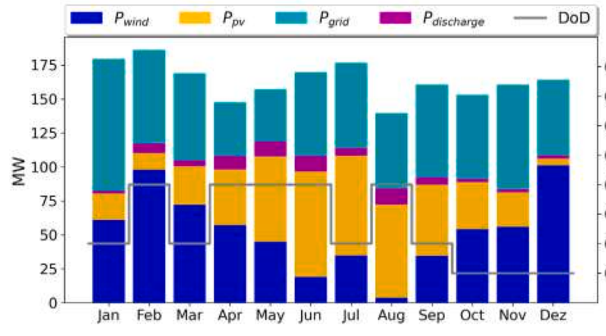
Therefore, the experimental results have shown that MEPS networks are able to provide feasible solutions to the proposed ESS control problem with performance comparable to state-of-the-art MORL techniques. Specifically, H1/S1 and H0/S1 outperformed Deep Q Networks as well as the combination of MO-CMA-ES and Soft Actor-Critic in the proposed multi-objective ERM problem. Furthermore, both H1/S1 and H0/S1 solutions are composed of ANNs with an average of 30 connections and 15 nodes. Compared to MPSAC networks (75 nodes and 776 connections) and MODQN networks (131 nodes and 4936 connections), H1/S1 and H0/S1 solutions comprise much lighter networks. These light networks present a suitable solution to microgrids because they can be deployed as controllers in devices with low computational power.

An important observation regarding MEPS configuration is that, contrary to the benchmark results, the use of hypervolume contribution as the density measure (S1) achieved higher hypervolume values than MEPS counterparts using crowding distance (S0). This indicates that, although benchmarking algorithms using test environments are necessary to validate new techniques, test environments are not globally sufficient to attest a model's performance. In fact, when it comes to real problems such as energy management-based problems, some models are better suited than others.

Next, we analyze the microgrid functioning of both the H1/S1 and H0/S1 solutions. As the multi-objective approach provides alternatives for a decision maker to select knowledge-based solutions, we have employed a multi-criteria decision technique, namely TOPSIS [73], to select a solution that satisfies an equal preference over the three objectives. Fig. 18 illustrates the MG behavior of each solution in a



(a) MG behavior of H1/S1 solution.



(b) MG behavior of H0/S1 solution.

Fig. 18. MG behavior analysis of H1/S1 and H0/S1 solutions selected from TOPSIS with equal preference over the three objectives.

year of operation. The total amount of wind and solar energy consumed per month are indicated by P_{wind} and P_{pv}, respectively. The amount of energy bought from the public grid is indicated by P_{grid}, and the total energy discharged from the battery per month is denoted by $P_{discharge}$.

The more conservative behavior of using lower DoDs in the H0/S1 solution leads to a cost of \$567 194.54, CO_2 emissions of 60.45 tons of CO_2eq./kWh, and a 0.06% ESS degradation per year. In contrast, the H1/S1 solution adopts higher DoD values, specially from September to November when there is less wind and solar radiation. Although allowing a higher use of the ESS, H1/S1 solution leads to both lower cost of \$537 287.03 and lower ESS degradation of 0.04%, but a higher CO_2 emission of 60.66 tons of CO_2eq./kWh.

6. Conclusion

In this paper, we have presented a novel neuroevolutionary MORL algorithm for learning multiple policies in multi-objective environments. The proposed algorithm, the Multi-objective Evolutionary Policy Search (MEPS), searches for deterministic policies parameterized by ANNs in MORL environments. The MEPS is a model-free multi-policy algorithm that belongs to the actor-only family of RL algorithms. Therefore, MEPS estimates preference values for discrete actions and does not depend on gradient updates or value function estimation. Additionally, a survivor selection mechanism has been proposed based on a heavy-tailed distribution that selects solutions from all non-dominated fronts between generations. Accordingly, we conducted a twofold analysis to evaluate the performance of the proposal.

In the first part of the analysis, five different existing benchmark MORL environments, along with a sixth and novel MORL environment, the Discontinuous Deep Sea Treasure (DDST), were selected. In addition, three state-of-the-art MORL algorithms, namely Pareto Q-Learning (PQL), Q-Managed (QM), and Multi-Policy Soft Actor-Critic (MPSAC), were executed and the obtained results were compared with the MEPS results. We conducted an extensive analysis of four versions of MEPS in the benchmark functions, with a focus on not only the hypervolume measure but also the complexity of the networks obtained by MEPS in terms of the number of nodes and connections. Furthermore, MEPS achieved competitive results in all six benchmarks.

In the second part, we presented a novel ERM modeling as a MORL problem, where the agent controls the depth of discharge (DoD) of the ESS present in the grid. The proposed ERM model incorporates a Lithium-Ion degradation model to monthly update the battery capacity, based on the previous DoDs and usage. Subsequently, we assessed the performance of MEPS in this model using real-world time-series data on load and environmental characteristics. Comparing the results to those obtained using MPSAC and multi-objective deep Q networks, we found that MEPS is capable of generalizing better to unseen scenarios and can compete with deep learning-based MORL algorithms. In addition to achieving competitive results, MEPS also generated smaller networks with an average of 30 connections and 14 nodes.

Regarding the neuroevolutionary algorithm proposed, MEPS suffered from some drawbacks. For instance, the different mutation parameters require either a careful selection or a hyperparameter optimization. Additionally, due to its stochasticity, multiple runs should be performed to assess an average result. Besides, the current implementation of MEPS is restricted to problems with discrete action spaces. In spite of the aforementioned drawbacks, a neuroevolutionary algorithm like MEPS proved to be an efficient controller for ESS depth of discharge in MGs. Additionally, future work is to investigate new ways in which crossover can be performed for neuroevolution. In summary, from a perspective of sustainability and energy savings, the results presented in this work are becoming more relevant every day, and represent concrete benefits for the environment in a future where the integration of renewable energy sources is increasing every day.

CRediT authorship contribution statement

G.M.C. Leite: Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **S. Jiménez-Fernández:** Writing – review & editing, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization. **S. Salcedo-Sanz:** Writing – review & editing, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization. **C.G. Marcelino:** Writing – review & editing, Visualization, Supervision, Project administration, Methodology, Investigation, Formal analysis, Conceptualization. **C.E. Pedreira:** Writing – review & editing, Supervision, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization.

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

The authors are unable or have chosen not to specify which data has been used.

Acknowledgments

This work was partially funded by the Brazilian research agencies: CNPq-National Council for Scientific and Technological Development, Brazil (Grant Number 306258/2019-6), FAPERJ- Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro, Brazil (Grant Number E-26/200.840/2021), Ph.D. Scholarship from

CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, Brazil (CAPES-PROEX), and support from CAPES, Brazil (Grant number: 23038.006308/2021-70). This research has also been partially supported by projects PID2020-115454GB-C21 and TED2021-131777B-C22 of Spanish Ministry of Science and Innovation (MICINN). The authors thank UAH and UFRJ for the infrastructure used to conduct this work.

References

[1] A. Mahmoudan, P. Samadof, S. Hosseinzadeh, D.A. Garcia, A multigeneration cascade system using ground-source energy with cold recovery: 3E analyses and multi-objective optimization, Energy 233 (2021) 121185, http://dx.doi.org/10.1016/j.energy.2021.121185.

[2] S. Choudhury, Review of energy storage system technologies integration to microgrid: Types, control strategies, issues, and future prospects, J. Energy Storage 48 (2022) 103966, http://dx.doi.org/10.1016/j.est.2022.103966.

[3] N. Eghbali, S.M. Hakimi, A. Hasankhani, G. Derakhshan, B. Abdi, Stochastic energy management for a renewable energy based microgrid considering battery, hydrogen storage, and demand response, Sustain. Energy Grids Netw. 30 (2022) 100652, http://dx.doi.org/10.1016/j.segan.2022.100652.

[4] IRENA, Renewable Energy Statistics, The international renewable energy agency, abu dhabi, in: Renewable Power Generation Costs in 2019, 2020.

[5] EC, Transport, Commission Outlines Ambitious Plan to Increase Mobility and Reduce Emissions, European Commission, Brussels, Press Release, 2011.

[6] Association internationale pour l'évaluation du rendement scolaire, Global EV Outlook 2016: Beyond One Million Electric Cars, IEA, 2016.

[7] N. Hatziargyriou, H. Asano, R. Iravani, C. Marnay, Microgrids, IEEE Power Energy Mag. 5 (4) (2007) 78–94, http://dx.doi.org/10.1109/MPAE.2007.376583.

[8] L. Luo, S.S. Abdulkareem, A. Rezvani, M.R. Miveh, S. Samad, N. Aljojo, M. Pazhoohesh, Optimal scheduling of a renewable based microgrid considering photovoltaic system and battery energy storage under uncertainty, J. Energy Storage 28 (2020) 101306, http://dx.doi.org/10.1016/j.est.2020.101306.

[9] N.P. Padhy, Unit commitment-a bibliographical survey, IEEE Trans. Power Syst. 19 (2) (2004) 1196–1205, http://dx.doi.org/10.1109/TPWRS.2003.821611.

[10] A. Rezaee Jordehi, An improved particle swarm optimisation for unit commitment in microgrids with battery energy storage systems considering battery degradation and uncertainties, Int. J. Energy Res. 45 (1) (2021) 727–744, http://dx.doi.org/10.1002/er.5867.

[11] L. Alvarado-Barrios, A.R. del Nozal, J.B. Valerino, I.G. Vera, J.L. Martínez-Ramos, Stochastic unit commitment in microgrids: Influence of the load forecasting error and the availability of energy storage, Renew. Energy 146 (2020) 2060–2069, http://dx.doi.org/10.1016/j.renene.2019.08.032.

[12] International Electrotechnical Commission and others, IEC 61970: Energy Management System Application Program Interface (EMS-API), International Electrotechnical Commission (IEC), Geneva, Switzerland, 2004.

[13] S. Salcedo-Sanz, C. Camacho-Gómez, R. Mallol-Poyato, S. Jiménez-Fernández, J. Del Ser, A novel Coral Reefs Optimization algorithm with substrate layers for optimal battery scheduling optimization in micro-grids, Soft Comput. 20 (11) (2016) 4287–4300, http://dx.doi.org/10.1007/s00500-016-2295-7.

[14] M.A. Hossain, H.R. Pota, S. Squartini, A.F. Abdou, Modified PSO algorithm for real-time energy management in grid-connected microgrids, Renew. Energy 136 (2019) 746–757, http://dx.doi.org/10.1016/j.renene.2019.01.005.

[15] M.H. Mostafa, S.H.E.A. Aleem, S.G. Ali, A.Y. Abdelaziz, P.F. Ribeiro, Z.M. Ali, Robust energy management and economic analysis of microgrids considering different battery characteristics, IEEE Access 8 (2020) 54751–54775, http://dx.doi.org/10.1109/ACCESS.2020.2981697.

[16] G.M.C. Leite, C.G. Marcelino, C.E. Pedreira, S. Jiménez-Fernández, S. Salcedo-Sanz, Evaluating the risk of uncertainty in smart grids with electric vehicles using an evolutionary swarm-intelligent algorithm, J. Clean. Prod. 401 (2023) 136775, http://dx.doi.org/10.1016/j.jclepro.2023.136775.

[17] J. Yuan, G. Zhang, S.Y. Samson, Z. Chen, Z. Li, Y. Zhang, A multi-timescale smart grid energy management system based on adaptive dynamic programming and Multi-NN Fusion prediction method, Knowl.-Based Syst. 241 (2022) 108284, http://dx.doi.org/10.1016/j.knosys.2022.108284.

[18] P. Mannion, K. Mason, S. Devlin, J. Duggan, E. Howley, Multi-objective dynamic dispatch optimisation using multi-agent reinforcement learning, in: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems, 2016, pp. 1345–1346.

[19] E. Oh, H. Wang, Reinforcement-learning-based energy storage system operation strategies to manage wind power forecast uncertainty, IEEE Access 8 (2020) 20965–20976, http://dx.doi.org/10.1109/ACCESS.2020.2968841.

[20] F. Liu, Q. Liu, Q. Tao, Y. Huang, D. Li, D. Sidorov, Deep reinforcement learning based energy storage management strategy considering prediction intervals of wind power, Int. J. Electr. Power Energy Syst. 145 (2023) 108608, http://dx.doi.org/10.1016/j.ijepes.2022.108608.

[21] Q. Wu, Q. Feng, Y. Ren, Q. Xia, Z. Wang, B. Cai, An intelligent preventive maintenance method based on reinforcement learning for battery energy storage systems, IEEE Trans. Ind. Inform. 17 (12) (2021) 8254–8264, http://dx.doi.org/10.1109/TII.2021.3066257.

[22] Y. Shang, W. Wu, J. Guo, Z. Ma, W. Sheng, Z. Lv, C. Fu, Stochastic dispatch of energy storage in microgrids: An augmented reinforcement learning approach, Appl. Energy 261 (2020) 114423, http://dx.doi.org/10.1016/j.apenergy.2019.114423.

[23] V.-H. Bui, A. Hussain, H.-M. Kim, Double deep Q-learning-based distributed operation of battery energy storage system considering uncertainties, IEEE Trans. Smart Grid 11 (1) (2019) 457–469, http://dx.doi.org/10.1109/TSG.2019.2924025.

[24] R. Lu, S.H. Hong, X. Zhang, A dynamic pricing demand response algorithm for smart grid: Reinforcement learning approach, Appl. Energy 220 (2018) 220–230, http://dx.doi.org/10.1016/j.apenergy.2018.03.072.

[25] Y. Liu, D. Zhang, H.B. Gooi, Optimization strategy based on deep reinforcement learning for home energy management, CSEE J. Power Energy Syst. 6 (3) (2020) 572–582, http://dx.doi.org/10.17775/CSEEJPES.2019.02890.

[26] S.B. Slama, M. Mahmoud, A deep learning model for intelligent home energy management system using renewable energy, Eng. Appl. Artif. Intell. 123 (2023) 106388, http://dx.doi.org/10.1016/j.engappai.2023.106388.

[27] A. Mathew, M.J. Jolly, J. Mathew, Improved residential energy management system using priority double deep Q-learning, Sustainable Cities Soc. 69 (2021) 102812, http://dx.doi.org/10.1016/j.scs.2021.102812.

[28] J.-H. Syu, G. Srivastava, M. Fojcik, R. Cupek, J.C.-W. Lin, Energy grid management system with anomaly detection and Q-learning decision modules, Comput. Electr. Eng. 107 (2023) 108639, http://dx.doi.org/10.1016/j.compeleceng.2023.108639.

[29] P. Vamplew, J. Yearwood, R. Dazeley, A. Berry, On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts, in: Australasian Joint Conference on Artificial Intelligence, 2008, pp. 372–378, http://dx.doi.org/10.1007/978-3-540-89378-3_37.

[30] A. Pan, W. Xu, L. Wang, H. Ren, Additional planning with multiple objectives for reinforcement learning, Knowl.-Based Syst. 193 (2020) 105392, http://dx.doi.org/10.1016/j.knosys.2019.105392.

[31] F. Song, H. Xing, X. Wang, S. Luo, P. Dai, Z. Xiao, B. Zhao, Evolutionary multi-objective reinforcement learning based trajectory control and task offloading in UAV-assisted mobile edge computing, IEEE Trans. Mob. Comput. (2022) http://dx.doi.org/10.1109/TMC.2022.3208457.

[32] M. Xu, J. Wang, Learning strategy for continuous robot visual control: A multi-objective perspective, Knowl.-Based Syst. 252 (2022) 109448, http://dx.doi.org/10.1016/j.knosys.2022.109448.

[33] J. Chen, H. Xing, Z. Xiao, L. Xu, T. Tao, A DRL agent for jointly optimizing computation offloading and resource allocation in MEC, IEEE Internet Things J. 8 (24) (2021) 17508–17524, http://dx.doi.org/10.1109/JIOT.2021.3081694.

[34] H. Xing, Z. Xiao, R. Qu, Z. Zhu, B. Zhao, An efficient federated distillation learning system for multitask time series classification, IEEE Trans. Instrum. Meas. 71 (2022) 1–12, http://dx.doi.org/10.1109/TIM.2022.3201203.

[35] Y. Li, R. Wang, Z. Yang, Optimal scheduling of isolated microgrids using automated reinforcement learning-based multi-period forecasting, IEEE Trans. Sustain. Energy 13 (1) (2021) 159–169, http://dx.doi.org/10.1109/TSTE.2021.3105529.

[36] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.

[37] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, John Wiley & Sons, 2014.

[38] C.J.C.H. Watkins, Learning from delayed rewards, 1989.

[39] R.S. Sutton, Learning to predict by the methods of temporal differences, Mach. Learn. 3 (1) (1988) 9–44, http://dx.doi.org/10.1007/BF00115009.

[40] P. Dayan, C. Watkins, Q-learning, Mach. Learn. 8 (3) (1992) 279–292, http://dx.doi.org/10.1007/BF00992698.

[41] J.N. Tsitsiklis, Asynchronous stochastic approximation and Q-learning, Mach. Learn. 16 (3) (1994) 185–202, http://dx.doi.org/10.1023/A:1022689125041.

[42] D.J. White, Multi-objective infinite-horizon discounted Markov decision processes, J. Math. Anal. Appl. 89 (2) (1982) 639–647, http://dx.doi.org/10.1016/0022-247X(82)90122-6.

[43] D.M. Roijers, P. Vamplew, S. Whiteson, R. Dazeley, A survey of multi-objective sequential decision-making, J. Artificial Intelligence Res. 48 (2013) 67–113, http://dx.doi.org/10.1613/jair.3987.

[44] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Inc., 2001.

[45] V. Pareto, Cours d'Économie Politique, Vol. 1, Librairie Droz, 1964.

[46] C.G. Marcelino, G.M.C. Leite, E.F. Wanner, S. Jiménez-Fernández, S. Salcedo-Sanz, Evaluating the use of a Net-Metering mechanism in microgrids to reduce power generation costs with a swarm-intelligent algorithm, Energy 266 (2023) 126317, http://dx.doi.org/10.1016/j.energy.2022.126317.

[47] H. Yan, D. Zhang, Qilu, X. Duo, X. Sheng, A review of spinel lithium titanate (Li4Ti5O12) as electrode material for advanced energy storage devices, Ceram. Int. 47 (5) (2021) 5870–5895, http://dx.doi.org/10.1016/j.ceramint.2020.10.241.

[48] C. Marcelino, M. Baumann, L. Carvalho, N. Chibeles-Martins, M. Weil, P. Almeida, E. Wanner, A combined optimisation and decision-making approach for battery-supported HMGS, J. Oper. Res. Soc. 71 (5) (2020) 762–774, http://dx.doi.org/10.1080/01605682.2019.1582590.

[49] UNECE, Life cycle assessment of electricity generation options, 2022, https://unece.org/sed/documents/2021/10/reports/life-cycle-assessment-electricity-generation-options Available in 16th march 2022.

[50] Red Eléctrica, The Spanish Electricity System. Preliminary report 2021, 2022, https://www.ree.es/en/datos/publications/annual-system-report/the-spanish-electricity-system-preliminary-report-2021 Available in 11th January 2022.

[51] D. Roberts, S. Brown, The economics of firm solar power from Li-ion and vanadium flow batteries in California, MRS Energy Sustain. 9 (2022) 129–141, http://dx.doi.org/10.1557/s43581-022-00028-w.

[52] M. Musallam, C.M. Johnson, An efficient implementation of the rainflow counting algorithm for life consumption estimation, IEEE Trans. Reliab. 61 (4) (2012) 978–986, http://dx.doi.org/10.1109/TR.2012.2221040.

[53] A. Maheshwari, N.G. Paterakis, M. Santarelli, M. Gibescu, Optimizing the operation of energy storage using a non-linear lithium-ion battery degradation model, Appl. Energy 261 (2020) 114360, http://dx.doi.org/10.1016/j.apenergy.2019.114360.

[54] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197, http://dx.doi.org/10.1109/4235.996017.

[55] M. Emmerich, N. Beume, B. Naujoks, An EMO algorithm using the hypervolume measure as selection criterion, in: International Conference on Evolutionary Multi-Criterion Optimization, 2005, pp. 62–76, http://dx.doi.org/10.1007/978-3-540-31880-4_5.

[56] K.O. Stanley, R. Miikkulainen, Evolving neural networks through augmenting topologies, Evol. Comput. 10 (2) (2002) 99–127, http://dx.doi.org/10.1162/106365602320169811.

[57] S. Sarti, G. Ochoa, A NEAT visualisation of neuroevolution trajectories, in: International Conference on the Applications of Evolutionary Computation (Part of EvoStar), 2021, pp. 714–728, http://dx.doi.org/10.1007/978-3-030-72699-7_45.

[58] C. Yue, P.N. Suganthan, J. Liang, B. Qu, K. Yu, Y. Zhu, L. Yan, Differential evolution using improved crowding distance for multimodal multiobjective optimization, Swarm Evol. Comput. 62 (2021) 100849, http://dx.doi.org/10.1016/j.swevo.2021.100849.

[59] B.C. Arnold, Pareto distribution, in: Wiley StatsRef: Statistics Reference Online, 2014, pp. 1–10, http://dx.doi.org/10.1002/9781118445112.stat01100.pub2.

[60] C.G. Marcelino, G.M.C. Leite, C.A.D.M. Delgado, L.B. de Oliveira, E.F. Wanner, S. Jiménez-Fernández, S. Salcedo-Sanz, An efficient multi-objective evolutionary approach for solving the operation of multi-reservoir system scheduling in hydro-power plants, Expert Syst. Appl. 185 (2021) 115638, http://dx.doi.org/10.1016/j.eswa.2021.115638.

[61] C.F. Hayes, R. Rădulescu, E. Bargiacchi, J. Källström, M. Macfarlane, M. Reymond, T. Verstraeten, L.M. Zintgraf, R. Dazeley, F. Heintz, E. Howley, A.A. Irissappane, P. Mannion, A. Nowé, G. Ramos, M. Restelli, P. Vamplew, D.M. Roijers, A practical guide to multi-objective reinforcement learning and planning, Auton. Agents Multi-Agent Syst. 36 (1) (2022) 1–59, http://dx.doi.org/10.1007/s10458-022-09552-y.

[62] M.T. Jensen, Reducing the run-time complexity of multiobjective EAs: The NSGA-II and other algorithms, IEEE Trans. Evol. Comput. 7 (5) (2003) 503–515, http://dx.doi.org/10.1109/TEVC.2003.817234.

[63] A. McIntyre, M. Kallada, C.G. Miguel, C. Feher de Silva, M.L. Netto, neat-python, 2019, https://github.com/CodeReclaimers/neat-python.

[64] P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, E. Dekker, Empirical evaluation methods for multiobjective reinforcement learning algorithms, Mach. Learn. 84 (1) (2011) 51–80, http://dx.doi.org/10.1007/s10994-010-5232-5.

[65] K. Van Moffaert, A. Nowé, Multi-objective reinforcement learning using sets of Pareto dominating policies, J. Mach. Learn. Res. 15 (107) (2014) 3663–3692.

[66] T.H.F. de Oliveira, L.P. de Souza Medeiros, A.D.D. Neto, J.D. Melo, Q-Managed: A new algorithm for a multiobjective reinforcement learning, Expert Syst. Appl. 168 (2021) 114228, http://dx.doi.org/10.1016/j.eswa.2020.114228.

[67] D. Chen, Y. Wang, W. Gao, Combining a gradient-based method and an evolution strategy for multi-objective reinforcement learning, Appl. Intell. 50 (10) (2020) 3301–3317, http://dx.doi.org/10.1007/s10489-020-01702-7.

[68] P. Vamplew, R. Dazeley, C. Foale, Softmax exploration strategies for multiobjective reinforcement learning, Neurocomputing 263 (2017) 74–86, http://dx.doi.org/10.1016/j.neucom.2016.09.141.

[69] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: Empirical results, Evol. Comput. 8 (2) (2000) 173–195, http://dx.doi.org/10.1162/106365600568202.

[70] T.T. Nguyen, N.D. Nguyen, P. Vamplew, S. Nahavandi, R. Dazeley, C.P. Lim, A multi-objective deep reinforcement learning framework, Eng. Appl. Artif. Intell. 96 (2020) 103915, http://dx.doi.org/10.1016/j.engappai.2020.103915.

[71] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.

[72] C.G. Marcelino, G.M.C. Leite, S. Jiménez-Fernández, S. Salcedo-Sanz, An improved C-DEEPSO algorithm for optimal active-reactive power dispatch in microgrids with electric vehicles, IEEE Access 10 (2022) 94298–94311, http://dx.doi.org/10.1109/ACCESS.2022.3203728.

[73] G.-H. Tzeng, J.-J. Huang, Multiple Attribute Decision Making: Methods and Applications, CRC Press, 2011.