Implementation details of reinforcement learning based real-time power flow management in microgrids

Rima Oulhaj, Pierre Garambois, and Lionel Roucoules

Laboratoire d'Ingénierie des Systèmes Physiques et Numériques, Arts et Métiers Institute of Technology, HESAM Université
{rima.oulhaj, pierre.garambois, lionel.roucoules}@ensam.eu

Abstract. Real-time Power flow Management (RPM) in electric microgrids is a sequential decision making problem that can be formulated as a discounted infinite-horizon Markov Decision Process (MDP). Policy gradient methods have shown great results in solving discounted infinite-horizon MDPs in a wide range of applications. This paper provides implementation details of actor-critic policy gradient algorithms for RPM. We present the microgrid model, the MDP formulation of the RPM problem and the different test environments (i.e., instances of microgrid sizing choices and reward function settings, denoted Environment 1, 2, 3 and 4). We provide the hyperparameters of the algorithms and the algorithm optimizations used that are recommended in the literature.

1 Microgrid model

The microgrid (MG) is modeled as the concatenation of devices that either inject or draw power into/from the MG. At each time step, power flows inside the microgrid are balanced and verify the following equation:

$$\forall t: \quad P_t^L + P_t^{Ren} + P_{out,t}^G + P_t^B + P_t^U = 0 \tag{1}$$

Where P_t^L is the power load which verifies $P_t^L \leq 0$. P_t^{Ren} is the renewable generation given by $P_t^{Ren} = P_t^{PV} + P_t^{WT}$ where P_t^{PV} and P_t^{WT} are the photovoltaic (PV) panel and wind turbine (WT) generation respectively with $P_t^{PV} \geq 0$ and $P_t^{WT} \geq 0$. $P_{out,t}^G$ is the total generation of a set G of controllable generators given by $P_{out,t}^G = \sum_{g \in G} P_{out,t}^g$ where $\forall g, P_{out,t}^g \geq 0$. P_t^B is the equivalent power flow of a set B of batteries given by $P_t^B = \sum_{b \in B} P_t^b$ where $\forall b \in B, P_t^b \leq 0$ in the charge mode and $P_t^b \geq 0$ in the discharge mode. P_t^U is the utility grid power flow that is responsible of balancing the power flows inside the microgrid.

1.1 Controllable generation model

The power output $P_{out,t}^g$ of each controllable generator $g \in G$ at time step t is given by:

$$P_{out,t}^g = \min\left(P_N^g \cdot \mu_g, (\alpha_g \cdot \Delta t + P_{out,t-1}^g) \cdot \mu_g, P_t^g\right) \tag{2}$$

Where Δt is the time step length in $[h]$ and for each $g \in G$, P_t^g is the power generation command signal (the demanded power), P_N^g is the nominal power output, μ_g is the efficiency coefficient, α_g is given by $\alpha_g = \dfrac{P_N^g}{\Delta T_g}$ and ΔT_g is the response time in $[h]$.

1.2 Battery model

For each battery $b \in B$, let W_t^b be the stored energy in $[Wh]$ at time step t:

$$W_t^b = W_{t-1}^b \cdot (1 - \sigma_{SD}^b) + (P_{C,t}^b \cdot \eta_C^b - \frac{P_{D,t}^b}{\eta_D^b}) \cdot \Delta t \tag{3}$$

Where W_{t-1}^b is the stored energy from the previous time step, σ_{SD}^b is the self-discharge rate, η_C^b and η_D^b are the charge and discharge efficiencies respectively, $P_{C,t}^b$ and $P_{D,t}^b$ are the charge and discharge power respectively given by $P_{C,t}^b = -\min(0, P_t^b)$, $P_{C,t}^b \geq 0$ and $P_{D,t}^b = \max(0, P_t^b)$, $P_{D,t}^b \geq 0$. For each $b \in B$ we define the state of charge (SOC) at each time step t as:

$$SOC_t^b = \frac{W_t^b}{W_{NC}^b} \tag{4}$$

Where W_{NC}^b is the nominal capacity (NC) of $b \in B$ in $[Wh]$. The SOC must verify the following constraint at each time step:

$$SOC_{min}^b \leq SOC_t^b \leq SOC_{max}^b \tag{5}$$

Where SOC_{min}^b and SOC_{max}^b are the minimum and maximum SOC respectively and are given characteristics of battery b.

1.3 Wind Turbine Model

We use a parametric model for wind turbine power curves that incorporates wind speed data [1]. Wind power production P_t^{WT} can be calculated as a function of the wind speed $V_{W,t}$, air density ρ, rotor area A_{rotor} and power coefficient $C_p(\lambda_t, \beta)$ with λ_t being the tip-speed ratio and β the blade pitch angle.

$$P_t^{WT} = \frac{1}{2}\rho \cdot A_{rotor} \cdot (V_{W,t})^3 \cdot C_p(\lambda_t, \beta) \tag{6}$$

The power coefficient $C_p(\lambda_t, \beta)$ is given by:

$$C_p(\lambda_t, \beta) = c_1(\frac{c_2}{\lambda_i} - c_3\beta - c_4\lambda_i\beta - c_5\beta^{\chi} - c_6)e^{-c_7/\lambda_i} + c_8\lambda_t \tag{7}$$

Where:

$$\lambda_i^{-1} = (\lambda_t + c_9\beta)^{-1} - c_{10}(\beta^3 + 1)^{-1} \tag{8}$$

Parameter	β	c_1	c_2	c_3	c_4	c_5	χ	c_6	c_7	c_8	c_9	c_{10}
Value	0	0.22	120	0.4	0	0	0	5	12.5	0	0.08	0.035

Table 1. Parameters setting for the power coefficient C_p [1]

The tip-speed ratio λ_t can be formulated as a function of the rotor's radius $D_{rotor}/2$ and rotational speed ω_t as well as the wind speed $V_{W,t}$:

$$\lambda_t = \frac{\omega_t \cdot (D_{rotor}/2)}{V_{W,t}} \tag{9}$$

Where:

$$\omega_t = \min\left(\omega_{max}, \max(\omega_{min}, \frac{\lambda_{opt}}{D_{rotor}/2} \cdot V_{W,t})\right) \tag{10}$$

And:

$$\lambda_{opt} = \text{argmax}_{\lambda,\beta=0}\, C_p(\lambda, \beta) \tag{11}$$

Where ω_{min} and ω_{max} are the minimum and maximum rotor rotational speed respectively. We use the parameters in table 1 for equations 7 and 8. Using this model, the power curve is scaled by the nominal power of the turbine, and then the cut-in and cut-off wind speeds are applied [1].

1.4 PV panel model

The power output P_t^{PV} of the PV module is given by:

$$P_t^{PV} = A^{PV} * \mu^{PV} * I_r(t) \tag{12}$$

Where A^{PV} is the surface area of the PV module, μ^{PV} is the efficiency coefficient and $I_r(t)$ is the global horizontal solar irradiance during time step t.

1.5 Controllable power flows constraints

We define the controllable power flows constraints as the minimum and maximum bounds of the demanded generation and the storage power flows. These bounds can be written as:

$$\forall g, \forall t: \ 0 \leq P_t^g \leq P_{max,t}^g \tag{13}$$

$$\forall b, \forall t: \ P_{min,t}^b \leq P_t^b \leq P_{max,t}^b \tag{14}$$

Eq. 2 yields:

$$P_{max,t}^g = \min\left(P_N^g \cdot \mu_g, (\alpha_g \cdot \Delta t + P_{out,t-1}^g) \cdot \mu_g\right) \tag{15}$$

Eqs. 3-4-5 yield:

$$P^b_{min,t} = -P^b_{C_{max},t} \tag{16}$$

And:

$$P^b_{max,t} = -\max(P^b_{C_{min},t}, -P^b_{D_{max},t}) \tag{17}$$

$P^b_{C_{max},t}$ (eq. 18) is the maximum possible charge power at t which must verify the maximum SOC constraint, with $P^b_{C_{max},t} \geq 0$. $P^b_{D_{max},t}$ (eq. 19) is the maximum possible discharge power at t which must verify the minimum SOC constraint, with $P^b_{D_{max},t} \geq 0$. $P^b_{C_{min},t}$ (eq. 20) is the minimum charge power needed to maintain SOC^b_t above SOC^b_{min} at all times, i.e., to compensate for the self-discharge when necessary, in which case $P^b_{C_{min},t} \geq 0$, otherwise it is set to $-\infty$.

$$P^b_{C_{max},t} = \max\left(0, \frac{SOC^b_{max} \cdot W^b_{NC} - W^b_{t-1} \cdot (1 - \sigma^b_{SD})}{\Delta t \cdot \eta^b_C}\right) \tag{18}$$

$$P^b_{D_{max},t} = \max\left(0, \frac{W^b_{t-1} \cdot (1 - \sigma^b_{SD}) - SOC^b_{min} \cdot W^b_{NC}}{\Delta t} \cdot \eta^s_D\right) \tag{19}$$

$$P^b_{C_{min},t} = \begin{cases} P^b_{SOC_{min},t} & \text{if} \quad W^b_{t-1} \cdot (1 - \sigma^b_{SD}) < SOC^b_{min} \cdot W^b_{NC} \\ -\infty & \text{otherwise} \end{cases} \tag{20}$$

$P^b_{SOC_{min},t}$ is the charge power necessary to maintain the energy level above $(SOC^b_{min} \cdot W^b_{NC})$ when the self-discharge alone is sufficient to violate this constraint. We calculate it as:

$$P^b_{SOC_{min},t} = \frac{SOC^b_{min} \cdot W^b_{NC} - W^b_{t-1} \cdot (1 - \sigma^b_{SD})}{\Delta t \cdot \eta^b_C} \tag{21}$$

2 Power loads, meteorological and energy market data

We use real power loads data, meteorological data (global horizontal solar irradiance and wind speed) and energy market data to simulate the RPM using the infinite horizon MDP formulation. This means that values of power loads P^L_t (eq. 1), global horizontal solar irradiance $I_r(t)$ (eq. 12), wind speed $V_{W,t}$ (eq. 6), grid prices and feed-in tariffs C^L_t and C^F_t resp. (eq. 28) are coming from our data collection.

The power loads data are provided by the Ecole Nationale Supérieure d'Arts et Métiers (Aix-en-Provence campus). Meteorological data are extracted using the ODRE [1] open-source API. Load costs data are extracted from the Bourses de l'électricité Spot France website[2]. The feed-in tariffs are calculated by multiplying the load costs by a coefficient (set to 0.2). Using this data and the microgrid model in section 1, the obtained PV and WT generation profiles are shown in figures 1 and 2 respectively. The power load profiles, feed-in tariffs and load costs data are shown in figures 3, 4 and 5 respectively.

[1] https://opendata.reseaux-energies.fr/

[2] https://www.services-rte.com/fr/visualisez-les-donnees-publiees-par-rte/bourses-de-lelectricite-spot-france.html

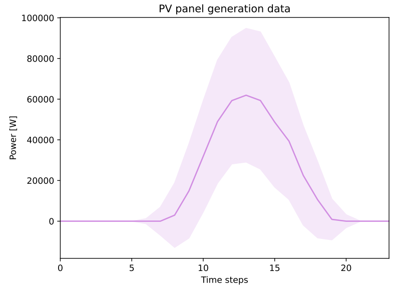

Fig. 1. PV panel generation data

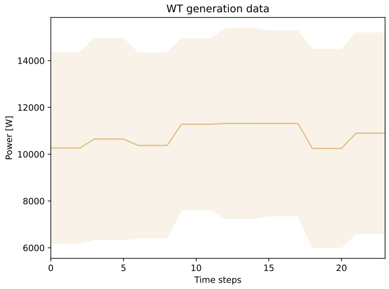

Fig. 2. WT generation data

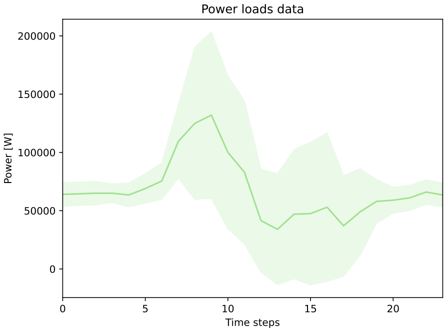

Fig. 3. Power loads data

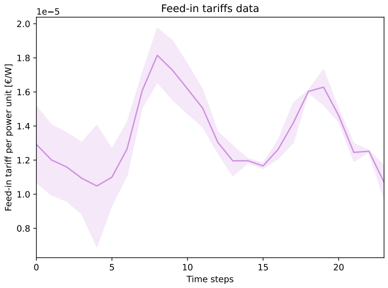

Fig. 4. Feed-in tariffs data

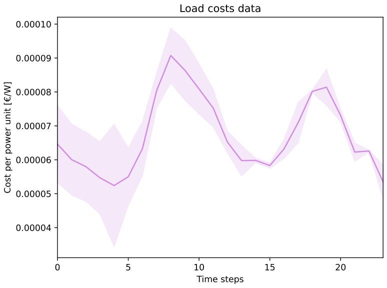

Fig. 5. Load costs data

3 The infinite-horizon MDP formulation of the RPM problem

3.1 State space

The state space $\mathcal{S}$ is continuous and given by:

$$\forall t \; [s_t \in \mathcal{S}] \Leftrightarrow s_t = [P_t^{Ren}, P_t^L, s_t^G, s_t^B, C_t^L, C_t^F] \tag{22}$$

Where $s_t^G = (P^g_{max,t})_{g \in G}$ and $s_t^B = (P^b_{min,t}, P^b_{max,t})_{b \in B}$. C_t^L and C_t^F are the load cost and the feed-in tariffs in $[€/Wh]$.

3.2 Action space

The action space is continuous and given by:

$$\mathcal{A} = \prod_{g \in G} [0,1] \times \prod_{b \in B} [-1,1] \tag{23}$$

Where:

$$a_t \in \mathcal{A} \Leftrightarrow a_t = ((a_t^g)_{g \in G} \ , \ (a_t^b)_{b \in B}) \tag{24}$$

$$\forall t \leq T, \ \forall b \in B, \ \forall g \in G: \ \ 0 \leq a_t^g \leq 1, \ -1 \leq a_t^b \leq 1$$

And:

$$\forall t \leq T, \forall g \in G: \ P_t^g = a_t^g \cdot P^g_{max,t} \tag{25}$$

$$\forall t, \leq T, \forall b \in B: \ P_t^b = \begin{cases} a_t^b \cdot P^b_{max,t} & if \quad a_t^b \geq 0 \\ -a_t^b \cdot P^b_{min,t} \ otherwise \end{cases} \tag{26}$$

3.3 Different reward functions

Economic reward function The normalized economic reward function is given by:

$$\hat{R}^E(s_t, a_t) = \frac{R^E(s_t, a_t) - R^E_{min}}{R^E_{max} - R^E_{min}} \tag{27}$$

Where R^E is defined as:

$$R^E(s_t, a_t) = C_t^F \cdot \max(0, P_t^U) + C_t^L \cdot \min(0, P_t^U) \tag{28}$$

The utility grid power flow P_t^U is given by eq. 1. R^E_{min} and R^E_{max} are empirical bounds of R^E that are computed using our training data and are given by $R^E_{min} = C^L_{max} * P^U_{min}$ and $R^E_{max} = C^F_{max} * P^U_{max}$ with:

$$P^U_{min} = P^L_{min} - \sum_{b \in B} SOC^b_{max} * W^b_{NC} \tag{29}$$

$$P^U_{max} = P^{PV}_{max} + P^{WT}_{max} + \sum_{g \in G} P^g_N + \sum_{b \in B} SOC^b_{max} * W^b_{NC} \tag{30}$$

Values $C^L_{max}, C^F_{max}, P^{PV}_{max}, P^{WT}_{max}$ and P^L_{min} are the maximum load cost, maximum feed-in tariff, maximum PV generation, maximum WT generation and minimum power load (in negative values) respectively that are derived from our training data.

Rule-based reward function This reward function is used for the particular case where the microgrid consists only of renewable energy and a battery ($G = \emptyset$, $B = \{b\}$).
Let $\mathcal{S}_1 = \{s_t \in \mathcal{S} |\ \delta_t > 0\}$, $\mathcal{S}_2 = \{s_t \in \mathcal{S} |\ \delta_t < 0\}$ and $\mathcal{S}_3 = \{s_t \in \mathcal{S} |\ \delta_t = 0\}$ where $\delta_t = P_t^{Ren} - P_t^L$. We define $\mathcal{S}_{11}$ and $\mathcal{S}_{12}$ as: $\mathcal{S}_{11} = \{s_t \in \mathcal{S}_1 |\ \delta_t \geq -P_{min,t}^b\}$, $\mathcal{S}_{12} = \{s_t \in \mathcal{S}_1 |\ \delta_t < -P_{min,t}^b\}$.
A partition of $\mathcal{S}$ is therefore given by $\{\mathcal{S}_{11}, \mathcal{S}_{12}, \mathcal{S}_2, \mathcal{S}_3\}$.
Sets $\{\mathcal{A}_{11}, \mathcal{A}_{12}\}$, $\{\mathcal{A}_{21}, \mathcal{A}_{22}, \mathcal{A}_{23}\}$ and $\{\mathcal{A}_{31}, \mathcal{A}_{32}\}$ are partitions of $\mathcal{A}$ where:

$$\mathcal{A}_{11} = \{a_t \in \mathcal{A} | P_t^b \geq 0\}, \mathcal{A}_{12} = \{a_t \in \mathcal{A} | P_t^b < 0\} \tag{31}$$

$$\mathcal{A}_{21} = \{a_t \in \mathcal{A} | P_t^b > 0 \wedge P_t^b < -\delta_t\}, \mathcal{A}_{22} = \{a_t \in \mathcal{A} | P_t^b > 0 \wedge P_t^b \geq -\delta_t\}, \mathcal{A}_{23} = \{a_t \in \mathcal{A} | P_t^b \leq 0\} \tag{32}$$

$$\mathcal{A}_{31} = \{a_t \in \mathcal{A} | P_t^b = 0\}, \mathcal{A}_{32} = \{a_t \in \mathcal{A} | P_t^b \neq 0\} \tag{33}$$

The rule based reward is defined as:

$$\begin{aligned} R^{rule}(s_t, a_t) = & \sum_{\mathcal{S}_{xx} \in \{\mathcal{S}_{11}, \mathcal{S}_{12}\}} \left[\sum_{\mathcal{A}_{yy} \in \{\mathcal{A}_{11}, \mathcal{A}_{12}\}} 1_{\mathcal{S}_{xx}}(s_t) * 1_{\mathcal{A}_{yy}}(a_t) * r_{\mathcal{S}_{xx} \times \mathcal{A}_{yy}}(s_t, a_t) \right] \\ & + \sum_{\mathcal{A}_{yy} \in \{\mathcal{A}_{21}, \mathcal{A}_{22}, \mathcal{A}_{23}\}} 1_{\mathcal{S}_2}(s_t) * 1_{\mathcal{A}_{yy}(a_t)} * r_{\mathcal{S}_2 \times \mathcal{A}_{yy}}(s_t, a_t) \\ & + \sum_{\mathcal{A}_{yy} \in \{\mathcal{A}_{31}, \mathcal{A}_{32}\}} 1_{\mathcal{S}_3}(s_t) * 1_{\mathcal{A}_{yy}(a_t)} * r_{\mathcal{S}_3 \times \mathcal{A}_{yy}}(s_t, a_t) \end{aligned} \tag{34}$$

$$r_{\mathcal{S}_{11} \times \mathcal{A}_{12}}(s_t, a_t) = r_{\mathcal{S}_2 \times \mathcal{A}_{21}}(s_t, a_t) = \begin{cases} -\dfrac{P_t^b}{\delta_t} & \text{if} \quad \delta_t \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{35}$$

$$r_{\mathcal{S}_{12} \times \mathcal{A}_{12}}(s_t, a_t) = \frac{P_t^b}{P_{min,t}^b} \tag{36}$$

$$r_{\mathcal{S}_2 \times \mathcal{A}_{22}}(s_t, a_t) = \frac{1}{1 + |P_t^b + \delta_t|} \tag{37}$$

$$r_{\mathcal{S}_3 \times \mathcal{A}_{31}}(s_t, a_t) = 1 \tag{38}$$

$$r_{\mathcal{S}_{11} \times \mathcal{A}_{11}}(s_t, a_t) = r_{\mathcal{S}_{12} \times \mathcal{A}_{11}}(s_t, a_t) = r_{\mathcal{S}_2 \times \mathcal{A}_{23}}(s_t, a_t) = r_{\mathcal{S}_3 \times \mathcal{A}_{32}}(s_t, a_t) = 0 \tag{39}$$

4 Test cases: reinforcement learning environments

4.1 Microgrid sizing

Microgrid A: a microgrid with controllable generators Microgrid A includes two controllable generators denoted 'ctrl_gen_1' and 'ctrl_gen_2 (table 2), two batteries denoted 'battery_1' and 'battery_2' (table 3), a PV panel (table 5) and a WT (table 6). The variation interval of the wind turbine parameters can be found in [1].

Device label	P_N^g	μ_g	ΔT_g
Ctrl_gen_1	$45000\ W$	0.99	$1H$
Ctrl_gen_2	$15000\ W$	0.89	$1H$

Table 2. Controllable generator parameters — Microgrid A

Device label	σ_{SD}^b	η_C^b	η_D^b	W_{NC}^b	SOC_{min}^b	SOC_{max}^b
Battery_1	$0,0239$	$0,97$	$0,82$	$70000\ Wh$	$0,1$	$0,8$
Battery_2	$0,0239$	$0,97$	$0,89$	$60000\ Wh$	$0,1$	$0,93$

Table 3. Battery parameters — Microgrid A

Microgrid B: the renewables-storage microgrid This microgrid only includes renewable generation (tables 5 and 6) and a battery (table 4).

σ_{SD}^b	η_C^b	η_D^b	W_{NC}^b	SOC_{min}^b	SOC_{max}^b
$0,0239$	$0,97$	$0,82$	$90000\ Wh$	$0,1$	$0,9$

Table 4. Battery parameters — Microgrid B

A^{PV}	μ^{PV}
$700\ m^2$	$\mu^{PV} = 0.18$

Table 5. PV panel parameters — Microgrids A and B

D_{rotor}	ω_{min}	ω_{max}	Nominal power	Cut-in wind speed	Cut-out wind speed
100 m	aD_{rotor}^{b} $a = 1046, 558$ $b = -1, 0911$	cD_{rotor}^{d} $c = 705, 406$ $d = -0, 8349$	20000 W	3 ms^{-1}	25 ms^{-1}

Table 6. Wind turbine parameters — Microgrids A and B

4.2 Environments

Table 7 describes a set of environments each composed of a microgrid and a choice of reward function.

Environment label	Microgrid	Reward function
Environment 1	Microgrid A	$\hat{R}_E$
Environment 2	Microgrid B	$\hat{R}_E$
Environment 3	Microgrid B	$\alpha_E \cdot \hat{R}_E + \alpha_R \cdot R^{rule}$ $\alpha_E = 0.25$ $\alpha_R = 0.75$
Environment 4	Microgrid B	R^{rule}

Table 7. Test environments of the RL based RPM

5 Deep RL algorithms

5.1 Algorithms VPG, TRPO and PPO

Algorithms 1, 2 and 3 provide the implementation of three actor-critic policy gradient based algorithms which work by consecutive evaluations and updates of the actor parameters ω based on the objective functions J^{VPG} (eq. 40), J^{TRPO} (eq. 43) and J^{PPO} (eq. 46) respectively.

$$J^{VPG}(\omega) = \hat{\mathbb{E}}\left[\log(\pi_\omega(s,a))\hat{A}^{\pi_\omega}(s,a)\right] \tag{40}$$

$$\hat{A}^{\pi_\omega}(s_t, a_t) = \zeta_t + (\gamma\lambda)\zeta_{t+1} + ... + (\gamma\lambda)^{T-t+1}\zeta_{T-1} \qquad (41)$$

$$\zeta_t = r_t + \gamma Q^{\pi_\omega}(s_{t+1}, a_{t+1}) - Q^{\pi_\omega}(s_t, a_t)$$

$$Q^{\pi_\omega}(s, a) = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t = s, a_t = a, \pi_\omega] \qquad (42)$$

Where $\gamma \in]0, 1]$ is the discount factor and $\lambda \in]0, 1]$ is the Generalized Advantage Estimation (GAE) parameter.

$$J^{TRPO}(\omega) = \hat{\mathbb{E}}\left[r_\omega(s, a)\hat{A}^{\pi_\omega}(s, a) - \beta \cdot d_\omega(s, .)\right] \qquad (43)$$

$$d_\omega(s, .) = KL[\pi_\omega(s, .), \pi_{\omega_{old}}(s, .)] \qquad (44)$$

$$r_\omega(s, a) = \frac{\pi_\omega(s, a)}{\pi_{\omega_{old}}(s, a)} \qquad (45)$$

$$J^{CLIP}(\omega) = \hat{\mathbb{E}}[\min(r_\omega(s, a)\hat{A}^{\pi_\omega}(s, a), \mathrm{c}_\omega^\epsilon(s, a)\hat{A}^{\pi_\omega}(s, a))] \qquad (46)$$

Two neural networks are used to approximate π_ω and Q^{π_ω}. The old actor $\pi_{\omega_{old}}$ is intended to keep the actor from diverging, ω_{old} being the matrix of the actor's parameters before the update. Please note that since the actor and critic networks are trained in parallel, for each training epoch $k \in 0, 1, 2, ..., N-1$ we have $Q^{\pi_{\omega_k}} = Q^{\theta_k}$ in equation 42.

5.2 Hyper-parameters of algorithms VPG, TRPO and PPO

Table 8 provides the hyperparameters of algorithms 1, 2 and 3. We use the Adam optimizer in these algorithms in the actor and critic update steps. Following recommendations from [2], we use the non-stationary Adam i.e., we set the parameters β_1 and β_2 of the Adam optimizer to the same value.

5.3 The actor and critic networks

Figures 6 and 7 give the layouts of the actor and critic networks. Since the output of the actor network for a given state s is the mean and standard deviation of $\pi_\omega(s, .)$, we use appropriate activation functions for the output layer (using sigmoid and tanh functions) in order to generate mean and standard deviation vectors that match our action space $\mathcal{A} = \prod_{g\in G}[0, 1] \times \prod_{b\in B}[-1, 1]$. An important implementation detail is that we use the truncated multivariate normal instead of the multivariate normal distribution to sample actions. Indeed, using the normal distribution introduces the possibility of infeasible actions which we want to omit.

Algorithm 1 NPG

```
Require: Initial actor network parameters $\omega_0$
Require: Initial critic network parameters $\theta_0$
Require: Trajectory length $T$
Require: Number of trajectories per training epoch $M$
Require: Number of training epochs $N$
Require: Hyper-parameters $\lambda$ and $\gamma$
  for $k \in 0, 1, 2, ..., N-1$ do
    Sample $M$ trajectories $(s_0^m, a_0^m, r_0^m, s_1^m, a_1^m, r_1^m, ..., s_{T-1}^m, a_{T-1}^m, r_{T-1}^m)_{m\in 1,...M}$ using the actor network $\pi_{\omega_k}$
    Evaluate the $Q$-value function at each $t$ and for each $m$ using the critic network $Q^{\theta_k}$ to compute values $\big(Q^{\theta_k}(s_t^m, a_t^m)_{t\in 0,...,T-1}\big)_{m\in 1,...,M}$
    Compute the actor loss function
      $$J^{VPG}(\omega_k) = \frac{1}{T \cdot M} \sum_{t=0}^{T-1} \sum_{m=1}^{M} \Big[\log(\pi\omega_k(s_t^m, a_t^m))\hat{A}^{\pi_{\omega_k}}(s_t^m, a_t^m)\Big]$$
    Update the actor network
      $$\omega_{k+1} \leftarrow \text{argmax}_{\omega_k} \; J^{VPG}(\omega_k)$$
    Compute the critic loss function
      $$L^Q(\theta_k) = \frac{1}{(T-1) \cdot M} \sum_{t=0}^{T-2} \sum_{m=1}^{M} \Big[Q^{\theta_k}(s_t^m, a_t^m) - r_t^m - \gamma Q^{\theta_k}(s_{t+1}^m, a_{t+1}^m)\Big]^2$$
    Update the critic network $\theta_{k+1} \leftarrow \text{argmin}_{\theta_k} \; L^Q(\theta_k)$
  end for
```

Hyperparameter	Value
Trajectory length T	24
Number of trajectories M	30
Number of training epochs N	$3 \cdot 10^4$
Discount factor γ	0.99
GAE parameter λ (eq. 41)	0.95
PPO clipping parameter ϵ (eq. 46)	0.2
TRPO β coefficient (eq. 43)	1
Adam learning rate	10^{-4}
Adam betas β_1 and β_2	0.9 and 0.9

Table 8. Hyperparameters of VPG, TRPO and PPO (algorithms 1, 2 and 3 respectively).

Algorithm 2 TRPO

```
Require: Initial actor network parameters $\omega_0$
Require: Initial critic network parameters $\theta_0$
Require: Trajectory length $T$
Require: Number of trajectories per training epoch $M$
Require: Number of training epochs $N$
Require: Hyper-parameters $\lambda$, $\gamma$ and $\beta$
  Initialize old actor: $\omega_{old} \leftarrow \omega_0$
  for $k \in 0, 1, 2, ..., N-1$ do
    Sample $M$ trajectories $(s_0^m, a_0^m, r_0^m, s_1^m, a_1^m, r_1^m, ..., s_{T-1}^m, a_{T-1}^m, r_{T-1}^m)_{m \in 1,...M}$ using the actor network $\pi_{\omega_k}$
    Evaluate the $Q$-value function at each $t$ and for each $m$ using the critic network $Q^{\theta_k}$ to compute values $\left(Q^{\theta_k}(s_t^m, a_t^m)_{t \in 0,...,T-1}\right)_{m \in 1,...,M}$
    Compute the actor loss function
        $J^{TRPO}(\omega_k) = \frac{1}{T \cdot M} \sum_{t=0}^{T-1} \sum_{m=1}^{M} \left[ r_{\omega_k}(s_t^m, a_t^m) \hat{A}^{\pi_{\omega_k}}(s_t^m, a_t^m) - \beta \cdot d_{\omega_k}(s, .) \right]$
    Update the actor network
        $\omega_{k+1} \leftarrow \text{argmax}_{\omega_k} \; J^{TRPO}(\omega_k)$
    Compute the critic loss function
        $L^Q(\theta_k) = \frac{1}{(T-1) \cdot M} \sum_{t=0}^{T-2} \sum_{m=1}^{M} \left[ Q^{\theta_k}(s_t^m, a_t^m) - r_t^m - \gamma Q^{\theta_k}(s_{t+1}^m, a_{t+1}^m) \right]^2$
    Update the critic network $\theta_{k+1} \leftarrow \text{argmin}_{\theta_k} \; L^Q(\theta_k)$
    $\omega_{old} \leftarrow \omega_{k+1}$
  end for
```

Algorithm 3 PPO

```
Require: Initial actor network parameters $\omega_0$
Require: Initial critic network parameters $\theta_0$
Require: Trajectory length $T$
Require: Number of trajectories per training epoch $M$
Require: Number of training epochs $N$
Require:
Require: Hyper-parameters $\lambda$, $\gamma$ and $\epsilon$
  Initialize old actor: $\omega_{old} \leftarrow \omega_0$
  for $k \in 0, 1, 2, ..., N-1$ do
    Sample $M$ trajectories $(s_0^m, a_0^m, r_0^m, s_1^m, a_1^m, r_1^m, ..., s_{T-1}^m, a_{T-1}^m, r_{T-1}^m)_{m \in 1, ... M}$ using the actor network $\pi_{\omega_k}$
    Evaluate the $Q$-value function at each $t$ and for each $m$ using the critic network $Q^{\theta_k}$ to compute values $\left(Q^{\theta_k}(s_t^m, a_t^m)_{t \in 0, \ldots, T-1}\right)_{m \in 1, \ldots, M}$
    Compute the actor loss function
      $J^{CLIP}(\omega_k) = \frac{1}{T \cdot M} \sum_{t=0}^{T-1} \sum_{m=1}^{M} \left[\min(r_{\omega_k}(s,a) \hat{A}^{\pi_{\omega_k}}(s_t^m, a_t^m), \mathrm{c}_{\omega_k}^{\epsilon}(s_t^m, a_t^m) \hat{A}^{\pi_{\omega_k}}(s_t^m, a_t^m))\right]$
    Update the actor network
      $\omega_{k+1} \leftarrow \mathrm{argmax}_{\omega_k} \; J^{CLIP}(\omega_k)$
    Compute the critic loss function
      $L^Q(\theta_k) = \frac{1}{(T-1) \cdot M} \sum_{t=0}^{T-2} \sum_{m=1}^{M} \left[Q^{\theta_k}(s_t^m, a_t^m) - r_t^m - \gamma Q^{\theta_k}(s_{t+1}^m, a_{t+1}^m)\right]^2$
    Update the critic network $\theta_{k+1} \leftarrow \mathrm{argmin}_{\theta_k} \; L^Q(\theta_k)$
    $\omega_{old} \leftarrow \omega_{k+1}$
  end for
```

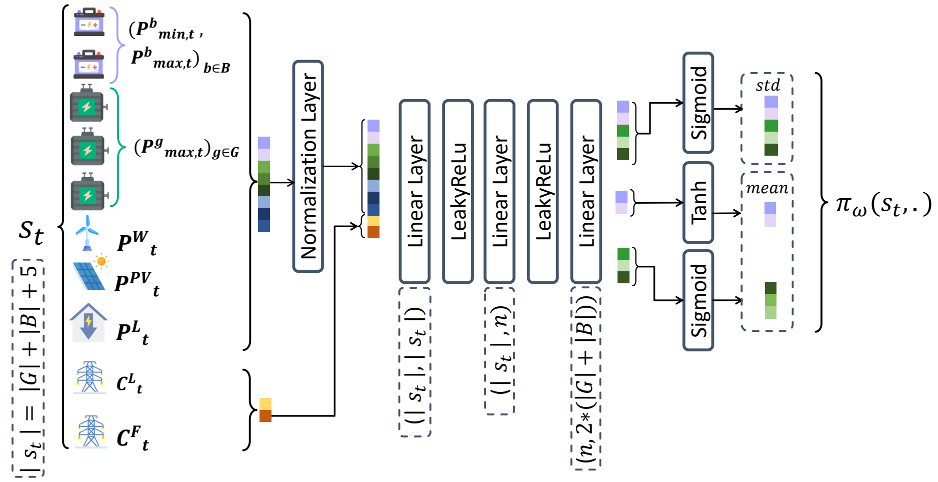

Fig. 6. Layout of the actor network

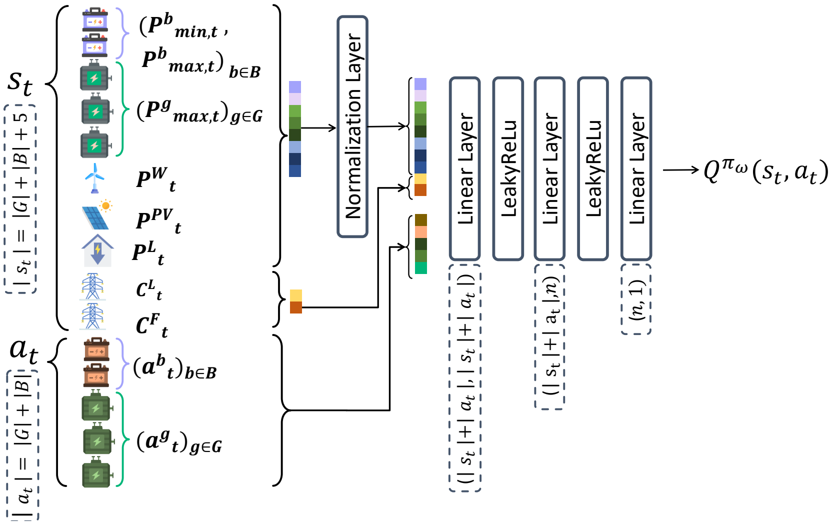

Fig. 7. Layout of the critic network

6 Evaluating the RL algorithms: the dispersion across runs (DaR) metric

The DaR metric was proposed by [3]. Let N be the number of training epochs within one run of the RL algorithm and let L be the number of times that we run the training process. For each $l \in 1, ..., L$ we define the l-th learning curve $\mathcal{C}_l^N$ of the actor network as $\mathcal{C}_l^N = (\mathcal{R}_l^1, \mathcal{R}_l^2, ..., \mathcal{R}_l^N)$ where $\mathcal{R}_l^i$ is the cumulative reward obtained after the i-th training epoch of the l-th run of the training process. Low-pass filtering or smoothing of the learning curves before computing the DaR is recommended by [3]. For this purpose we use the following exponential smoothing function that is defined for a curve $\mathcal{C} = (y_1, ..., y_N)$ of length N such that $\forall i \in [1, N] : y_i \in \mathbb{R}$:

$$\begin{aligned} \text{smooth}_\varsigma(\mathcal{C}) &= (\bar{y}_1, ..., \bar{y}_N) \\ \bar{y}_1 &= y_1 \\ \forall n \in 2, ..., N : \ \bar{y}_n &= \varsigma \cdot y_n + (1 - \varsigma) \cdot y_{n-1} \end{aligned} \tag{47}$$

Where $0 < \varsigma < 1$ is the smoothing factor. We calculate the DaR at a set of evaluation points by sampling them at a certain frequency $f \in \mathbb{N}$:

$$\text{sample}_f(\mathcal{C}) = (y_{t \cdot f})_{t \in 1, ..., \lceil \frac{N}{f} \rceil} \tag{48}$$

DaR has an across-runs variability axis, which means that it is calculated for L learning curves $(\mathcal{C}_l^N)_{l \in L}$:

$$\begin{aligned} DaR_{f,\varsigma}\left((\mathcal{C}_l^N)_{l \in L}\right) &= \Big(IQR(S_l^{f,\varsigma})\Big)_{l \in L} \\ S_l^{f,\varsigma} &= \text{sample}_f(\text{smooth}_\varsigma(\mathcal{C}_l^N)) \end{aligned} \tag{49}$$

7 The local optimum problem in the economic RPM of a microgrid with controllable generators

In the economic RPM problem (environment 1 in table 7), a very important observation is that the reinforcement learning algorithms reach a local optimum when the microgrid includes controllable generators. Figure 8 shows this phenomenon while using PPO. We can see that the controllable generators (denoted 'ctrl_gen_1' and 'ctrl_gen_2') are always operated near their maximum capacity and the batteries (denoted 'battery_1' and 'battery_2') are discharged in the first time step and never recharged. The controllable generators and batteries' parameters used in this experiment are given in tables 2 and 3. Note that in the test case shown, the batteries' initial energy levels are generated randomly.

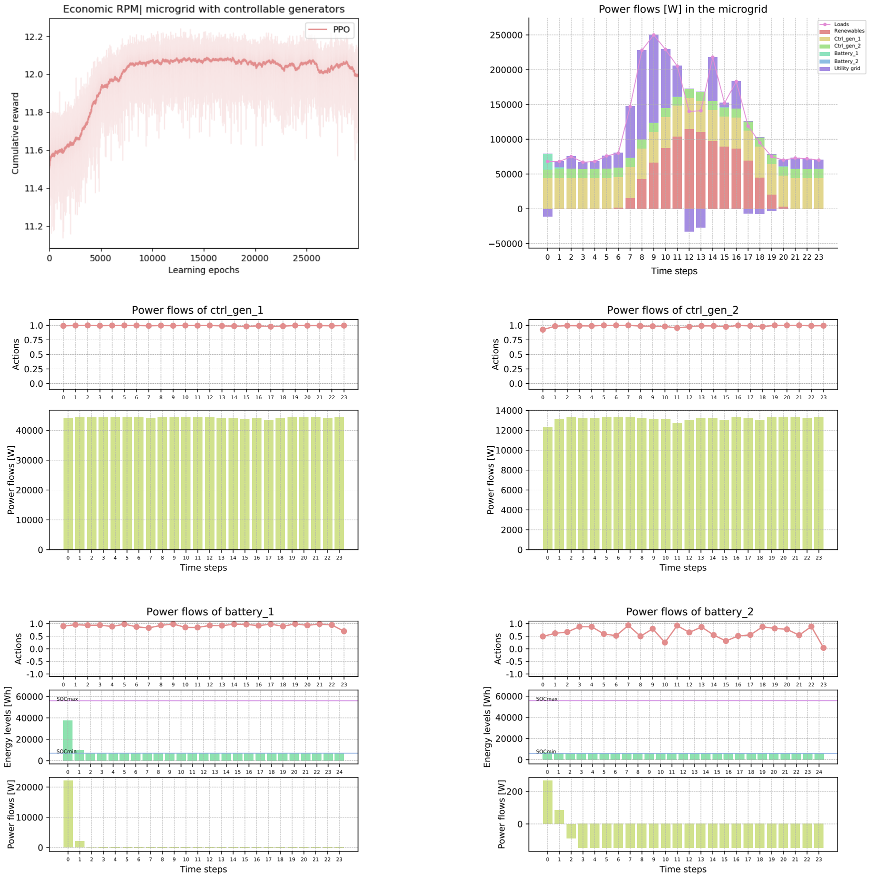

Fig. 8. Economic RPM of a microgrid with controllable generators: the RL algorithm is stuck in a local optimum where the generators are operated near their maximum capacity and the batteries are never recharged.

References

[1] Yves-Marie Saint-Drenan et al. “A parametric model for wind turbine power curves incorporating environmental conditions”. In: *Renewable Energy* 157 (2020), pp. 754–768. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2020.04.123.

[2] Shibhansh Dohare, Qingfeng Lan, and A. Rupam Mahmood. “Overcoming Policy Collapse in Deep Reinforcement Learning”. In: *Sixteenth European Workshop on Reinforcement Learning*. 2023.

[3] Stephanie C.Y. Chan et al. “Measuring the Reliability of Reinforcement Learning Algorithms”. In: *International Conference on Learning Representations*. 2020.