Artificial intelligence for smart grid management

Aleksandr Petrusev

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : GENIE ELECTRIQUE

Arrêté ministériel : 25 mai 2016

Présentée par

Aleksandr PETRUSEV

Thèse dirigée par **Nouredine HADJSAID,** Grenoble INP et co-dirigée par **Patrick REIGNIER,** Grenoble INP, co-encadrée par **Vincent DEBUSSCHERE**, Grenoble INP, et **Rémy RIGO-MARIANI**, CNRS.

préparée au sein du **Laboratoire de Génie Electrique de Grenoble** dans **l'École Doctorale Electronique, Electrotechnique, Automatique, Traitement du Signal (EEATS)**

Artificial intelligence for smart grid management

Thèse soutenue publiquement le 18.11.2022
devant le jury composé de :

Monsieur Yvon BESANGER
Professeur, Grenoble INP, Président
Madame Zita VALE
Professeure, Institute of Porto, Rapporteur
Monsieur Jens HAUBROCK
Professeur, Univ. Bielefeld, Rapporteur
Monsieur Nouredine HADJSAID
Professeur, Grenoble INP, Directeur de thèse
Monsieur Come SOULEZ
Ingénieur Enedis, Invité
Monsieur Patrick REIGNIER
Professeur, Grenoble INP, Co-Directeur
Monsieur Vincent DEBUSSCHERE
Maître de Conférences, Grenoble INP, Encadrant
Monsieur Rémy RIGO-MARIANI
Chargé de Recherche, CNRS, Co-encadrant

Acknowledgements

First and foremost, I would like to thank the esteemed members of the jury before whom I will defend my thesis and my supervisors, Vincent Debusschere, Rémy Rigo-Mariani, Patrick Reignier and Nouredine Hadjsaid, for their guidance, invaluable advices and patience during my PhD study. Their immense knowledge and plentiful experience have encouraged me in all the time of my academic research.

I would like to thank my colleagues from the laboratory. It is their kind support that have made these three years a wonderful time. I am also grateful to Enedis for their financial support, and Come Soulez and Benoit Bouzigon in particular for being my supervisors from Enedis side. My appreciation also goes out to my friends for their encouragement and support all through my work. Big thanks to my wife Iuliia, who has always been with me along the way, and who has been an endless source of inspiration and help. Finally, I would like to thank my parents. They did everything they could and even more, to raise me and my sister and help us achieve what we wanted.

Abstract

The capacities of traditional algorithms used by distribution system operators may be not sufficient to effectively manage the grid with increasing complexity. The growing penetration of intermittent renewable energy sources can cause overvoltage, and massive integration of electric vehicles leads to significant consumption peaks that can be too dangerous for the grid, surpassing the thermal limits of the lines. As such, faster tools are required to regulate power flows and manage operational constraints in distribution grids. A solution may lie in the use of flexibilities, i.e., storage, photovoltaic, and wind farm power regulation along with demand response. With an increasing volume of digital data about the grid, particularly from installed smart meters, machine learning algorithms can obtain significant performance in flexibility management with an already extensive literature on the subject.

This thesis focuses on the development of machine learning-based tools that help distribution system operators to cope with the increasing complexity of distribution grid management. Three different tools are proposed. The first one detects PV installations in a distribution grid without a connection agreement. The second tool is a controller based on reinforcement learning (RL) algorithms (TD3PG and PPO) to regulate the voltage in a distribution grid. The performance of this controller under load and impedance uncertainties is specially investigated. The third tool is dedicated to the control of power exchanged between distribution and transmission grids by using an RL-based controller.

Résumé

Les capacités des algorithmes traditionnels utilisés par les gestionnaires de réseau de distribution peuvent ne pas être suffisantes pour gérer efficacement le réseau avec une complexité croissante. La pénétration croissante des sources d'énergie renouvelables intermittentes peut provoquer des surtensions, et l'intégration massive des véhicules électriques entraîne des pics de consommation importants qui peuvent être trop dangereux pour le réseau, dépassant les limites thermiques des lignes. A ce titre, des outils plus rapides sont nécessaires pour réguler les flux de puissance et gérer les contraintes opérationnelles du réseau de distribution. La solution possible est l'utilisation de flexibilités, c'est-à-dire la régulation de puissance de stockage, photovoltaïque et éolienne ainsi que la réponse à la demande. Avec un volume croissant de données numériques sur le réseau, en particulier à partir de compteurs intelligents installés, les algorithmes d'apprentissage automatique peuvent obtenir des performances significatives en matière de gestion de la flexibilité avec une littérature déjà étendue sur le sujet.

Cette thèse est dédiée au développement d'outils basés sur l'apprentissage automatique qui aident les opérateurs de réseaux de distribution à faire face à la complexité croissante de la gestion des réseaux de distribution. Trois outils différents sont proposés. Le premier détecte les installations photovoltaïques dans un réseau de distribution sans contrat de raccordement. Le deuxième outil est un contrôleur basé sur des algorithmes d'apprentissage par renforcement (RL) (TD3PG et PPO) pour réguler la tension dans un réseau de distribution. Les performances de ce contrôleur sous des incertitudes de charge et d'impédance sont particulièrement étudiées. Le troisième outil est dédié au contrôle de la puissance échangée entre les réseaux de distribution et de transport à l'aide d'un contrôleur basé sur du RL.

Contents

Illustrations

Acronyms

	Acronym
AI	Artificial Intelligence
CNN	Convolution Neural Network
CL	Convolutional Layer
DDPG	Deep Deterministic Policy Gradient
DER	Distributed Energy Resources
DQN	Deep Q-Network
DSO	Distribution System Operator
EPI	Energy Performance Index
ESS	Energy Storage System
EV	Electric Vehicle
FDIR	Fault Detection Isolation and Recovery
GHI	Global Horizontal Irradiance
LSTM	Long Short-Term Memory
LV	Low Voltage
MLP	Multi-Layer Perceptron
MPC	Model Predictive Control
MV	Medium Voltage
NN	Neural Network
OLTC	On-Load-Tap-Changer
OPF	Optimal Power Flow
PCA	Principal Component Analysis
PPO	Proximal Policy Optimization
p.u	Per Unit
PV	Photovoltaic
RES	Renewable Energy Sources
RL	Reinforcement Learning
RMSE	Root-Mean-Square Error
RNN	Recurrent Neural Network
SAC	Soft Actor-Critic
SCADA	Supervisory Control And Data Acquisition
SGD	Stochastic Gradient Descent
SOC	State Of Charge

SVM	Support Vector Machines
TLP	Typical Load Profile
TSC	Time Series Classification
TSF	Time Series Forecasting
TSO	Transmission System Operator
TD3PG	Twin Delayed Deep Deterministic Policy Gradient
VPI	Voltage Performance Index

General introduction

Context

The growing integration of distributed renewable energy sources (RES), such as photovoltaic (PV) and wind turbines, as well as new types of consumers, such as electric vehicles (EV), strongly impacts the operation and planning of distribution grids. Thus, distribution system operators (DSOs) need new tools to cope with increasing complexity due to more volatile energy profiles and greater numbers of controllable assets. This thesis is then dedicated to the development of such tools, especially based on artificial intelligence, to address the following challenges:

- The intermittent nature of RES introduces uncertainty and variability into power profiles and makes voltage and power management a more complex task. The situation is even worse for distribution grids with a high share of PV owners that may not have connection agreements and may not be monitored at a centralized level.

- Additional peaks in consumption due to EV fleets and an abrupt decrease in distributed power generation (e.g., for PV due to clouds) require more reactivity from the DSO to maintain voltage within specified limits. New assets can provide new degrees of freedom that will advantageously complement traditional levers, such as load tap changers of transformers or capacitors banks, which may not react fast enough, and would age rapidly with such frequent usage.

- Traditional DSO controls are based on a model of the grid, which uses the nominal grid (i.e., line) parameters and real-time measurements of power and voltage (or their estimation). However, measurements can be partially unavailable. The same applies to line parameters, especially in low voltage grids where the impedance values are not perfectly known. In addition, those impedances can degrade over time due to aging which can affect the performance of grid state estimation and/or control.

- The integration of RES at the distribution level incurs more volatility of the power profiles at the interface with the transmission network. The power at this interface may then be subject to constraints/limits enforced by the transmission system operator (TSO). Thus, the DSO shall simultaneously control both the voltage and the (active and reactive) power in its grid at the interface with the TSO.

The increase in the volume of data available at medium and low voltage levels, as well as the increase in computing power in recent years, allow the use of artificial intelligence (AI) or, in particular, machine learning, to effectively solve the problems listed above.

Organization of this thesis and principal contributions

This thesis is composed of five core chapters.

The first chapter, "Artificial Intelligence in Distribution Grid Operations", presents the challenges faced by the DSO due to the growing integration of distributed renewable energy sources. This is followed by an explanation of why AI can help to cope with these emerging challenges compared to existing, more conventional, approaches. Finally, examples of existing solutions for the problems that we have decided to address are analyzed, and the shortcomings of these solutions and opportunities for improvement are given.

The second chapter presents the implementation of neural network-based tools to detect PV installations in a distribution grid without any connection agreement. The principal contributions developed in this chapter are:

- The proposition of two different approaches to detect PV– based on classification and prediction. Their sensitivity analysis and comparison;
- The development of algorithms that use only temperature and aggregated net load measurements to detect PV. Calibration of their parameters and investigation of the necessary data to achieve the highest accuracy.

In the third chapter, "Reinforcement Learning for Fast Voltage Control", two reinforcement learning (RL) algorithms are proposed to prepare a controller for voltage regulation purposes in a distribution grid. The efficiency of the algorithms to control storage systems to mitigate both under and overvoltages is evaluated. The main contributions include:

- Comparison (using a dedicated performance metric) of two different types of RL algorithms (off-policy and on-policy) that, according to the literature, show promising results. A large range of sensitivity analyses is conducted, notably on key hyperparameters;
- Training of RL algorithms to implement a voltage control strategy that is based only on the voltage measurements. Thus, the algorithms do not use potentially sensitive private data such as bus load and generation and do not need an external forecast, but implicitly embed a prediction of the power at each bus of the systems.

The fourth chapter, "Reinforcement Learning for Voltage Control Under Impedance Uncertainties", further explores the robustness of RL-based controllers to mitigate uncertainties on line impedances due to a lack of knowledge of the grid a priori or to their continual degradation over time. The principal contributions are:

- The implementation of a two-stage training strategy with offline learning (on a model of the grid) followed by an online adaptation of the controller based on regular updates of its parameters;
- The comparison of the proposed strategy with an RL-based controller trained only offline and with a conventional optimization-based approach based on a grid model and conventional load flow equations.

The fifth chapter "Reinforcement Learning for Power in a Distribution Grid" considers both voltage regulation and potential power limits imposed at the TSO-DSO interface. The chapter is dedicated to the investigation of the effectiveness of RL algorithms to fulfill both objectives simultaneously. The principal contributions developed in this chapter are:

- The proposition of a reward function, which allows to effectively train RL-based controllers to maintain voltage and power exchange within specified limits taking into account the SOC of batteries;
- Training of RL-based controller to maintain various power limits (imposed at the TSO-DSO interface) based only on information about aggregated net consumption at the substation level without energy meters at each bus of the distribution grid.

The manuscript ends with a general conclusion that exposes the most pertinent results and some future research works.

List of publications during the thesis

The state of art study presented in Chapter 1 led to communication in a national press magazine on Electrical Engineering:

- A. Petrusev, R. Rigo-Mariani, V. Debusschere, P. Reignier, N. Hadjsaid, "L'intelligence artificielle au service de la gestion des réseaux de distribution," *Revue de l'électricité et de l'électronique* N°2/2022.

One of the approaches for PV detection proposed in Chapter 2 was published in an international conference paper:

- A. Petrusev, R. Rigo-Mariani, V. Debusschere, P. Reignier, N. Hadjsaid, "Model-free Detection of Distributed Solar Generation in Distribution Grids Based on Minimal Exogenous Information," *ELECTRIMACS 2022*, 2022.

The comparison of the two approaches from Chapter 2 for PV detection was published in an international conference paper:

- A. Petrusev, R. Bauer, R. Rigo-Mariani, V. Debusschere, P. Reignier, "Comparing Time Series Classification And Forecasting To Automatically Detect Distributed Generation," *IEEE Madrid PowerTech*, 2021.

Finally, the results of voltage control using RL algorithms proposed in Chapters 3 and Chapter 4 were submitted to an international journal in April 2022:

- A. Petrusev, M.A. Putratama, R. Rigo-Mariani, V. Debusschere, P. Reignier, N. Hadjsaid, "Reinforcement Learning for Robust Voltage Control in Distribution Grid under Uncertainties," *Sustainable Energy, Grids and Networks* (accepted November 2022).

1 Artificial Intelligence in Distribution Grid Operations

1.1 Context of the study

The energy sector is currently undergoing a transition from a fossil-fuel-based model toward a smarter, more sustainable model. One of the main actors affected by these changes is distribution system operators (DSOs). DSOs are the managers of the distribution grids that operate at low (LV) and medium voltages (MV) (below 50 kV in France) and are responsible for distributing the energy to end customers. The DSO responsibilities cover a wide range of historical tasks that are well mastered, such as [1]:

- Maintaining the distribution grid voltage within the contractual limits by controlling the means of regulation (e.g., capacitors, tap changers of transformers, reconfigurations, etc.);
- Control of power exchange between distribution and transmission grids;
- Optimization of the grid losses under operational constraints (e.g., by reconfiguration of the grid), while maintaining a reliable system configuration;
- Taking equipment out of service (e.g., transformers) for maintenance or repair;
- Diagnostics and prevention of faults and incidents in the distribution grid (i.e., ensuring the continuity and quality of power supply);
- Isolation of faults and restoration of the grid with the highest possible speed and reliability (FDIR - fault detection isolation and recovery);
- Planning and grid expansion;
- Metering and grid data management, etc.

In France, more than 500 DSOs of the main utility company (Enedis) provide such services in 29 distribution agencies that control 95% of the distribution grids in the country (Figure 1.1) [2]. However, their environment is deeply evolving. The first reason lies in the increasing integration of distributed energy resources (DER) in MV and LV grids, such as photovoltaic (PV) and wind power plants. In addition, load profiles change significantly due to new electricity usage such as electric vehicles (EVs), which create load peaks at the end of the day due to almost simultaneous connections. Such intermittent generation from DER and non-conventional loads strongly impact the distribution grid operation (e.g., increasing energy losses or leading to voltage violations) and make it more difficult to control [3].

However, new means of regulating the distribution grid, called flexibilities, are also emerging. Flexibilities include, for instance, EVs battery when connected to the grid, as well as classical storage systems and/or the ability of customers to control part or totality of their load (i.e., demand response). DSOs can access such means of regulation while remunerating the providers of flexibility for their participation availability [4]. In addition, flexibilities include the partial control of renewable-based DER such as PV inverters [5], which allows large-scale power regulation in the grid and allows an even higher share of DER. Finally, the concept of prosumers can be defined, as non-passive consumers who can change their consumption or produce energy themselves and share the surplus with others [6]. All these changes require coordination of MV controls with LV supervision to optimize the grid management while ensuring its stability.

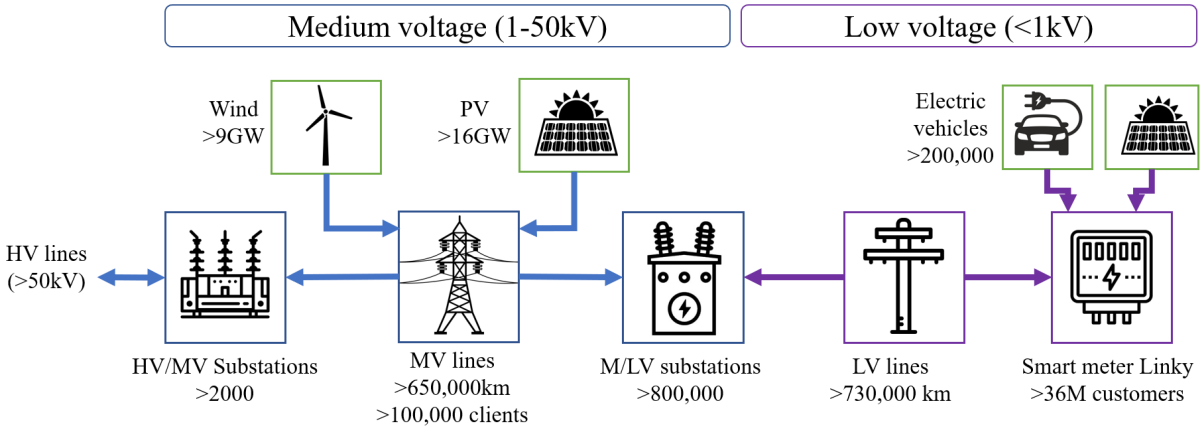


Figure 1.1.Distribution grid under Enedis supervision in France

Traditionally, expert systems were proposed to help with the tasks of the DSO. An example of an expert system is a supervisory control and data acquisition (SCADA), relying on optimal power flow (OPF) computation and Fault diagnosis and Restoration assistant [7]. However, expert systems have drawbacks such as:

1. Lack of creative responses that human experts are capable of;
2. Difficulty in automating complex processes;
3. Little or no flexibility and ability to adapt to changing conditions;
4. Difficulty to recognize the absence of a response.

In addition, all the new flexibility leverages (e.g., batteries and demand response), as well as the introduced new variables in the energy balance equation (e.g., PV, wind plants, and EV) that affect load and generation profiles, make the work of the DSO more complex. Thus, the capacities of expert systems may not be sufficient to effectively manage complex grids. This requires the development of new tools that can overcome this challenge and assist the DSO in some tasks.

Over the past few years, the installation of smart meters (“Linky” by Enedis) has brought a significant volume of information about the LV grid. This leads to an increased

interest in the use of artificial intelligence (AI), or more specifically machine learning, to replace some of the existing tools and bring additional functionality to distribution grid management.

1.2 Machine learning

Machine learning is a field of study of artificial intelligence that is based on mathematical and statistical approaches to enable programs to "learn" from data, that is, to improve their performance in solving tasks without being explicitly programmed for each of them. Learning algorithms can be categorized according to their way of learning and their functionality into three main categories – Unsupervised learning, supervised learning, and reinforcement learning. Algorithms of each category can be applied to different problems, as shown in Figure 1.2.

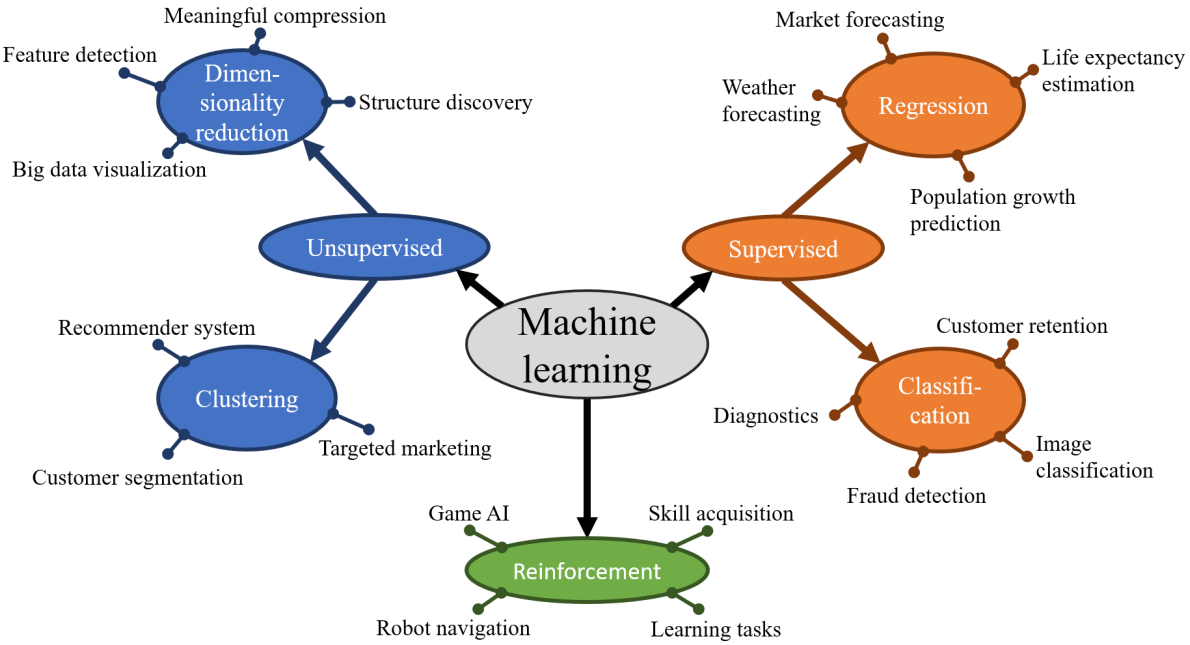


Figure 1.2. Main categories and corresponding examples of machine learning algorithms

1.2.1 Unsupervised learning

Unsupervised learning is a category of algorithms that uses unlabeled data (i.e., data without any characteristics tagged by humans) to analyze and cluster it. These algorithms can automatically identify hidden patterns or classify data without human intervention. They can be used alone (e.g., customer clustering) or combined with other algorithms in a pre-processing data phase. The two main types of unsupervised learning are clustering and dimensionality reduction.

1.2.1.1 Clustering

Clustering includes data mining algorithms that group unlabeled data based on their characteristics. Clustering algorithms are used to process raw, unclassified data objects into groups represented by structures or patterns, that may not be obvious to humans.

One of the most widely used clustering algorithms is K-means. Its operation is briefly described here. For the training, the number of clusters *k* is first set. The algorithm then starts iteratively assigning all data points to these *k* groups (Figure 1.3). During clustering, each group is defined by creating a centroid for this group. The data points closest to the given centroid will be grouped into the same category.

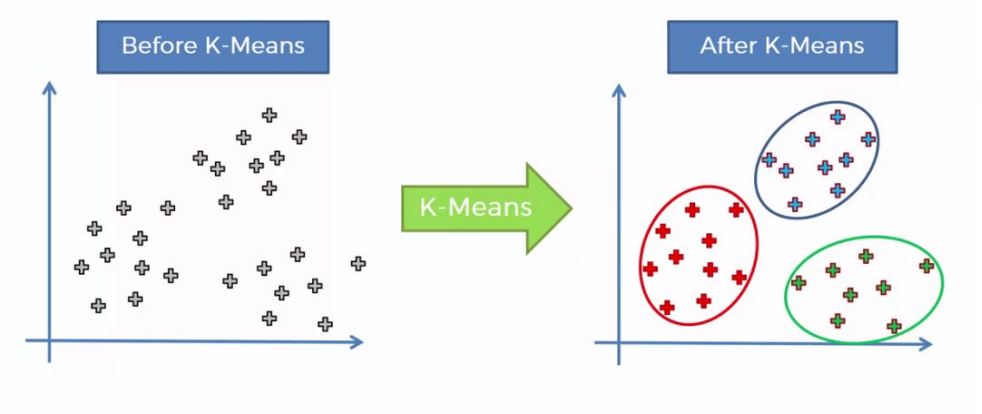


Figure 1.3. Example of K-means clustering with number of clusters k =3 [8]

1.2.1.2 Dimensionality reduction

Although a larger dataset provides more information, and therefore usually allows getting higher accuracy for machine learning algorithms trained on this dataset, it also affects the performance of the algorithms – i.e., algorithms need more processing power and time to be trained. In these cases, dimensionality reduction algorithms can be applied to reduce the number of dimensions or features (independent variables) and at the same time preserve data integrity. This technique also helps visualize high-dimensional data and can mitigate the overfitting of other machine learning algorithms (i.e., when they fit training data very accurately and have poor generalization).

An example of a technique for dimensionality reduction is the principal component analysis algorithm (PCA). It summarizes a large dataset into a smaller set of "summary indices" called principal components [9]. To do this, it sets *k* principal components in directions that maximize the variance of the dataset. This way, it allows building a compact internal representation of data from *n*-dimensional space into *k*-dimensional subspace (where $k<n$).

Another example of dimensionality reduction is the autoencoder. It compresses the data (encoding) by minimizing the number of available features and then recreates from these compressed data again the input (decoding).

1.2.2 Supervised learning

Supervised learning is a category of machine learning algorithms that are trained on labeled datasets. They are trained iteratively until they detect internal patterns and relationships between input and output data. These trained algorithms are then applied to new data that is not necessarily identical to the samples of the training dataset. A properly trained algorithm based on the similarities between its training dataset and new input in the operational phase can then produce the output for that new input. The most popular applications of such algorithms are regression (value prediction) and data classifications.

1.2.2.1 Linear regression

Regression algorithms are used to produce a numerical relationship between given input and output. This allows predicting the value of one variable based on the values of other variables. Linear regression is one of the most basic examples. It looks for a line or surface that minimizes the differences between predicted and actual output values. For every input variable of the vector $\mathbf{X} = [x_1, x_2 \ldots x_n]$, linear regression adjusts the coefficients $\boldsymbol{\theta} = [\theta_1, \theta_2 \ldots \theta_n]$, which are called weights, to obtain the desired line $f = x_1\,\theta_1 + x_2\,\theta_2 + \ldots + x_n\,\theta_n + b$, where b – is the bias term (constant). The training consists in finding the best weights values that minimize the error between the estimated and measured/historical output data.

1.2.2.2 Support vector machine

A support vector machine (SVM) is a supervised algorithm, which can be used for both, regression and classification. It is based on two main ideas. The first is the concept of margin, which is the distance between a decision boundary and samples in a training set, as shown in Figure 1.4. The choice of separation boundary must maximize this margin (i.e., find the maximum distance between data points of the classes). The objective is to find the optimal separating boundary for the dataset by formulating the problem as a quadratic optimization, i.e., maximization of the Euclidian distance between the data set and the boundary. In its most simple type, SVM supports binary classification, i.e., separates data points into two classes. Support vectors are data points that are the closest to the decision boundary and limit its orientation and margin.

To be able to deal with cases where the data points are not linearly separable, the second key idea of SVMs is to transform the representation space of the input data into a higher dimensional space, in which there is probably a linear separation. This technique is known as the "kernel trick".

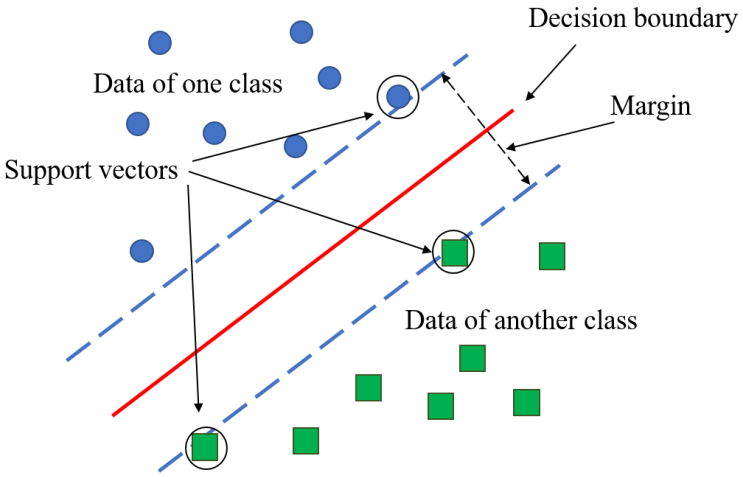


Figure 1.4. Support vector machine principle

1.2.2.3 Neural network

Neural networks (NN) can be considered a logical extension of linear regression. In terms of structure and principle of execution, NN has two main distinctions compared to linear regression. The first one is the use of an activation function, which has a nonlinear relationship between the input f and the output g (usually with a saturation effect). An example of an activation function is the sigmoid (1.1), which maps any input f to a value between 0 and 1. Its graphic representation and along with other examples of activation functions are presented in Figure 1.5.

$$g(f) = \frac{1}{1+e^{-f}} \tag{1.1}$$

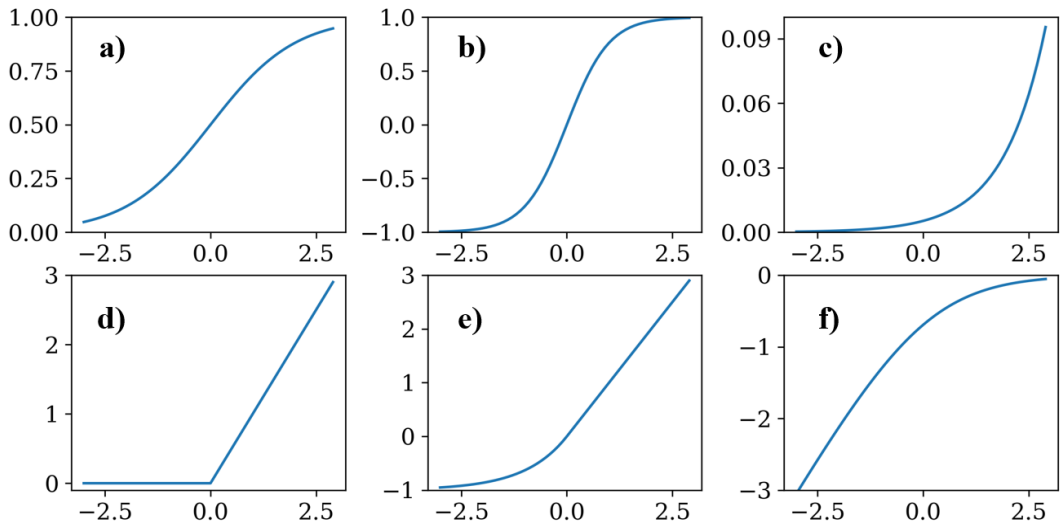


Figure 1.5. Activations functions – a) Sigmoid b) Tanh c) SoftMax d) ReLU e) ELU f) Log Sigmoid

The second distinction is the higher complexity of the structure. Instead of only one output function $f_1(\mathbf{X})$ for n input variables $\mathbf{X} = [1, x_1, x_2 \ldots x_n]$ as in linear regression, NN has a vector of k functions $\mathbf{F}(\mathbf{X}) = [f_1, f_2, \ldots f_k]$. Then each of these functions is converted by the activation function $g(f_k)$ and the resulting values are used as new input together with bias (a node with constant value), forming a new layer. In other words, the vector (layer) $\mathbf{G}=[g_1, g_2, \ldots g_k]$ is fed to another subsequent layer, as shown in Figure 1.6. The required number of such layers typically increases with the complexity of the objective. Through different layers, we pass from low-level parameters to higher-level parameters, rising in abstraction concerning the data, which is often incomprehensible to the human operator of the algorithm. NN is explained more in detail in section 2.2.

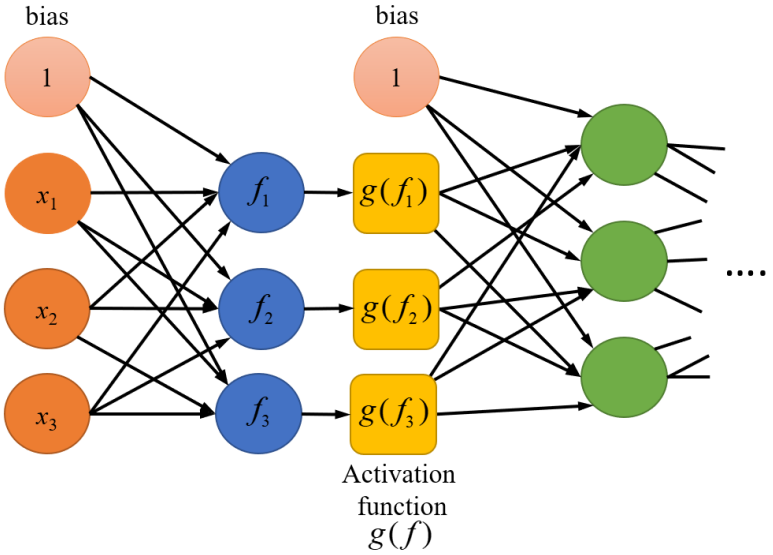


Figure 1.6. Neural network structure example

Convolutional neural networks (CNN) are one of the popular types of NN, which automatically extracts patterns and features from the input data before feeding it to the next layers, thanks to a special operation called convolution (more details are in section 2.2.2). CNN is especially popular for image and time series applications.

1.2.3 Reinforcement learning

Reinforcement learning (RL) is a type of machine learning algorithm designed to solve tasks without correct examples (as opposed to supervised learning). Instead, the algorithm gets the sense of how good its actions are based on a reward. The reward function assigns positive values for the desired behavior and negative values for the undesired ones. During the training, the RL algorithm interacts with its environment and receives the reward based on the evaluation of its actions through the reward function. The goal of the RL algorithm is to maximize, for each action, the sum of the future expected rewards. Thus, by trial and error, after a sufficient number of interactions, the algorithm starts adjusting its actions for new situations in such a way as to obtain a higher expected reward.

These algorithms are used for tasks where we do not know the optimal behavior of the system that is controlled. RL algorithms can be classified according to their approach to model-based and model-free. In model-based algorithms, the agent (controlled entity) either has access to a complete model of the environment or attempts to learn it through interactions. A model of the environment here refers to a function that predicts state transitions and rewards. The main drawback of model-based algorithms is that a ground-truth model of the environment is usually not available to the agent. Model-free algorithms adjust their policy based on the consequences of their actions. One of the main advantages of model-free RL algorithms over model-based RL and classical optimization algorithms is their ability to optimize even without any knowledge about their environment but based only on their own experience. RL algorithms can be also classified by their behavior during training. On-policy algorithms use the same policy once learned. Off-policy algorithms can use a policy, which is different from the learned one. Both types have their advantages and drawbacks. RL, its types, and some of its examples are described in detail in section 3.2.

1.2.4 Machine learning relevance

Most of the principles of machine learning algorithms were developed back in the 20th century. For example, the basic neural network (NN) was presented in 1969 in a paper by Marvin and Papert [10]. Temporal-difference (reinforcement learning method) was formulated in 1988 by R. Sutton [11]. However, computers in that era did not have sufficient processing power to efficiently perform any complex computation required by large models and suffered from a lack of sufficient data. Thus, research in this field has slowed down. But in recent years, machine learning algorithms have started to show their rapidly growing efficiency thanks to the increase in computing power and data volume. This growth also was accelerated by GPU (Graphics Processing Unit) development. The relevance can also be demonstrated by the dynamic of the machine learning market size in Europe. As shown in Figure 1.7, in 2013 the total market was estimated at $324M, but in just five years, in 2018, it increased to $6.9 billion, and its annual growth rate to 2025 is expected to be 44% (year-to-year) [12]. This demonstrates how currently relevant machine learning is.

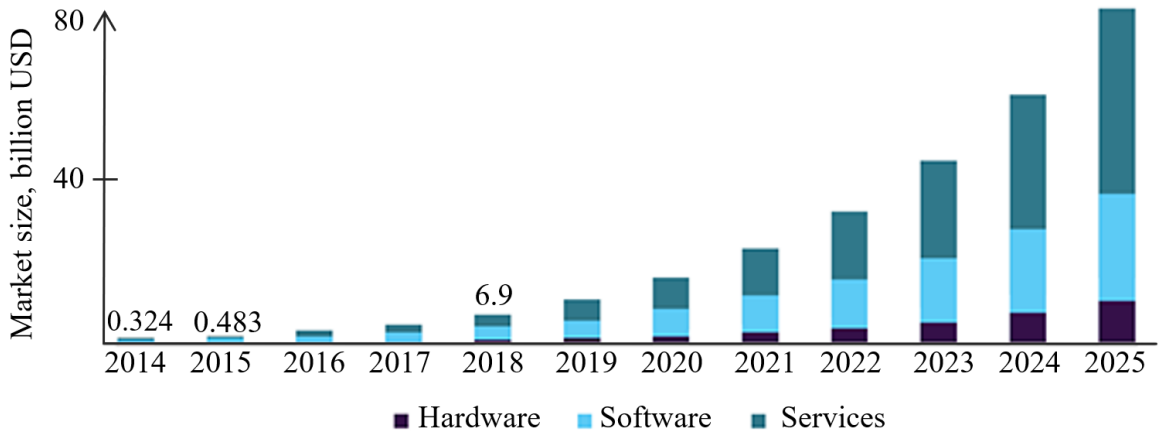


Figure 1.7. Europe machine learning market size, by component, 2014-2025 (USD Billion) [12]

1.3 Selected applications of AI-based solutions for DSOs

Based on the capacities and specificities of machine learning, it is reasonable to say that it can contribute to the daily objectives of DSOs when classical tools/techniques become insufficient. Considering the growth of renewable energy sources, it is interesting to consider the challenges associated with it, as well as possible solutions using artificial intelligence.

1.3.1 PV detection

The share of renewable energy continues to increase rapidly in Europe, and in particular in France. Hydropower energy is still the most significant renewable energy source in the country. However, new installations are limited by the availability of suitable locations. In addition, hydropower plants need huge investments and time for construction, typically require massive deforestation, and often have a high cost of transmission and losses due to the distance from consumption centers. In this regard, it is important to consider also other renewable energy sources that do not have these disadvantages. Wind and solar generation are getting more popular and constantly increasing their share in the total renewable energy balance. They have fewer requirements for location, can be small (PV) and medium (both PV and wind) size and need less time for construction. Moreover, over the past two decades, their installation has been financially supported by government programs.

The dynamics of installed PV capacity in France for 2010-2020 are shown in Figure 1.8 [14]. At the end of 2021, PV capacity reached 13.06 GW, thus representing a 21% share of the installed renewable capacity mix in France. But even though solar capacity has been growing faster last few years (adding, for instance, 2.68GW in 2021 compared to 1.2GW for wind capacities), wind power plants still have a higher total installed capacity of 18.68GW [15].

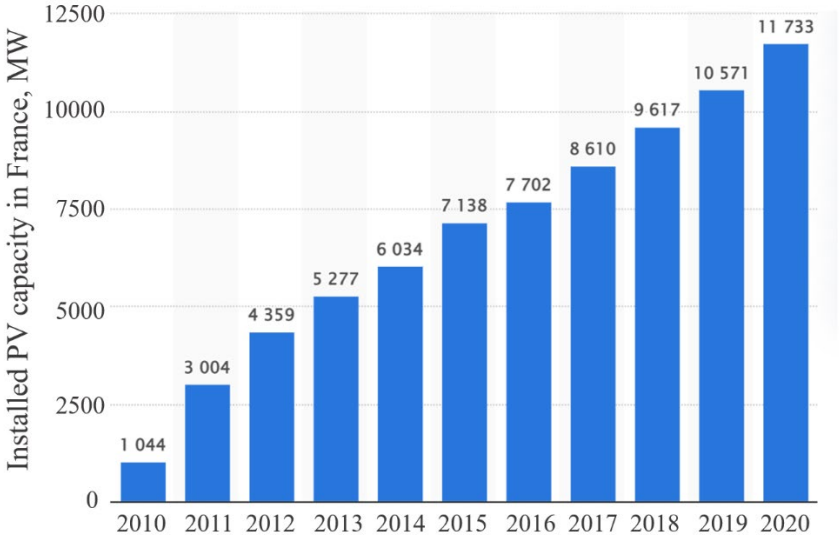


Figure 1.8. PV capacity in France between 2010 and 2020, MW [14]

Regardless of the advantages of solar and wind power plants, they suffer from high intermittency of generation and very limited controllability. Due to the still relatively small share of wind and solar generation in the total production mix (36.8 TWh and 14.3 TWh, respectively, for 468 TWh consumption in 2021 in France [16]), they do not have such a significant impact on the transmission grid. However, they can impact more local distribution grids, raising the issues of power flow management and voltage quality at such voltage levels [17].

To deal with the intermittency of renewable power generation, it is necessary to use generation or net consumption predictors. Their use allows knowing and preparing the necessary capacities in advance and more quickly adjusting the production to the consumption. Typically, different types of NN can be used for this task. For example, the most standard type of NN, multilayer perceptron (MLP), can be used for the load [18], PV [19], or wind [20] generation forecast, where the choice of activation function impacts the final accuracy. Even such a basic type of NN can outperform more classical models of time series analysis and prediction such as, for instance, the autoregressive integrated moving average [19]. A more sophisticated option is to use MLP with autoencoders [21]. Several consecutive autoencoders at the beginning of the network can better prepare the features of the MLP and thus improve its accuracy. Similarly, CNN (Convolutional Neural Network) can be used for this task, as they already embed feature extraction (by convolution as furtherly presented in Chapter 2) [22]. But the NN can show even faster and more efficient training if it is combined with the K-means method. K-means in this case divide the dataset into small clusters with similar data points, which are used separately for the training of CNN. This method shows higher accuracy compared to MLP and CNN alone [23].

Recurrent neural networks (RNN) and long short-term memory (LSTM) are specifically used for sequential data. A recurrent neural network is a type of NN that uses its previous outputs as inputs thanks to its recurrent connections between nodes (Figure 1.9). Due to this, it works not only with the last input, but also maintains an internal state (“memory”) of the previous ones, which helps improve the forecast. LSTM is a type of RNN with a more complex structure, which has so-called gates that supposedly better regulate the flow of information through the nodes. Thus, they can be a good solution for obtaining a prediction of the entire time series.

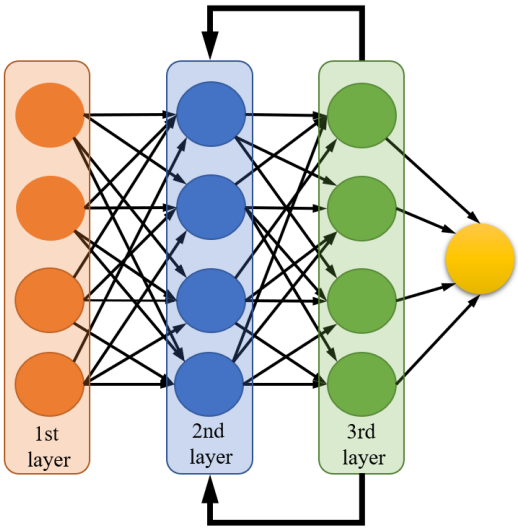


Figure 1.9. Recurrent neural network principle

RNN can use different inputs for solar power prediction which depend on the available data and time resolution. They can use, for example, only historical PV power outputs to predict an average generation for the next hour [24]. Others can use global solar irradiation, wind speed, ambient temperature, and PV module temperature to predict day-ahead power with 15-min timesteps [25]. Finally, the biggest volume of data is necessary for a prediction with 5-min resolution – wind speed, temperature, humidity, global horizontal radiation, diffuse horizontal radiation, wind direction, and current power output of PV [26]. Accuracy comparison between PV power forecasting approaches is difficult because their performance is measured based on root-mean-square error (RMSE) for different datasets and different nominal powers. Thus, relative root-mean-square error (rRMSE) is more commonly used to perform such comparisons [27].

Similar solutions for wind power forecast use typically such characteristics for the input as wind speed, air density, wind direction, ambient temperature, etc. [28]. Prediction resolution varies from 1 min [29] (with the normalized mean absolute error between 14.8% and 16.2%) to 1 h [30] (with the normalized mean absolute error between 5.2% and 6%) for the day-ahead prediction.

However, before applying prediction algorithms for generation, the DSO has to know where and what type of renewable energy is installed. This is usually not a problem for wind power plants, which typically are directly connected to the MV and can be easily supervised. However, this is not always the case for PV. A lot of small PVs are owned by households connected at LV level and are typically installed on the roofs. Each household taken individually does not noticeably affect the distribution grid, but their combined impact can be significant. Moreover, such households do not necessarily have a connection agreement with DSO. In this case, their behind-the-meter PV is “hidden” for the DSO and can unpredictably change the total net consumption at the bus in the distribution grid (point of common coupling). With the constantly increasing share of PV in the total energy mix, this problem becomes even

more relevant. A possible solution is to use the tool which can detect such "hidden" PV based on some other exogenous data.

One of the approaches to identifying customers with PV power generation is using net energy consumption data from smart meters [31]. The detection tools can be based on different algorithms. For instance, clustering algorithms (e.g., K-means or self-organizing maps) can divide the net consumption profiles of all households in a given bus into two groups that are labeled as "without PV" (for a cluster with higher total energy consumption) and "with PV" (for a cluster with lower energy consumption), as shown in Figure 1.10.

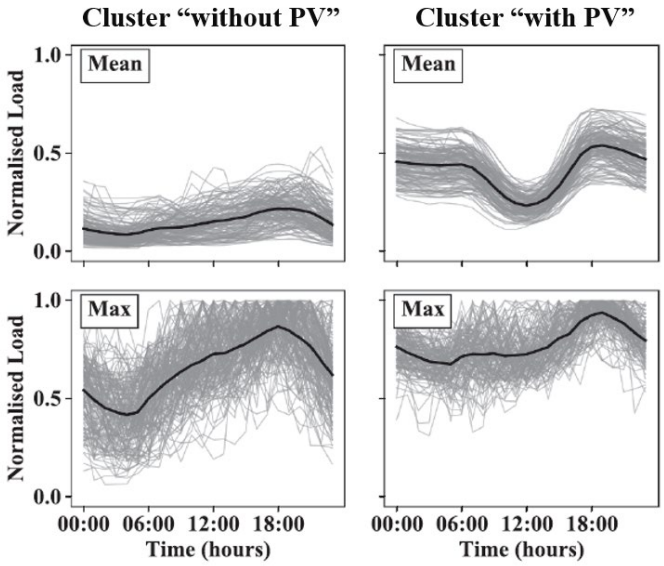


Figure 1.10. Mean and max net consumption profiles for clusters labeled as "without PV" and "with PV" [31]

The solution to reducing the size of the data profiles stored for each household to identify solar prosumers is to use algorithms for dimensionality reduction. The disadvantage of this method is that the clustering of customers in two groups (with and without PV) is implemented without considering the historical data on the consumption of these customers during the previous years. Thus, the algorithm can identify customers as "with PV" if their total consumption or peak consumption is below the average value of other customers, but such a consumption profile is also possible even without PV. The second problem is the need for local measurements from each household, which raises privacy issues and needs the consent of clients.

Another approach for solar prosumer identification is to formulate the objective in the form of a change-point detection problem [32]. A divergence function (a function, which measures the difference between two probability distributions [33]) can be used to measure the dissimilarity between two different daily profiles. One of these profiles is denoted as a "typical load profile" (TLP) for a given consumer. If this TLP and the other daily profile are sufficiently different, the algorithm based on the divergence function detects a change point. i.e., abnormal energy consumption behavior. Change points in customer net load can be caused by other abnormalities (e.g., a new EV, a sudden drop in temperature, etc.). Therefore, the presence of

the unauthorized PV installation is further verified through the analysis of the positive correlation between abnormal consumption behavior and "standard PV energy output". The problems with this approach lie in that the algorithm compares the real measurements of the consumer with his TLP. Thus, temperature and seasonality are not considered. Even an additional check of the positive correlation with PV generation may not help. The following example of false PV detection with such an approach can be considered. With an increase in solar radiation, the temperature typically also rises. Therefore, during the cold season, the customer can reduce consumption with an increase of solar radiation (i.e., abnormal consumption has a positive correlation), but the reason will not be in the installed PVs, but in the fact that he needs less energy for heating. In addition, the comparison requires local PV measurement in the neighborhood to constitute a "standard PV profile", as well as local consumption measurements for each customer.

A completely different algorithm for PV detection is based on very high-resolution color satellite imagery (0.3 meters per pixel) [35]. Random Forest Classification (supervised machine learning technique) detects PV panels in provided images. The main problem of the algorithm lies in its principle - it needs high-resolution satellite imagery. Moreover, it needs labeled training data, thus requiring manual preparation (Figure 1.11).

Figure 1.11. The examples of PV detection are based on very high-resolution color satellite imagery [35]

Another solution for PV detection is not to directly distinguish between net consumption profiles with and without PV but to preliminarily extract features from these profiles that describe their discrepancy [34]. The SVM algorithm then uses these features as input to detect whether the customer has PV or not. The disadvantage of the method is that it needs the output power data from the PV of known customers for the training and running phases to extract the features' net consumption profiles.

Finally, we can also mention a method that uses local Global Horizontal Irradiance (GHI) for PV detection. GHI is the total amount of shortwave radiation received from above by a surface horizontal to the ground. The solution models a PV generation based on GHI and tries to identify PV production patterns in aggregated power flow measurements [36]. The drawback of this approach is the need for a GHI for each bus which may not be practically available. Moreover, the accuracy of detection is lower, if the expected orientation of PV is unknown.

All presented approaches mainly rely on various data such as satellite imagery, solar radiation, nearby PV installations, or detailed weather data for PV detection. Thus, it is interesting to develop algorithms that do not need any special data and can detect PV installations based only on net metering and other always available data. This will be the subject of the first part of the work (Chapter 2).

1.3.2 Voltage control

Knowing the exact locations of all variable RES and having a forecast of their expected power is not enough. The problem of correctly controlling the distribution grid still stands to maintain the voltage at all nodes within the specified limits and regulate the power flow at the primary substation – interface with the transmission grid. Under normal operating conditions, DSOs shall maintain the voltage at each bus within the given limits to guarantee the quality of electricity supplied to consumers and optimize grid operational constraints. These limits, along with other parameters that are important for grid normal operation, are specified in grid codes. An example of requirements for voltage regulation in normal operating conditions (without unpredicted failures) for France is summarized in Table 1.1 [37].

Table 1.1.

STEADY-STATE VOLTAGE REQUIREMENTS IN FRANCE [37]

High voltage limits	*Medium voltage limits*	*Low voltage limits*
±5%	±5%	±10%

The massive integration of RES challenges voltage regulation due to more volatile profiles [38]. Compared to the past, the DSO now faces not only undervoltage problems but also overvoltage situations due to excessive local generation in some buses at some periods, especially when it coincides with low consumption periods. This is aggravated by the fact, that distribution lines typically have a higher ratio of resistance to reactance compared to transmission grids, which makes the distribution grid more sensitive to active power changes.

Conventionally, DSO used for voltage regulation such leverages as transformer on-load tap changers (OLTC), regulated capacitor banks, and/or grid reconfigurations. However, these discrete tools are not suitable for frequent regulation, which can lead to the faster aging of the components [39]. Moreover, their response time may be insufficient compared to the power changes of RES. Another possible solution is the use of flexibility. This includes shifts in prosumers consumption, regulation of solar and wind power plants, and charging and discharging of storage systems (stationary or mobile like electric vehicles). Most of these options are already available for DSOs (Figure 1.1).

Various optimization-based solutions have been proposed for a look ahead and/or near real-time decision-making processes to control flexibilities to maintain the voltage within specific limits, such as described in [40] and [41]. However, these optimization approaches require consumption values from the meters as well as grid data (i.e., topology and line impedance). Thus, the performance of these controllers depends on the capacity to mitigate

uncertainties. The first uncertainty is related to consumption and production forecast. But even with perfect power predictions, the impedances may not be perfectly known at the low voltage level in cases of old facilities. The performance of model-based optimization controllers can be degraded by drifting line parameters from their original values due to aging and weather conditions or by a lack of knowledge of the grid. In addition, the second problem of optimization-based algorithms is that they may lead to a heavy computational burden.

Artificial intelligence (AI) offers a relevant alternative to deal effectively and rapidly with the different sources of uncertainties, not only on grid impedances but also in the local production and consumption forecasts. For instance, neural networks have already provided a noticeable rapidity in grid voltage prediction, allowing us to consider them in quasi-real-time (<1s) [42] and have proven their effectiveness to compute optimal flexibility control of inverters [43] and various DER [44] for voltage regulation. However, NN is supervised learning, and thus it first needs the correct commands from another optimization-based controller to be trained before mapping measurements and controls in actual deployment. This means that the NN needs similar data about the grid to be trained and properly operate later (even if NN is trained to perform optimal control based only on voltage measurements, as shown in Figure 1.12). Moreover, the NN is trained to reproduce the control strategies of a particular optimization model-based controller, thus it will also be biased in case of strong uncertainties – e.g., to the line parameters that are used to provide its training examples.

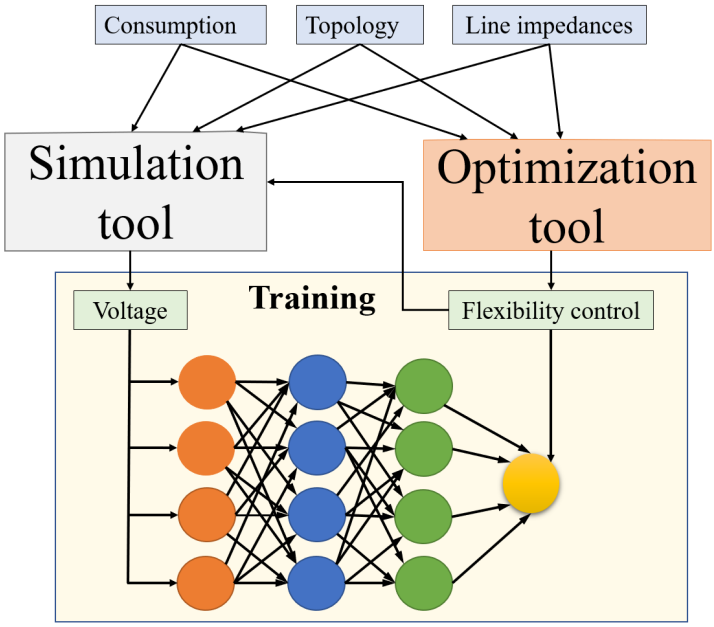


Figure 1.12. Typical synoptic of NN training for voltage control

Reinforcement Learning (RL) approaches can contribute to solving the problem of parameter uncertainty because RL algorithms do not need examples of correct actions to be trained, but only evaluations of how good the performed control was. If RL algorithms are trained on the real grid, the evaluation can be calculated based on the voltages obtained after control, i.e., RL algorithms (which are data-driven approaches) do not need to know the

topology or the line parameters to be trained (like in physics-based approaches). The task of voltage control in electrical grids by RL is widely presented in the literature. RL controllers shall first be trained offline on a model of the real grid or grid activity records. It is because the training of the algorithms on a real grid from scratch can lead to dangerous situations (deep undervoltage or high overvoltage) that are unacceptable.

One example of RL algorithms for voltage regulation is the use of tabular Q-learning [45]. To maintain the voltage, the algorithm can discretely adjust capacitor banks and transformer tap settings. However, the tabular RL algorithm suffers from the curse of dimensionality, i.e., it works well with a relatively small number of discrete states but cannot effectively be scaled up for large systems.

Two reinforcement learning techniques that perform voltage regulation with generation units and rely on two off-policy algorithms, Deep Q-Network (DQN) and deep deterministic policy gradient (DDPG), show more promising results [46]. Deep Q-Network uses NNs to approximate the state space and thus it does not need the tables as in tabular Q-learning. However, DDPG shows better performance, maintaining the voltage in the given limits 99.92% of the time in the study case compared to only 91.25% for DQN. The algorithms need an observation of both active/reactive power and voltage magnitude, which is their main disadvantage.

The literature suggests other ways to apply the deep deterministic policy gradient (DDPG) algorithm to voltage regulation. This can be, for instance, again the control of generation units, but with multiple agents [47]. The use of multiple deep reinforcement learning (DRL) agents allows each zone to be assigned to one of the agents. Each agent receives for its corresponding zone the values of voltages of each bus, loads, generations, and the corresponding power flows. Based on these values, it regulates the local power generation. The overall process is shown in Figure 1.13. Apart from the need to know the values of both the voltage and power, this approach has another drawback. The regulation is performed in several iterations before voltage violations are eliminated. However, this can be a problem, if the system has a long response time to new voltage measurements (e.g., 30 min).

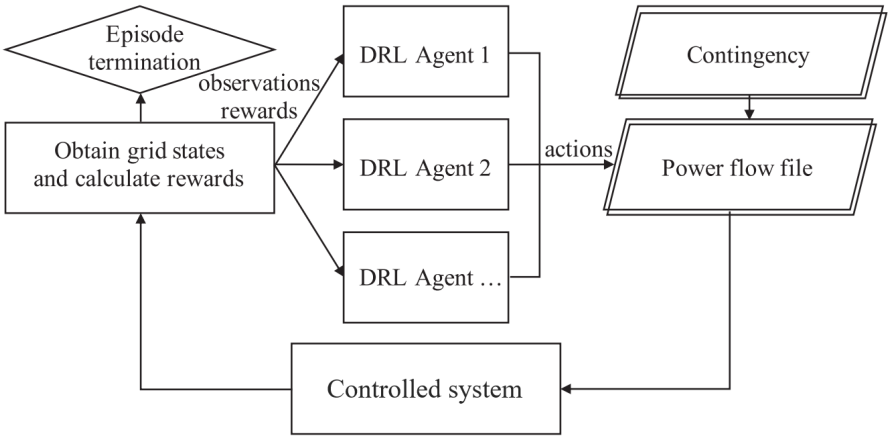


Figure 1.13. Information flow of the DDPG Agent training process [47]

Another application of DDPG for the voltage regulation task is the control of static Var compensator and PV curtailment in a distribution grid [48]. However, this approach also needs active and reactive powers of all consumers and DER as input, which can be problematic due to privacy concerns and practically impossible. A similar input is necessary for Soft Actor-Critic (SAC) algorithm which controls photovoltaic (PV) inverters, OLTC, and switched capacitors [49].

All the solutions mentioned above are implemented and tested disregarding the endogenous uncertainties on network parameters, which may significantly worsen control accuracy in real-life applications. Thus, it is interesting to investigate the efficiency of algorithms under grid parameter uncertainty. This way, the performance of DDPG can potentially be improved further by implementing a two-staged DDPG. In the first stage, DDPG is trained on the model of the grid with known parameters, and in the second stage, DDPG is connected to the target grid and continues its learning (and thus adapts to potential misrepresentations of the simulation environment) by adjusting its parameters through online feedback. Such an approach has been tested to control distributed generation units as well as tap transformer position [50]. However, in addition to the voltage values, an external load and generation predictor is required for the input of the algorithm.

A similar two-stage voltage control strategy (based on DDPG as well) is to mitigate voltage violations caused by electric vehicles (EVs) and loads consumption [51]. The observation/inputs of the DDPG algorithm consist of the voltage, consumption, and generation power. But in addition, the algorithm takes into account the state of charge (SOC) of EV batteries, which can become one of the main sources of local flexibility in upcoming years. However, it uses an external tool for day-ahead reactive power dispatch prediction and considers only power uncertainty, but not the uncertainty of grid parameters.

To summarize, RL solutions presented in the literature do mostly not consider uncertainties, especially for line impedances. They also typically need consumption values for all consumers, which can be a problem due to privacy issues in LV grids. Finally, a lot of them rely on external predictors for generation and consumption. In this regard, it is interesting to investigate the capability of various RL algorithms to regulate voltage in the grid under different uncertainties relying only on voltage measurements or its estimations. These algorithms can regulate common flexible assets in the distribution grid such as local storage. An additional undoubted advantage will be the use of these algorithms without an external power predictor but based only on currently available information. On-policy PPO and Off-policy DDPG showed the highest results for voltage regulation tasks in the literature. However, an improved version of DDPG, twin delayed DDPG (TD3PG) [84] appeared a few years ago and promised to be more performant, but has not yet been extensively investigated (it is presented in detail in section 3.2.2.2). Thus, for the off-policy algorithm, it may be a suitable candidate. Thereby, a significant part of this thesis will be dedicated to voltage regulation by RL algorithms (chapter 3), and then their performance under grid uncertainties (chapter 4).

1.3.3 Power control

Transmission and distribution grids are managed by two distinct operators. However, these grids are interdependent. Thus, regardless of the importance of voltage control in distribution grids, its regulation, if it is possible, should not lead to violation of power limits set for DSO by the transmission system operator (TSO). Power limits for the DSO-TSO interface (at the primary substation level) are necessary for the TSO to better predict and control power balance (between instantaneous generation and consumption) in the transmission grid. It became especially relevant with the growth of RES capacity because increased penetration of fluctuating decentralized generation results in an increase in errors in the generation forecast and, therefore, makes it more challenging to balance the grid [52].

On the other hand, the TSO can, via the DSO, use flexibilities at the distribution level to reduce its imbalances (Figure 1.14). Such regulation can be done by changing the power limit at the TSO-DSO substation, which will motivate the DSO to use the flexibilities. However, the use of flexibility by TSO and DSO requires a high level of coordination. When decisions are made on the TSO side, the impact of these decisions on the DSO side (and vice versa) needs to be taken into consideration [53]. Overall cooperation between system operators supports optimal system planning and operation and this is widely recognized by the regulators. Thus, optimal power management is one of the most important and complex responsibilities of distribution system operators.

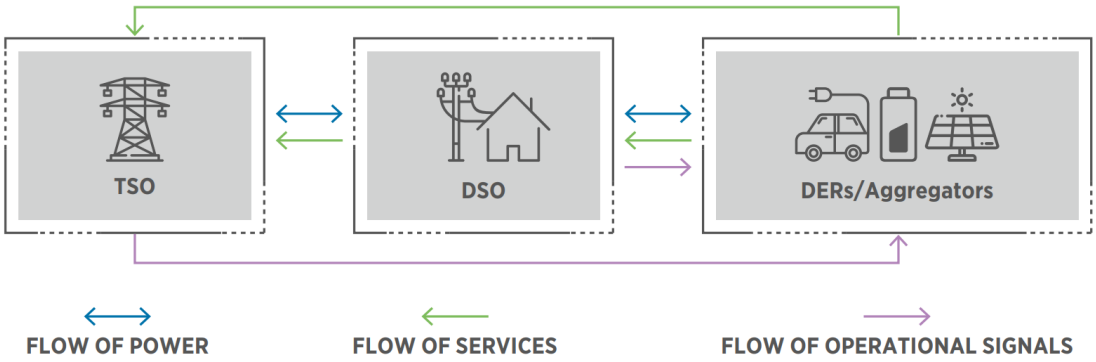


Figure 1.14. Interaction model between system operators and DERs [54]

For power management tasks, traditional algorithms, such as linear programming [55] and dynamic programming [56], require all the parameters of the grid, including topology, impedances, and power measurements at each bus, which is often not available for low voltage grids. Global optimization (metaheuristics) methods such as particle swarm optimizations [57] and/or genetic algorithms [58] are too slow to be applied in real time by the DSO [59]. Pretrained RL algorithms, on the contrary, show rapidity and do not need preliminary information about the grid to be trained.

For instance, one study investigates the performance of various RL algorithms for power management of storage units and thermostatically controlled loads in grids with high

penetration of DER [60]. It considers Deep Q Learning (DQN), SARSA, Actor-critic, and proximal policy optimization (PPO). On-policy PPO showed the best efficiency for the considered problem. However, all considered algorithms assume to have as an input a perfect forecast of temperature, load, and local generation for the next hour, which is not completely realistic.

Another example presents tabular Q-learning [61]. It regulates capacitors and generators to meet operation limits on maximal active and reactive powers. However, it suffers from the curse of dimensionality and uses measurements for each bus, which can be problematic in the LV grid due to privacy issues. The use of fuzzy Q-learning can be a solution to cope with the tabular problem of Q-learning. Fuzzy systems have the advantage of achieving good approximations in the Q-function and simultaneously make possible the use of the Q-Learning in continuous states-space problems. An example of fuzzy Q-learning uses multiple agents to control diesel generators, batteries, and fuel cells for power management in a microgrid [62]. However, along with all the state variables of the regulated assets, the algorithm needs consumption and generation data for each point of the grid.

Another approach for power regulation is the use of PPO for the agents which regulate both total power and voltage in the grid [63]. Each agent in the distribution grid manages active and the reactive power of PV inverters, with possible communication and coordination between agents, as shown in Figure 1.15. Nevertheless, the algorithm, in addition to voltage, also needs consumption in each bus of the grid as input.

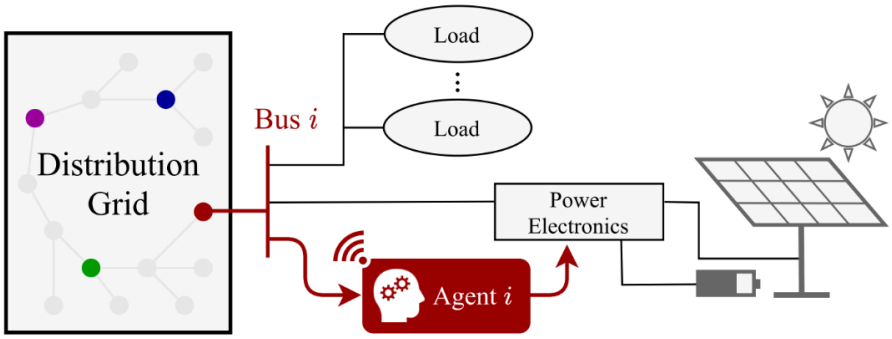


Figure 1.15. Distribution grid control principle, where each colored dot in the ''Distribution Grid'' box refers to one agent [63]

Finally, PV inverters with continuous output can be controlled based on the DDPG algorithm [64]. Its input, as in the previous example, contains voltages and active and reactive powers of all buses. But in this case, the algorithm is centralized and controls all inverters itself instead of distributing the objective among different agents.

Overall, the solutions presented in the literature for power management mainly need consumption and generation magnitudes at every bus of the grid. Thus, considering the increasing complexity of LV grids, it is interesting to investigate the performance of RL algorithms that only have information about the aggregated power at the substation level - i.e., without knowing the exact topology or consumption at the customers' level. Algorithms must

control both, active and reactive powers. And considering that batteries (both local storage and accumulators of EV) undoubtedly can be one of the main flexible options for power regulation in MV and LV grids, they should be considered as a regulated asset for the algorithms. Finally, particular interest is to study the performance of these RL algorithms to fulfill simultaneously multiple objectives or perform a trade-off between them (i.e., power limitation and voltage regulation). All these questions will be covered in Chapter 5.

1.4 Conclusion

This chapter presents the problem of the increasing complexity of distribution grid control. The capacities of traditional algorithms used by distribution system operators may not be sufficient to effectively manage the grid. The growing penetration of intermittent RES introduces tremendous grid challenges. As such, faster tools are required to regulate the power flows and manage operational constraints in the distribution grid. Moreover, RES can cause overvoltage, and massive integration of EVs leads to significant consumption peaks that can be too dangerous for the grid, surpassing the thermal limits of the lines. The solution may lie in the use of flexibilities, i.e., storage, PV, and wind farm power regulation along with demand response. With an increasing volume of digital data about the grid, particularly from installed smart meters, machine learning algorithms can obtain significant performance in flexibility management with an already extensive literature on the subject.

The main types of machine learning algorithms are presented in this chapter, including three categories: unsupervised, supervised, and reinforcement learning. The use of machine learning for solving relevant tasks of a distribution system operator regulating a grid with a high penetration of variable RES is considered. The first considered problem is the detection of local PV generation which is “hidden” for the DSO due to the absence of connection agreement. Literature research has shown existing solutions that mainly rely on a set of various hardly available data such as satellite imagery, solar radiation, nearby PV stations, or detailed weather data. Thus, the interest is to consider the effectiveness of a tool that uses only the net consumption of the bus for the sake of simplicity and explainability.

The second problem under consideration is the voltage control in MV and LV grids, which is highly affected by local generation in the case of massive renewable generation penetration. To cope with grid uncertainties (such as topology, impedances, and power uncertainty), it is proposed to use reinforcement learning algorithms, which do not need information about the grid but can learn directly from interaction with the system. The interest here is to develop a tool that does not rely on consumption measurements or external power predictors but performs control based only on voltage measurements.

Finally, overall cooperation between transmission and distribution system operators is considered. In this perspective, the maximum power flow limits set between distribution and transmission grids must be respected. Thus, it is necessary to investigate the performance of algorithms to fulfill both objectives – maintain the voltage and power within the given limits.

Current RL solutions mainly rely on the consumption and generation profiles of each consumer, which can raise privacy problems in some countries, such as France. Thus, it is of particular interest to consider the efficiency of an algorithm that uses only aggregated power at the substation level.

Each task listed above will be considered in this thesis and the obtained results will be presented in the following chapters throughout the manuscript.

2 Automatic Detection of Nodes in a Distribution Network with PV Production

2.1 Introduction

Distributed renewable energy sources, especially photovoltaic (PV), have grown rapidly over the past two decades. For this reason, information about PV generation becomes crucial for distribution system operators (DSOs) to perform regular operations such as state estimation, reconfiguration, and voltage management among other applications. However, household-owned PV plants do not necessarily have a connection agreement with the system operator and are therefore not monitored at a centralized level by default. If not identified properly, this generation is then "hidden" from the system operator's perspective. This may incur additional uncertainties in the net load measurement and forecast (by reducing the net load compared to the expected one during the daytime) and, in particular, makes it more difficult to securely operate the distribution grid. With the constant growth of installed PV capacities, this issue becomes more and more important. Thus, a relevant contribution to improving electrical grid management performances is to increase the system observability/knowledge with a tool that automatically detects nodes with PV production in the distribution grid based on smart meter data (i.e., it should mark the node either as "with PV", either as "without PV"). This method could be also interesting as the first step of PV desegregation from net metering in PV nodes.

As discussed in the first chapter, there exist several approaches to detect PV generation in distribution grids [31]-[36], but they often require specific data that the system operator may not have access to, such as local measurements. However, in some countries such as France, this direct access to local measurements relies notably on end-user acceptance. 'Behind the meter' measurements, accessible by third parties, do not ensure privacy-by-design. Other methods need various types of data, such as solar radiation data for a specific location [36], data from other PV stations [34], detailed weather data [33], or high-resolution satellite imagery [35]. Those methods are complex and rely on a significant volume of heterogeneous data. A simple and more direct detection tool makes sense for DSO, which does not require end-users' consent, nor such detailed data, and does not contain any detailed modeling of the grid, but strictly relies on metering of aggregated power at feeder level and temperature data. The motivation lies in the capacity of DSOs to use these available data without facing privacy issues.

Machine learning techniques such as neural networks (NNs) are used in a wide variety of contexts in the energy field [65]. However, the detection of PV installations has not been

extensively covered given the practical compromise between data limitation, simplicity of implementation, and accurate results. In this chapter, two different approaches based on NNs are proposed. The first method is based on Time Series Classification (TSC), and the second method relies on the concept of Time Series Forecasting (TSF) which is applied to the classification task.

First, the simulation setup and data for the experiment are described. Both methods and their principles are presented and sensitivity analysis of detection accuracy to magnitudes of hyperparameters is discussed. Then, these two approaches with similar (though not equal) assumptions are investigated, regarding their effectiveness in detecting PV production. Finally, conclusions are drawn.

2.2 Neural networks principle

All neural networks in this chapter are built using the python package PyTorch. PyTorch is an open-source machine learning library based on Torch and developed by Facebook's AI Research lab (FAIR) [66]. Two main types of NN are used here - multilayer perceptron (MLP) for time series forecasting and convolutional neural network (CNN) for time series classification.

2.2.1 Multilayer perceptron

The multilayer perceptron is the most common and basic type of NN [67]. It consists of three types of layers –input, hidden, and output layers, as shown in Figure 2.1. The input layer receives the n input signal $\mathbf{X} = [x_1, x_2 \ldots x_n]$ for processing. The required task, such as prediction and classification, is performed by the output layer, which provides the m output $\mathbf{F} = [f_1, f_2 \ldots f_m]$. An arbitrary number of hidden layers that are placed in between the input and output layers are the true computational engine of the MLP [68].

Each layer consists of bias (which does not have input connections) and neurons or perceptrons. Those perceptrons are simple computational units that perform a weighted sum of input signals and produce an output signal using an activation function. The linear combination W_1^1 of k^{th} perceptron g_k^1 of the Gidden layer p from the example in Figure 2.1 is calculated according to (2.1):

$$W_k^1 = \theta_{0-k}^1 + \theta_{1-k}^1 X_1 + \ldots + \theta_{n-k}^1 X_n \qquad (2.1)$$

Where θ_{n-k}^p is the weight (coefficient) between the node n (bias node is denoted by 0) of the p-1 layer (input layer is denoted by 0) and the k^{th} neuron of the p^{th} hidden layer.

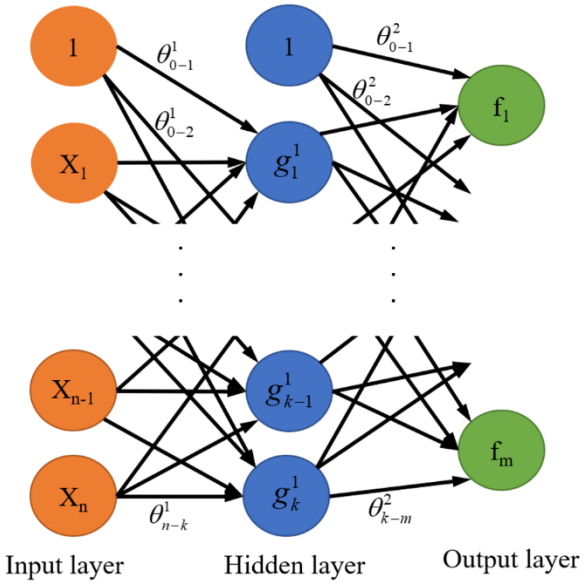


Figure 2.1. Neural network structure example with a single hidden layer

Finally, the weighted sum is passed through an activation function, which produces the actual output of the neuron g^1_k. The activation function allows non-linearity to be constructed in the neural network that provides a richer capability in the functions they can model. MLP is a feedforward class of NN, because data flows in the forward direction from the input to the output layer. Each layer is feeding the next one with the outputs of all its neurons.

MLP is supervised learning, i.e., it is trained on labeled datasets to detect relationships between input and output data. MLP is trained (learns the optimal weights and biases) with back-propagation by using gradient descent algorithms. It tries to iteratively adjust the weights to minimize the cost function L_{cost}. The cost function can be, for example, the error between the output of the NN **F** and the real output of the training set $\mathbf{F_r}$, i.e., the objective is to fit a model. For each iteration, after the NN has computed the output **F**, the gradient $\frac{\delta L_{\text{cost}}}{\delta \theta_i}$ is calculated to minimize L_{cost} concerning the weights of each layer of the NN, $\boldsymbol{\theta}^{\mathbf{i}}$. Then, each iteration of gradient descent updates the weights according to (2.2):

$$\boldsymbol{\theta}^{\mathbf{i}} = \boldsymbol{\theta}^{\mathbf{i}} - \alpha_{LR} \times \frac{\delta L_{\text{cost}}}{\delta \boldsymbol{\theta}^{\mathbf{i}}} \qquad \textit{(2.2)}$$

Where α_{LR} is the learning rate. This parameter determines the size of the gradient descent step at every iteration.

The process of training typically continues until convergence, i.e., until the new gradient $\frac{\delta L_{\text{cost}}}{\delta \theta_i}$ becomes lower than a specified threshold or when the loss function is nullified. Thus, the

process is similar to the least squares method. MLPs are universal approximators of any continuous function and can solve problems that are not linearly separable.

2.2.2 Convolutional neural network

A convolutional neural network (CNN) is a type of neural network designed to work with two-dimensional image data, but it can also be used for 1-dimensional time series. The main difference with conventional NN is that CNN extracts the features (that it will use) from the data using convolutions.

Convolution is central to CNN. It is nothing but a filter with weights, also called a kernel, that is applied to 1-dimensional (in the considered case) or 2-dimensional (images) input data. Depending on the filter, different features can be extracted (e.g., edge filters in image processing). The CNN can learn the weights of the filters and thus features automatically. Convolution reduces the number of inputs from a complex data set, i.e., extracts the representative features before they are fed into a conventional NN. That makes the process overall more efficient. The output of a convolution is called a feature map as it contains the extracted features.

A convolution is an element-wise product of the filter weights and the corresponding data points. Time series can be univariate or multivariate. Univariate time series are datasets comprised of a single series of observations with a temporal ordering (e.g., solar radiation power for the given timestep), and multivariate time series data means data where there is more than one observation for each timestep (e.g., values of temperature, cloud cover index, humidity and wind speed for the given timestep). For multivariate time series, a CNN with multiple input series should be used. An example of a simple convolution process with univariate time series is presented in Figure 2.2. In this case, we have a 1-dimensional input array with a length of 8 timesteps and a simple kernel with weights [1, 0, 1]. To obtain the first value of the feature map, we multiply element-wise the first three values of the array by the given kernel and then sum the result. This way, we obtain a value of 5. Then we shift by one value and repeat the process with the next three values until the entire feature map is filled. This “shifting” mechanism of the filter is implemented by shared weights at the convolution layer. The size of the filter determines the size of the extracted features - larger kernel sizes extract larger and more complex features.

A convolutional layer (CL) consists of one or several convolutional kernels, each kernel is applied to the complete input. The next layer will have the resulting feature maps as input. From this feature map, the next layer can again extract features. The main patterns of the extracted features are therefore more abstracted or coarse than those extracted from the raw input in the first layer.

Apart from convolutional layers (CLs), there are several other layer types than can be placed between the CLs. The most common is “max-pooling”. The pooling layer reduces the dimensionality and down-samples the feature map, making it more robust to changes in the position of the features in the data sequence. Pooling is a filter that takes the maximum or

average of a region. An example of max-pooling is presented in Figure 2.2. After convolution, we apply a max-pooling filter of size 1x2 to our feature map. The resulting array contains the biggest values for each pair of feature maps. The array obtained after convolution and pooling layers are connected to the conventional layers of the NN, which are called fully connected layers.

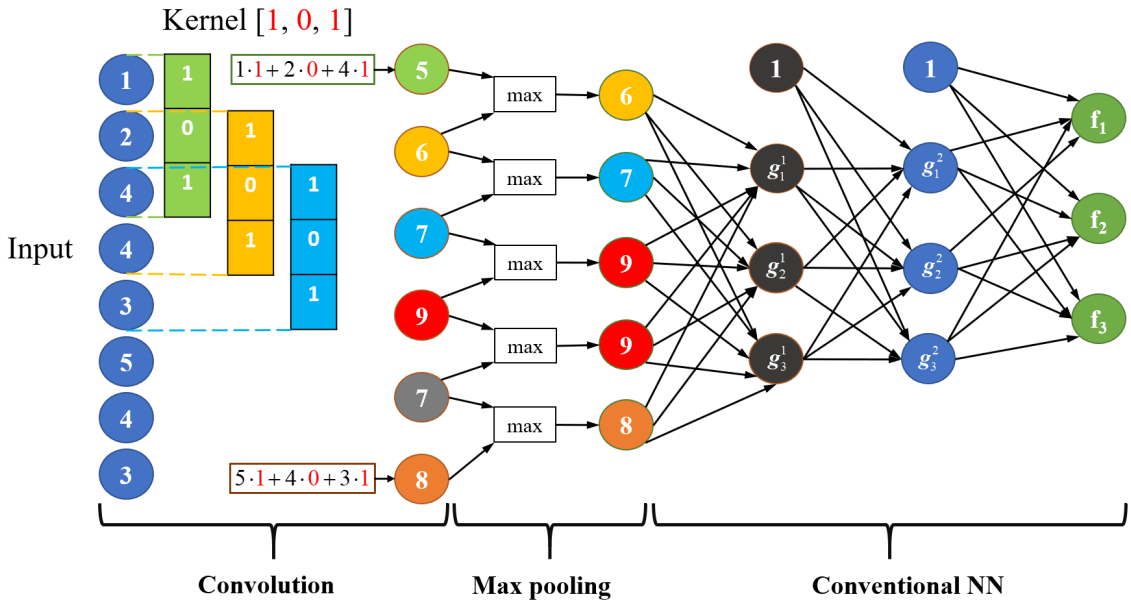


Figure 2.2. Convolutional neural network example, kernel with weights [1; 0; 1]

There are several advantages of the CL that make CNNs very suitable for time series data. They introduce translational invariance, which means that no matter when in the input the feature appears, the convolutions will filter it out. Convolutions also serve as a dimensionality reduction, with an appropriate choice of parameters. Finally, convolutions typically have fewer parameters than NNs for the same prediction quality, which makes them cheaper in terms of training time and may allow using less data for training.

2.3 Simulation setup

Two different databases of consumption, available online, are used for the experiments. The first dataset is “Smart meters in London” [69]. This database contains energy consumption data of 5,567 London households that participated in the UK Power Networks Low Carbon London study between November 2011 and February 2014. Analysis of only one consumption profile at a time is of no interest since the task of detecting PV, in this case, is too trivial. The installation of PV can be easily detected due to low consumption (see Figure 2.3) by a household around noon (when family members are usually away from home) and high PV generation at the same time, which leads to a reverse power flow (i.e., to the grid) even with small PV installations.

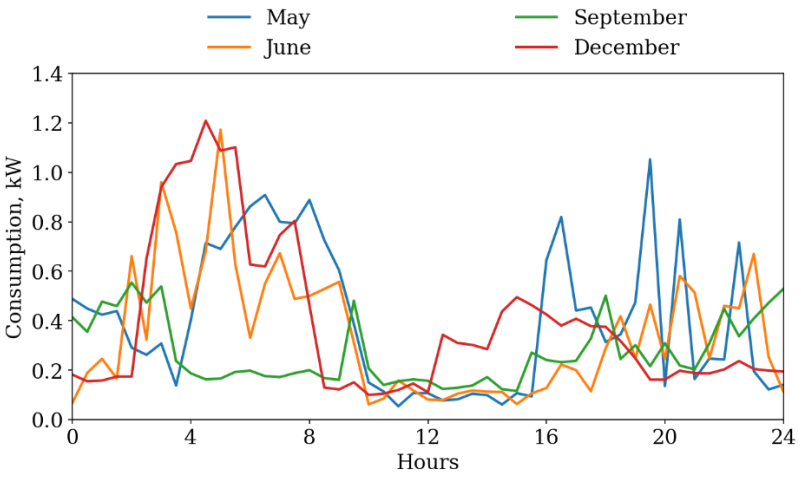


Figure 2.3. Consumption profiles for one household of "Smart meters in London" database

Thus, all 5,567 households are divided into 111 blocks with approximately 50 households in each to smooth power consumption profiles. The data of 14 blocks for the full years of 2012 and 2013 are extracted and converted into two-year profiles with a one-hour timestep for each node (examples of profiles are shown in Figure 2.4 (a) and Figure 2.4 (c)). The power at each timestep of the profile corresponds to the total consumption of the block for the given period.

The second database is Iowa Distribution Test Systems [70], which includes 1120 consumers connected to a grid located in the Midwestern United States. Similarly to the previous database, 14 aggregated profiles are prepared (examples of profiles are shown in Figure 2.4 (b) and Figure 2.4 (d)). Compared to the previous database, it contains two types of consumers: commercial buildings and households. The difference is seen on the 1-day profile (Figure 2.4 (d)). Loads 1 and 2 (which correspond to commercial buildings) have a peak consumption from 8 a.m. when employees arrive to work with the air conditioning system turned on, to 5 p.m., when they leave to go home. For loads 3 and 4 (which correspond to households), the situation is the opposite: peak values are in the morning and the evening. Thus, this database allows for testing more types of consumers. However, its data only cover a single year, 2017. Thus, both databases were used for the first method (described in section 2.4), but only the first database was used for the second detection tool (described in section 2.5) because this method needs more than a year for the test. Two datasets are also different over the year. The London database displays higher consumption in winter due to heating, and the Iowa database has higher consumption in summer due to air conditioning.

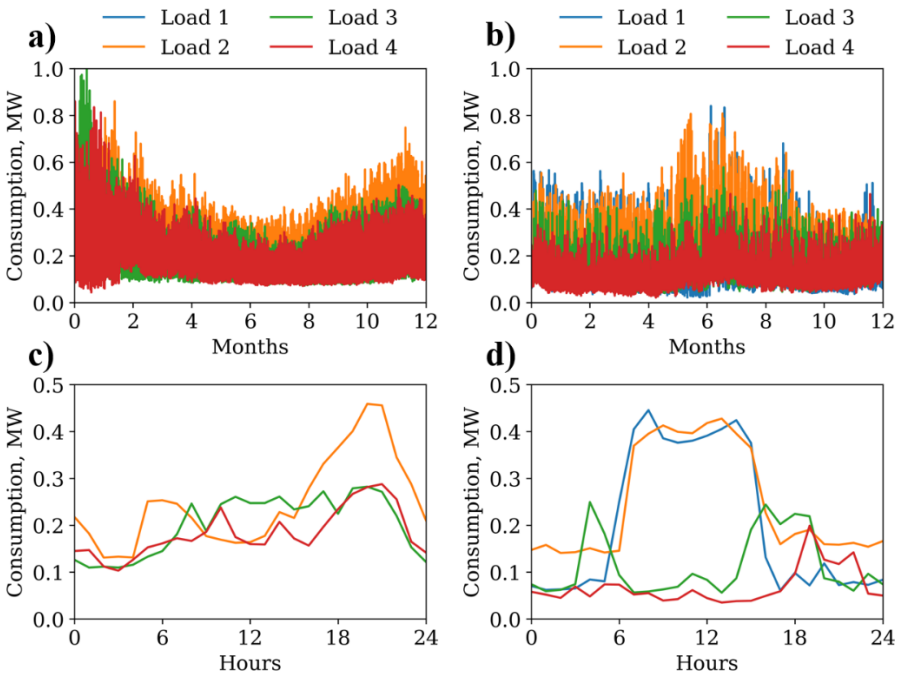


Figure 2.4. Load consumption profiles of "Smart meters in London" (a) and "Iowa distribution systems" datasets(b) for a year and one day (c and d) [71]

To model solar generation at some nodes of the test grid, representative solar radiation profiles from NREL [71] for a specific geographic position with a 1-hour timestep are used. Profiles from the places closest to London and Iowa are selected to model PV for the corresponding load profiles. As a result, seven different PV profiles for Iowa and London are prepared and scaled according to the conditions of the experiments (investigation for different PV sizes is given in section 2.6.2). Finally, the Darksky API [72] is used to obtain temperature data for the same geographic position and dates as net consumption.

The time step of 1 hour is chosen because the PV detection task does not require a fast response from the tool as for voltage/power regulation tasks and can rely on more available data with hourly resolution. The distribution grid considered in the study is the CIGRE-Network medium voltage distribution network [73], shown in Figure 2.5. However, PV detection tools, presented further, are trained to operate separately for each node, thus, they do not need to consider grid topology and its parameters and can be applied indiscriminately for any test systems. Thereby, the distribution grid is presented here just as an illustrative example.

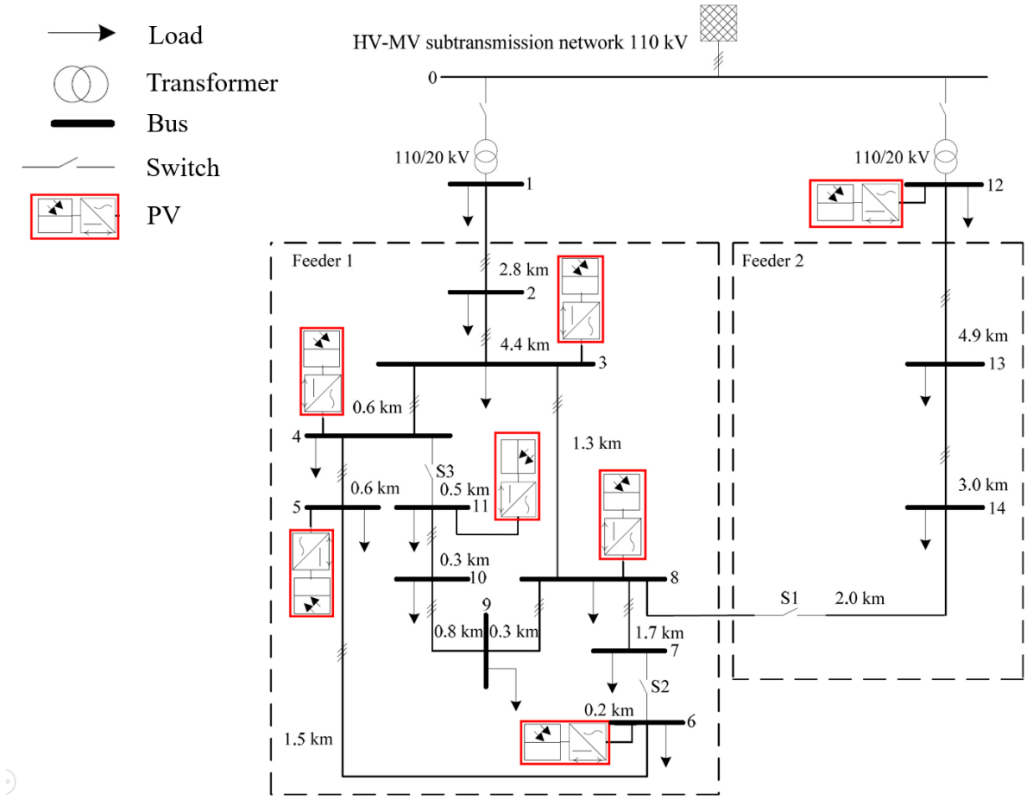


Figure 2.5. Medium voltage distribution grid with PV (in red) installed in seven random nodes [73]

The prepared load profiles were used to populate all 14 nodes of the distribution grid. The PV generation profiles were integrated into seven randomly selected nodes and scaled according to the consumption profiles of these nodes and the objectives of the experiments. The PV installed capacity in a node P_{nom}^{PV} as a percentage of the peak load at the same node max(P^{load}) is denoted λ (2.3).

$$\lambda = P_{nom}^{PV} / \max(P^{load}) \qquad \textbf{(2.3)}$$

The generated PV energy E^{PV} compared to the energy consumption of the same node E^{load} is denoted γ (2.4).

$$\gamma = E^{PV} / E^{load} \qquad \textbf{(2.4)}$$

The goal is to develop an approach that can correctly detect the PV connected to the nodes of a distribution grid, based on net consumption and temperature data of no more than two years.

2.4 Method A: Time series classification-based

2.4.1 Operation principle

The first approach relies on CNN [75]. The basic idea is that such neural networks, trained on the examples of net load profiles with and without PV, can notice specific patterns for the net profiles with PV and use them to detect similar profiles later. The overall implemented process is shown in Figure 2.6.

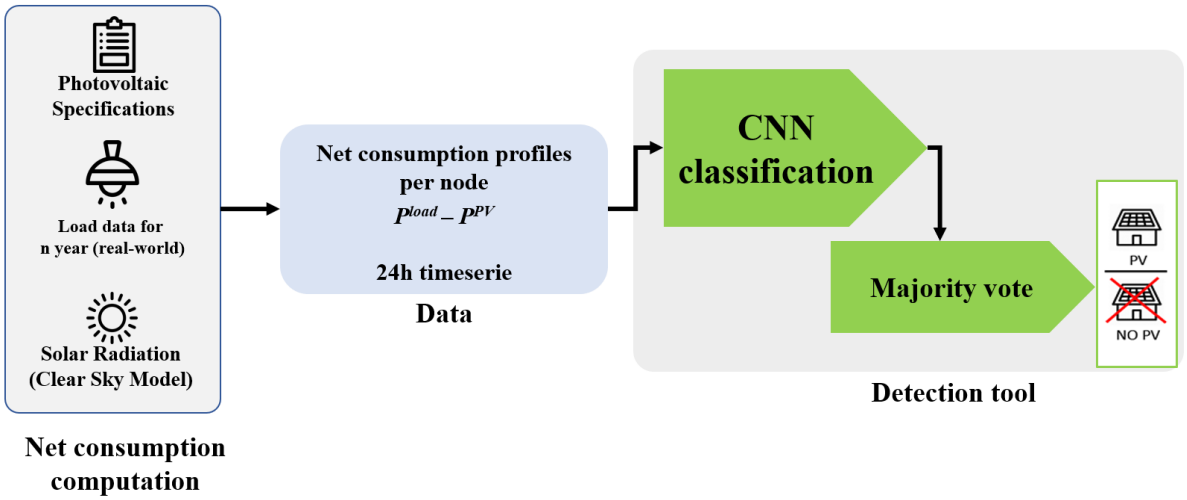


Figure 2.6. Operating principle of Method A [76]

The only input data is the simulated net consumption for the considered node (i.e., no access to the "behind the meter" disaggregated load and the generation profiles). Load data profiles (section 2.3) and solar radiation data with PV specifications are used to compute the net consumption for the experiment. Inputs of the NN are time series (TS) over 24h for the node. NN takes all time series of 24 h (i.e., time series of every day) for the considered period and classifies them into two categories: "With installed PV" (1) or "without" (0). Finally, the tool compares the detection results for separate days of the same bus and chooses the more frequent result as the final decision for this bus (i.e., "majority vote" is applied). Thus, CNN considers each univariate 24 h time series separately from the others. All models are aimed to be as simple as possible in terms of the number of trainable parameters (e.g., convolution kernel size, NN weights, and max-pooling kernel size).

The considered profiles cover the period from April to September, since the time series in these months are the most discriminable due to higher PV generation levels. For the "Iowa distribution systems" dataset, the algorithm uses three months of randomly chosen days for the training set and another three months for the test set. For "Smart meters in London", six months of the 2012 year are used as a training set, and six months of the 2013 year are used as a test set. The consumption profiles were first simulated with one set of nodes with PV (presented in Figure 2.5) and the second time with an "inverted" set of nodes (Table 2.1) to confirm the generalization of the methods.

Table 2.1.														
PV PRESENCE IN THE NODES OF THE CONSIDERED GRID														
Case	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
Standard	0	1	1	1	0	0	0	1	1	0	1	1	0	0
Inverted	1	0	0	0	1	1	1	0	0	1	0	0	1	1

The parameters of CNN are described in Table 2.2. These values correspond to the best-obtained performance during sensitivity analysis (i.e., the accuracy of CNNs with different hyperparameters was investigated). The results of the sensitivity analysis for different models are presented in Appendix 1.

Table 2.2.	
NN MODEL PARAMETERS FOR METHOD A	
Parameter	*Value*
Input	24h TS
Number of convolution layers	2
Number of fully connected hidden layers	1
Regularization	max pooling
Activation functions for hidden layers	Sigmoid
Number of nodes in the hidden layer	32
convolution filter (kernel) size	3
Optimizer	Adam
Loss function	binary cross entropy
Learning rate	0.001
batch size	200
Epochs	500

2.4.2 Results

The test accuracies for different rates of installed PV capacity λ are shown in Figure 2.7. The accuracy was calculated using the following expression:

$$Accuracy = \frac{N_{true}}{N_{results}} \qquad (2.5)$$

Where N_{true} is the number of correct detection results and $N_{results}$ is the number of all results for a given capacity λ. $N_{results}$ is equal to 28 in the considered case (detection results for each node in the standard and inverted cases - Table 2.1)

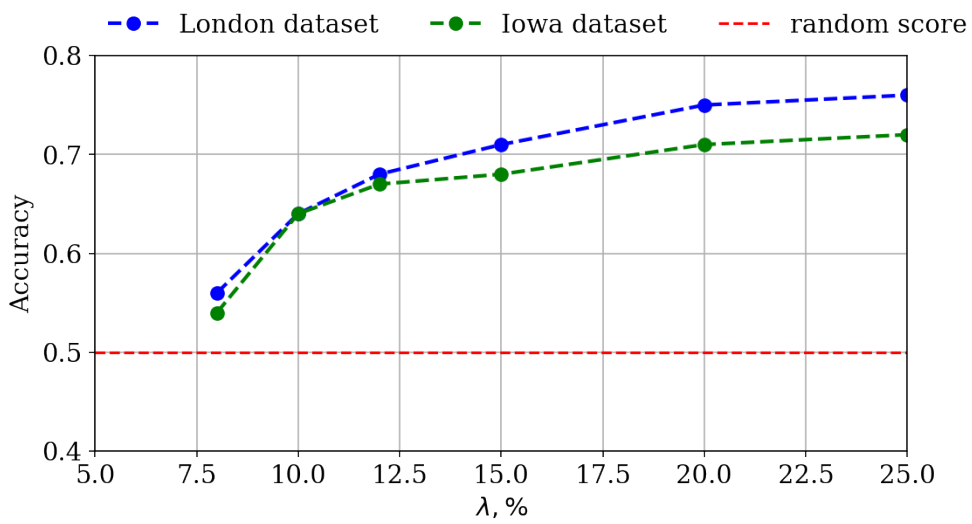


Figure 2.7. Accuracy test scores over six months of London dataset and three months of Iowa dataset, simulated with different rates of PV capacities (λ = 8%-25%)

For both datasets with λ = 8% of PV integration, the scores barely exceed 0.5 (which is a score for a random detection result). Thus, to see if the network can discriminate with higher PV production rates, the percentage is increased gradually to 25%. The increase of PV capacity above λ =25% makes the problem too trivial to solve. The CNN reaches an accuracy of 0.76 for the London dataset and 0.72 for the Iowa dataset for λ = 25%. The slightly worse results for the Iowa dataset can be explained by the presence of load profiles from commercial buildings (in six nodes out of fourteen). These profiles have a significantly higher consumption during sunshine hours when PV generation is at its maximum. Thus, the PV generation effect can be more easily confused with ordinary load fluctuations.

Examining why the CNN does not get an accuracy higher than 0.76, the reason can be found in the combination of model design and dataset. Figure 2.8 shows the London profile without PV and with installed PV, with a capacity of λ = 8% and λ =25% for three representative days of the considered period of 2013. It becomes clear that PV affects the load curve. However, the general pattern of smaller variations does not change. Hence, the profile remains similar to the load consumptions without PV. However, CNNs are designed to extract smaller patterns (depending on the filter size). Our basic assumption was that PV would affect small patterns of the net curves so that the CNN would detect them. It now becomes clear that this assumption does not hold. Also, CNNs detect similar variations independently from their absolute height. However, those absolute differences are necessary for classification (“with PV” and “without PV”) in the studied problem. This is the reason why CNNs are not able to perfectly classify nodes with and without installed PV.

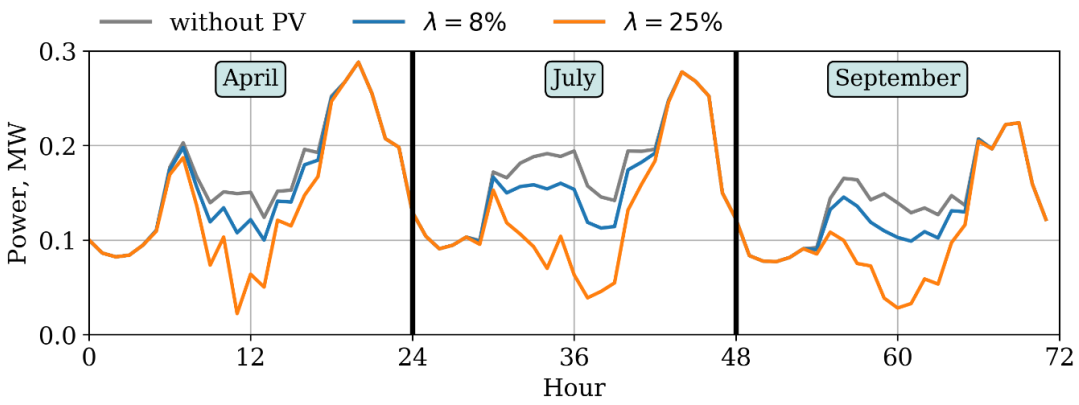


Figure 2.8. Net load profile of London dataset (2013) for three representative days of node 3 without PV, with PV of λ=8% and with PV of λ=25%

It is worth noting, that even though the three presented days follow each other in Figure 2.8, the profiles are not continuous and represent three separate days of the considered year. This also applies to all other plots with representative days in this manuscript.

2.5 Method B: Time series forecasting-based

2.5.1 Main principle

Method A has shown that the detection approach based on the shape of profiles does not show high accuracy, thus it is interesting to check whether the detection approach based on the absolute difference of profiles can be more promising. The second method is designed with this paradigm in mind. A second objective was to propose a more transparent method (avoiding the "black-box" effect, which is not acceptable for DSOs due to a lack of explainability) while still presenting a high accuracy. The objective is to use hourly net consumption profiles of the grid for two consecutive years (called year Y_{n-1} and year Y_n) and detect whether new PVs have been installed in the nodes of a distribution grid during the considered period (e.g., during the last year or month) compared to the previous year.

The developed tool consists of a conventional NN coupled with an analytical classification algorithm that operates separately for each node (i.e., rule-based detection here). The operating principle of the method is shown in Figure 2.9. The first part of the data (at the top) represents the year Y_{n-1} ("training set") and the second (at the bottom) the year Y_n, for which the recently installed PV shall be detected.

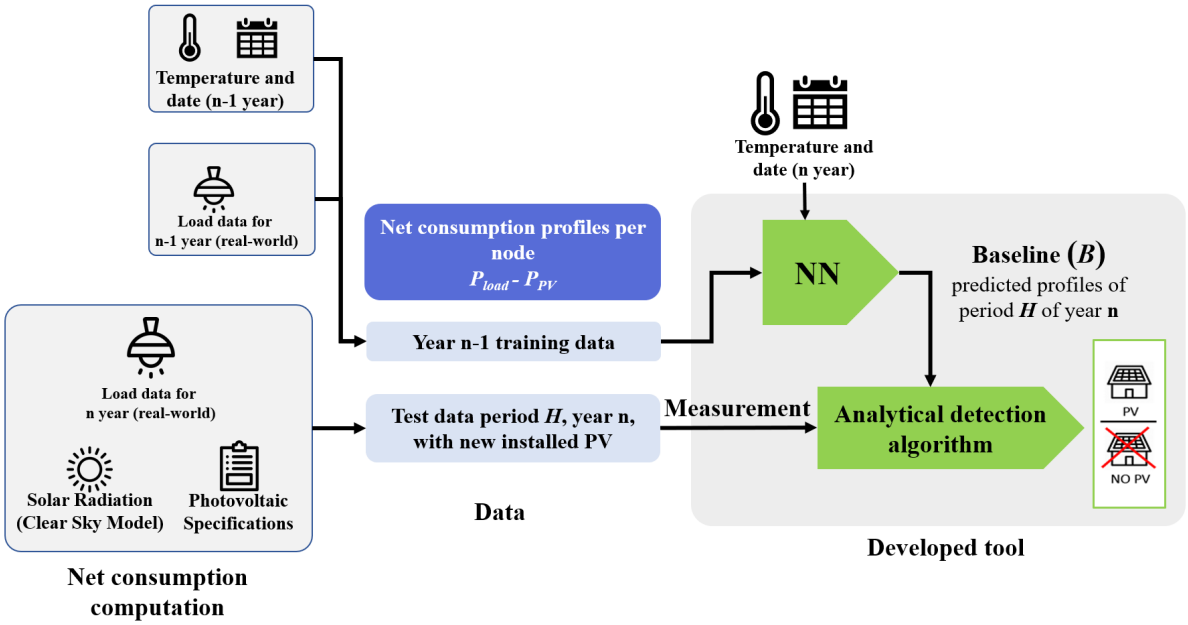


Figure 2.9. Principle of the method B [76]

Real consumption data (London project dataset from section 2.3 in this case) for Y_{n-1} are used to generate hourly energy consumption per node. Besides, the corresponding temperature, season, and timestamp are added for each sample point. Trained with those data for the year Y_{n-1}, the NN takes temperature data for the year Y_n and produces the estimated consumption for a given day of the year Y_n at an hourly resolution, assuming that no new PV has been added. This estimated (by NN) consumption, based only on the temperature (which, however, is produced after we know the real net consumption), is called the *baseline*, which is then fed into the classification algorithm. It is somewhat similar to a forecast.

The actual net energy consumption measured by the meters (which represents the difference between load and PV generation) for a given test period of year Y_n is also supplied to the classification algorithm and is called *measurement*. Those measurements for the experiment were generated by computing net consumption considering the consumption of households, photovoltaic specification, and solar radiation data.

Finally, the analytical detection algorithm compares the magnitudes of *baseline* and the *measurement* during the day and then during only the sunshine hours to detect PV units. If the difference between *baseline* and *measurement* values is higher during sunshine hours compared to night hours, the algorithm detects that new PV units have been added in the considered period.

2.5.2 Neural network for baseline estimation

After sensitivity studies of the NN used to estimate the *baseline*, five features (hour, season, temperature, weekend, holiday) were selected for NN that showed the strongest effect on *baseline* consumption estimation. The selected features were converted by hot encoding into 31 neurons of the first NN layer, as shown in Figure 2.10. Hot encoding is a process by which

categorical variables are converted into binary variables. If a data point belongs to a given category, then the corresponding component is set to 1, otherwise to 0.

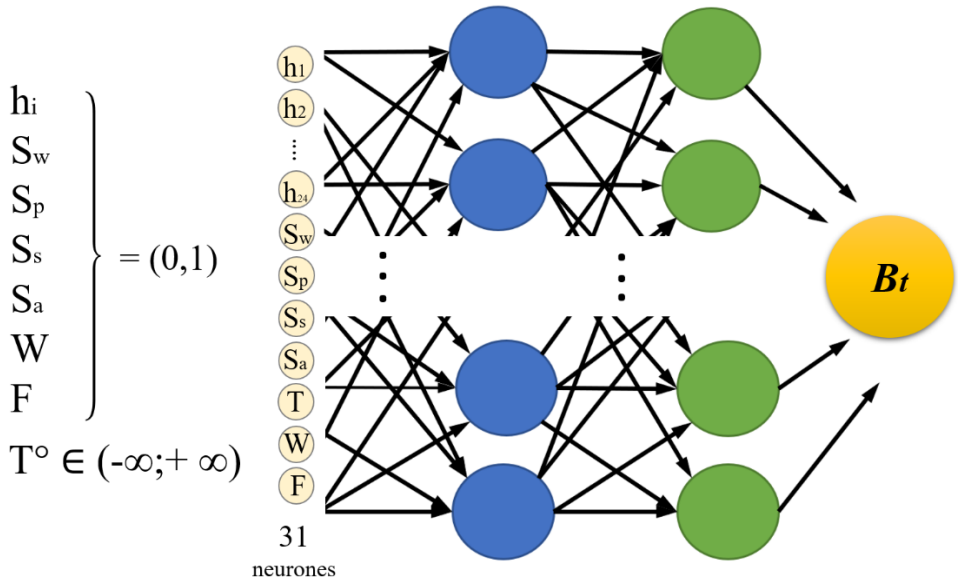


Figure 2.10. The neural network representation and its features

In the considered NN:

- 24 neurons to indicate at what hour of the day the simulation will be performed – h_i, where i = 1. ... 24 – binary values;
- Four neurons that refer to the season – S_w (winter), S_p (spring), S_s (summer), S_a (autumn) – binary values;
- The last 3 are temperature ($T°$ as a continuous value), weekend (W), and holiday (F) as binary values.

The NN architecture has then 31 neurons in the input layer, hidden layers, and the output layer with only one neuron, which computes the estimated consumption of the node for the chosen hour (B_t), given that no new PV is connected. The choice of the best parameters of NN is presented in section 2.6.1. The training time is between 10s and 20s per bus and 1000 epochs.

2.5.3 Analytical detection method

The analytical detection algorithm (rule-based) is presented in Figure 2.11. The core idea of the proposed method is that, in absence of PV generation, an average difference between the *baseline* (estimated net consumption without PV) and the *measurement* (real net consumption) values is roughly the same for hours of the day and of the night. Thereby, if this difference is higher during hours of the day, it is a sufficient reason to conclude that new PV has been installed in the considered (test) period - since the PV generation decreases the *measurement* values only during sunshine hours.

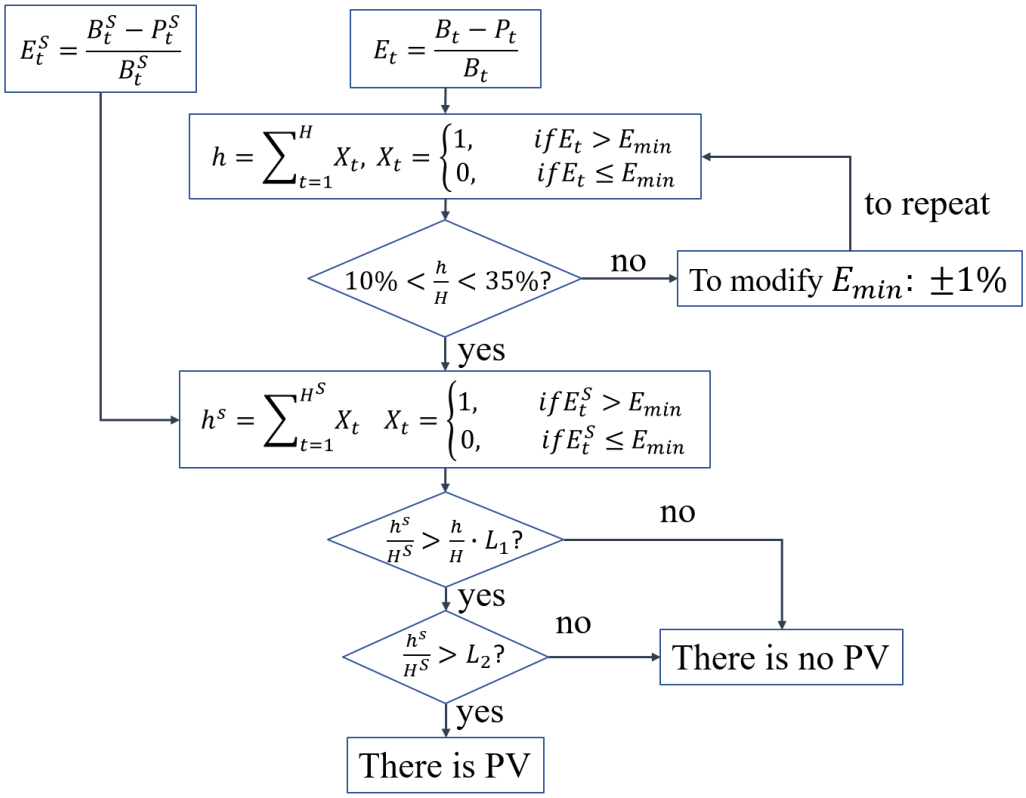


Figure 2.11. Synoptic of the proposed analytical detection algorithm

First, the algorithm calculates E_t (in %) as the difference between the *baseline* (B_t) and the *measurement* (P_t), for each hour t of the considered time horizon H (2.6):

$$E_t = \frac{B_t - P_t}{B_t} \qquad \textbf{(2.6)}$$

Then E^S is similarly computed as the relative difference between the *baseline* and the *measurement*, but only for "sunshine hours", H^S. That period is set to start at 8 a.m. and finish at 4 p.m. every day, as we can assume a noticeable solar radiation level whatever the season is within those hours. Thereby, the "sunshine time horizon" H^S is equal to 2,920 h for the whole year or less if $H < 8{,}760$ h.

The algorithm then enumerates the number of hours h when E_t exceeds the threshold value E_{min} =10% (obtained empirically). In other words, it counts the number of hours during which the *baseline* is at least 10% higher than the *measurement*. If the number of these hours h is outside a pre-defined interval of [10%; 35%] of the total number of hours H, the algorithm adjusts the threshold E_{min} (by steps of 1%), to avoid an overestimation (when $h/H > 30\%$) or underestimation ($h/H < 10\%$) by the NN, until h falls into the limits. This procedure can be considered as a vertical shift of the full *baseline* profile along the axis of power to correct the error of NN. E_{min} is the only threshold that needs adaptation. The values of the interval were obtained empirically through sensitivity studies that determined the best percentages.

Once E_{min} is set, the algorithm calculates h^S, the number of hours during which the *baseline* is higher than the *measurement* during the period H^S, by at least E_{min}.

Then the algorithm checks the first condition:

if h^S/H^S is at least $L1$ times higher than h/H. Where $L1$ is the first fixed limit, which is higher than 1. Thus, it checks that the hours when the *baseline* is higher than the *measurement* are more frequent during the sunshine period H^S than during the whole period H. If there is no new installed PV, h^S/H^S and h/H should be roughly similar. If PV were installed, then h^S/H^S will be noticeably higher than h/H.

If the first condition is true, the algorithm checks a second condition:

if h^S/H^S is higher than the second fixed limit $L2$, which is lower than 1. That is, at least during $L2 \cdot H^S$ hours the *baseline* is greater than the *measurement* by E_{min} %. This is necessary to avoid situations where the first condition is true but there is no PV on the node because the proportion of h^S/H^S is too small.

Finally, if both conditions are successful, the algorithm concludes that there is a new PV installed in this node.

2.5.4 Sensitivity analysis to hyperparameters

Hyperparameters of neural networks are parameters that directly impact the training process and the final results. To correctly choose the hyperparameters of the neural network which is used to estimate the consumption profile in Method B (i.e., to produce the *baseline*), a sensitivity analysis of the algorithm was performed on the net load profiles of the six most sunny months of the year.

First, the neural network structure was fixed at two hidden layers of 15 neurons each. Then the impact of the learning rate α_{LR} and the number of epochs (network updates – training iterations) were investigated. For each set of chosen hyperparameters, the accuracy was calculated according to (2.5) for all 28 net load profiles from Table 2.1. Additionally, each neural network was trained with 3 different seeds (initializations), and the average accuracy of the three results was taken. The results for two different optimizers, Adam and stochastic gradient descent (SGD), are shown in Figure 2.12.

Optimizers are mathematical functions that can optimize the weights of NN given the gradient. Most optimizers are built on the idea of gradient descent, a greedy approach of iteratively decreasing the cost (loss) function by following a gradient. SGD is a variant of gradient descent and one of its basic forms [77]. SGD subtracts the gradient multiplied by the learning rate α_{LR} from the weights of each layer $\boldsymbol{\theta}^{\mathbf{i}}$ (2.2). The main difference with gradient descent is that it uses only a subset of random samples (batch) for the update instead of the whole dataset. This makes it possible to significantly reduce the training time and obtain the same accuracy with a relatively low learning rate.

The adaptive optimization algorithm "Adam" is based on the concept of momentum and an adaptive learning rate, which varies with the size of the gradient [78]. Momentum is often referred to as rolling down a ball because it is conceptually equal to adding velocity. Each weight of neural network θ_i is modified through a momentum term υ_i, which is calculated as the moving average of the gradients (2.7). The momentum term γ can be seen as air resistance or friction, which decays momentum proportionally. The momentum accelerates the training process.

$$\upsilon_i = \gamma\upsilon_i + \alpha_{LR}\frac{\delta L_{\text{cost}}}{\delta\theta_i} \qquad (2.7)$$
$$\theta_i = \theta_i - \upsilon_i$$

Where L_{cost} is the cost function and α_{LR} is the learning rate.

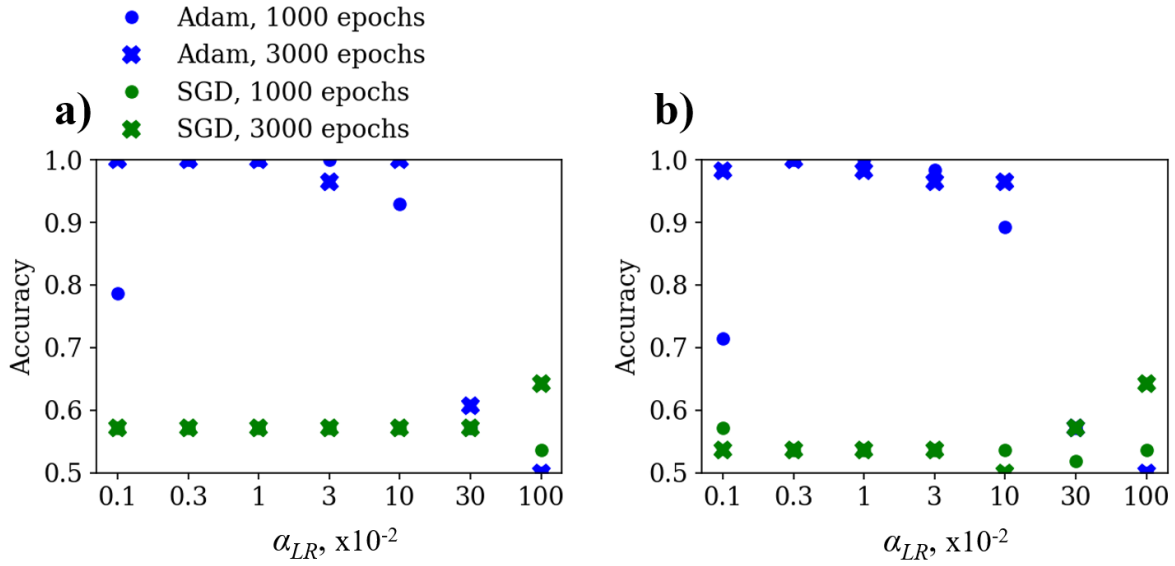


Figure 2.12. Accuracy results for Adam and SGD optimizers with different learning rates and PV capacity λ of 7.5% (a) and 7%(b)

The results in Figure 2.12 (a) are presented for installed PV capacity λ of 7.5%. SGD shows poor accuracy for all learning rates, varying in the range from 0.536 to 0.643. Results are then worse than those of method A and close to the accuracy of a random choice of 0.5. However, the Adam optimizer shows significantly better results (except for large α_{LR}). Its accuracy is equal to 1 for α_{LR} of 0.001, 0,003, 0.01 and 0.1. Training with 3000 epochs gives similar or better results (except α_{LR} of 0.03) than 1000 epochs. Due to the 100% accuracy for four different hyperparameters' choices, the difficulty of the detection task can be increased by reducing the PV capacity λ to 7%. In this case (Figure 2.12 (b)), the resulting accuracy slightly drops for both the SGD and Adam optimizers. However, the Adam optimizer still has an accuracy of 1 for the α_{LR} of 0.003. The impact of the number of epochs is ambiguous in this case. Results are slightly better with shorter training of 1000 epochs for α_{LR} of 0.01 and 0.03.

However, longer training of 3000 epochs allows better results to be obtained with α_{LR} of 0.001 and 0.1. Thereby, the Adam optimizer and learning rate α_{LR} of 0.003 are chosen for further tests.

Next, the impact of the NN structure is investigated. The results with one and two hidden layers with a different number of neurons in each are presented in Figure 2.13.

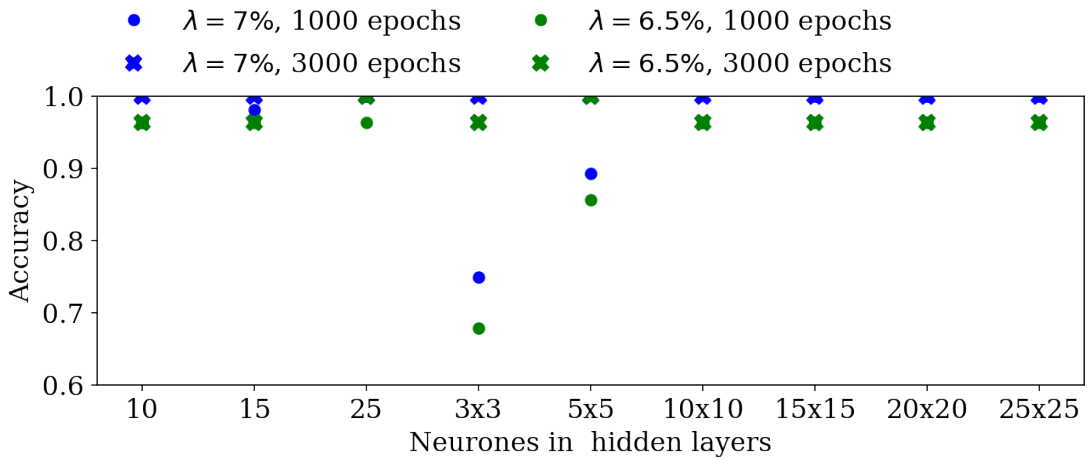


Figure 2.13. Accuracy results for different neural network structures and PV capacity of 6.5% and 7%

All structures trained during 3000 epochs show an accuracy of 1 for λ = 7%. Thus, the PV capacity is decreased to λ = 6.5% to increase the complexity of the task. In this case, only two structures still show perfect accuracy with 3000 epochs of training – one hidden layer with 25 neurons and two hidden layers with 5 neurons each. However, the 5x5 structure shows significantly worse results for a 1000 epochs training. Thus, one hidden layer with 25 neurons is chosen as the best structure.

The lower number of hidden layers allows the training to be accelerated, as can be demonstrated in Figure 2.14. In this figure, the learning losses during the training of a neural network with one hidden layer of 25 neurons and with two hidden layers with 25 neurons in each are shown. The maximum power of bus 1, bus 2, and bus 3 are 430 kW, 500 kW, and 500 kW, respectively.

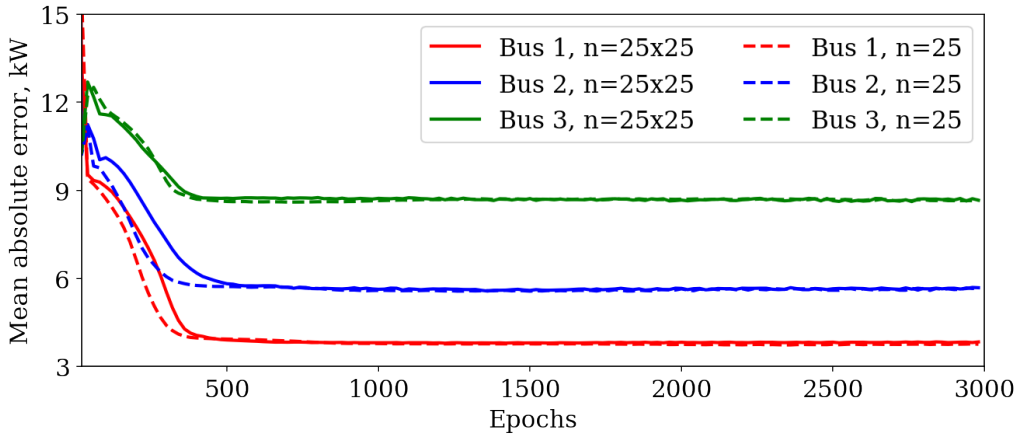


Figure 2.14. Learning loss during the training of bus 1-3 for neural networks with the structure of 25 and 25x25 neurons

As can be seen, for all the buses presented in Figure 2.14, a NN with a single hidden layer needs fewer epochs to be trained and shows similar or slightly lower error in the end. Too many hidden and neurons layers increase training time and also tend to overfit, i.e., poorly generalize the function, which is too biased to the training set. However, a too small number of neurons may not be enough to reproduce the necessary function. Thus, one hidden layer with 25 neurons presents the trade-off in the considered case.

Finally, the accuracy of the algorithm concerning the values of the limits *L1* and *L2* is studied, and the results are presented in Figure 2.15.

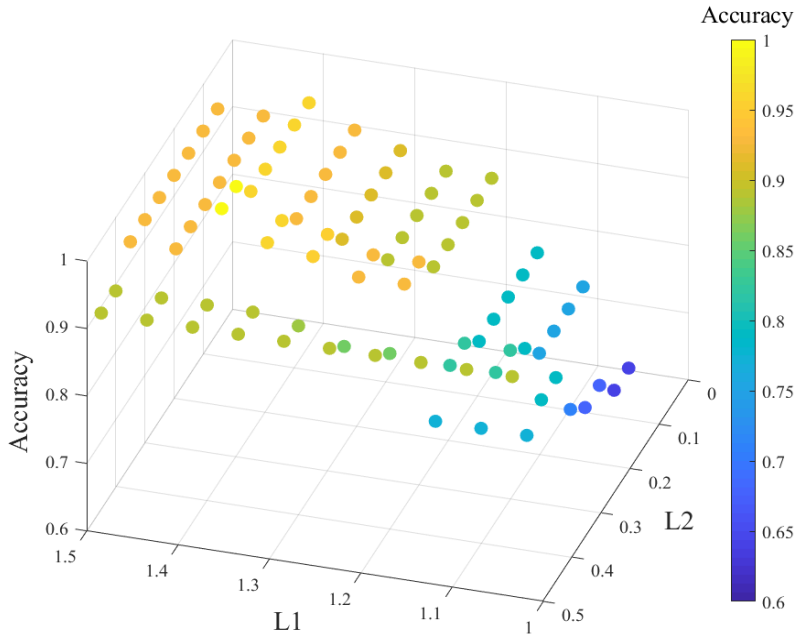


Figure 2.15. Accuracy results for different values of L1 and L2 of the analytical part of algorithm B

For the low values of *L2*, the accuracy of detection increases as *L1* increases from 1 to 1.4, and then slightly drops again. We can say that for low *L2*, the algorithm only checks whether the hours when the *baseline* is higher than the *measurement* are more frequent during the sunshine period than during the entire period. From the results, it can be concluded that for buses with PV, the *baseline* is higher than the *measurement* about 40% more often during the sunshine period.

The accuracy also increases with the increase of *L2* up to a value of *L2*=0.35 and then drops. For low values of *L1*, the algorithm only checks if the number of hours when the *baseline* is greater than the *measurement* by the given threshold E_{min} is sufficient. From Figure 2.15, it can be concluded, that a *baseline* of the buses with PV is greater than the *measurement* approximately 30 to 35% of the time. Thus, two couples of parameters showed the best accuracy: *L1*=1.4, *L2*=0.35, and *L1*=1.4, *L2*=0.3. However, the accuracy for *L2*=0.4 (which is close to 0.35) is noticeably lower than for *L2*=0.25 (which is close to 0.3), thereby *L1*=1.4, *L2*=0.3 are thus selected for the rest of the study assuming that these parameters allow the algorithm to be more robust and accurate.

All chosen NN parameters used in Method B are shown in Table 2.3. In general, the accuracy of Method B is not too sensitive to hyperparameters, because it does not need a perfect and accurate prediction of the absolute value of the load due to the adjustable threshold E_{min} used by the analytical detection method (section 2.5.3). Thus, Method B rather requires NN to have a just correct profile shape that is easier to reproduce.

Table 2.3.

PARAMETERS FOR NN AND ANALYTICAL ALGORITHM

Parameter	*Value*
Input	31 neurons
Number of fully connected hidden layers	1
Activation function for hidden layers	Sigmoid
Activation function for the output layer	Sigmoid
Number of nodes in the hidden layer	25
Optimizer	Adam
Loss function	Mean absolute error
Learning rate α_{LR}	0.003
Epochs	3000
L1	1.4
L2	0.3

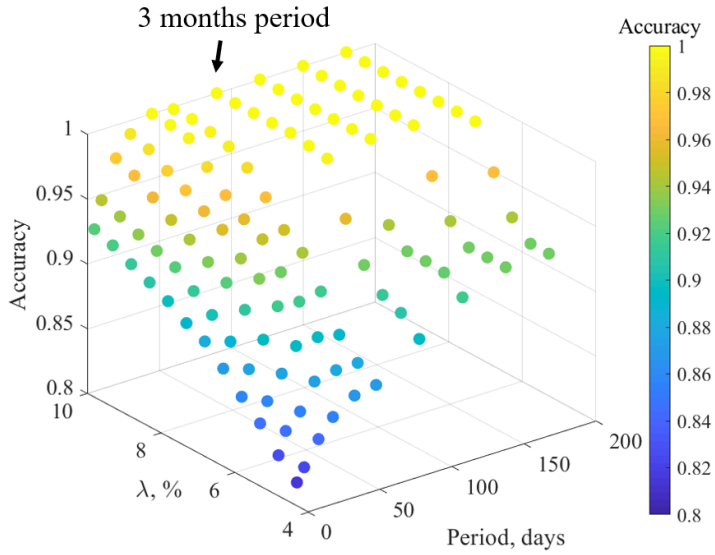
2.5.5 Detection results of method B for different PV sizes and observation periods

It is possible to increase the sensibility of the tool by analyzing only months with high levels of solar radiation instead of the whole year – i.e., while adapting the considered test period. To confirm this hypothesis, two different cases are considered: the application of the

tool with measured meter data over the six sunniest months (April to September) and over the six least sunny ones (October to March).

Method B with the hyperparameters chosen in the previous section is trained and its average accuracy is calculated for all possible periods of *d* test days over 6 months on a rolling basis. For instance, for *d* =40, the average accuracy of 142 possible 40-day periods between April and September is calculated. The dependence of the average accuracy of the tool concerning the considered test period and to λ for all nodes of the grid is presented in Figure 2.16 for the months of April-September and in Figure 2.17 for the months of October-March.

The average accuracy from April to September varies between 0.8 (λ = 4.5%, over a period of five days) and 1.0 (λ >7%, over a period of more than three months). The results show that the average accuracy for the sunniest months is higher for longer periods because this allows smoothing out the impact of cloudy days when PV generates less energy. As previously observed, it is also natural that a higher λ facilitates the detection of PV, so the average accuracy is also higher.



April - September

Figure 2.16. Average accuracy dependence from April to September with Method B

The results are less performant when the test periods (i.e., successive days) occur from October to March (Figure 2.17). The average accuracy while applying the detection tool over those months varies between 0.87 (λ =10.0%, over a period of five months) and 0.52 (λ = 4.5%, over a period of six months). For λ < 8.5% the average accuracy is higher for shorter periods because during these months the PV systems do not generate enough power to be detected, but there are still a few days with high solar radiation when detection is possible.

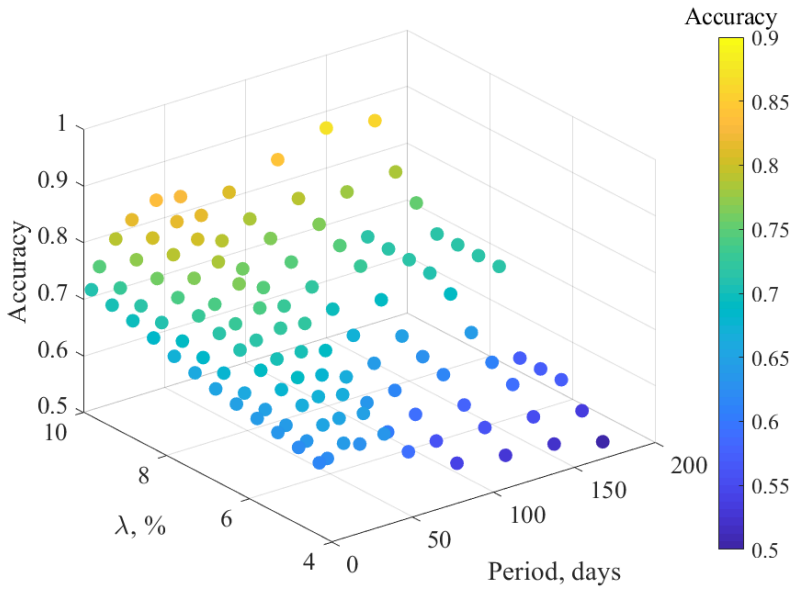


Figure 2.17. Average accuracy dependence from October to March with Method B

From Figure 2.16 it can be concluded that, on average, three test months are required to obtain the maximum detection accuracy. Considering the test periods of the three sunniest months of the year (from May to July), a 100% accuracy of the method on the test system can be achieved with λ = 6.2% and γ = 3.1%. An example of the standard case of PV location (Table 2.1), is shown in Table 2.4. The results of the metrics that are used to detect PV (Figure 2.11) are also presented in this table: the share of hours when the *baseline* is greater than the *measurement* h/H, the same share only during sunshine period h^S/H^S, and their relation $\frac{h}{H}/\frac{h^S}{H^S}$. All PV detection results are correct in this case. The worst and the best $\frac{h}{H}/\frac{h^S}{H^S}$ results for the buses with PV are highlighted in red and green color, correspondingly.

Table 2.4.

ALGORITHM METRICS FOR PV DETECTION

Bus	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*
γ, %	0	3.3	3.2	3.3	0	0	0	3.1	3.0	0	3.0	3.7	0	0
h/H	11.7	34.4	33.6	34.6	15.2	21.6	20.7	33.5	34.0	14.9	33.5	32.8	18.8	33.1
h^S/H^S	12.6	50.6	59.3	60.1	13.4	22.4	23.4	57.0	62.3	11.7	54.0	46.0	23.8	35.8
$\frac{h}{H}/\frac{h^S}{H^S}$	1.08	1.47	1.76	1.74	0.88	1.03	1.13	1.70	1.83	0.79	1.61	1.40	1.27	1.08
PV	no	yes	yes	yes	no	no	no	yes	yes	no	yes	yes	no	no

The value of $\frac{h}{H}/\frac{h^S}{H^S}$ for node 12 is equal to 1.4, which is close to the limit of PV detection. But the sensitivity of the detection depends on consumption profiles (for the two considered years). All buses presented in Table 2.4 have installed PV with the same λ of 6.2%,

but their resulting $\frac{h}{H} / \frac{h^S}{H^S}$ ratio varies from 1.4 to 1.83. Thus, even though the detection tool will not be able to detect PV in node 12 for $\lambda < 6.2\%$ (ratio of 1.40), it will still successfully detect the PV on node 9 for $\lambda > 3.9\%$, which corresponds to $\gamma > 2.1\%$ (ratio of 1.83).

These results are significantly superior to similar solutions presented in the literature. For instance, the solutions presented in [31] and [34] use net consumption profiles from Pecan Street Dataport [79] for testing. In this dataset, the PV penetration level γ is 56%. The accuracy of the tools is 98% and 95.5% for [31] and [34], respectively. The accuracy of the tool from [36] is 94.8%. It uses measurements from a real-life setup in the region of Basel, Switzerland, with a PV penetration level of $\lambda = 73\%$. The disadvantage of the PV detection tool proposed in this chapter, which allows detecting such small PVs compared to other solutions, is the need for a consumption profile before the PV was installed.

It should be noted that the sensitivity of the proposed tool depends on consumption profiles (for two years). Thus, a sudden change in consumer behavior, such as increased consumption during sunshine, may affect performance. However, if the number of inhabitants changes, this should not have a significant impact on the results, because the algorithm can compensate for this by adjusting the threshold.

2.6 Conclusion

In this chapter, two different methods have been proposed for detecting PV installations in a distribution grid without connection agreement. Both approaches use only smart meter and temperature data which makes them data-sufficient, model-free, and simple. This is not the case with most solutions from the literature [31]-[36], that require specific data, such as satellite images or solar radiation for specific spots. While the first method (named A) explores convolutional neural networks (CNNs) with a wide range of architectures and settings, Method B is more transparent and combines a conventional Multi-Layer Perceptron with an analytical classification algorithm.

The overall approach of using neural networks for such classification performs well on the considered problem, as the network can be trained offline within minutes and can then analyze any period within seconds.

Sensitivity analysis of detection accuracy to different magnitudes of hyperparameters was performed for both solutions, and the best parameters were used to run these algorithms on a 14-bus distribution grid with randomly installed PV. It was found that the forecasting tool works better than the direct classification. It is mainly explained by the fact that neural networks are a lot better at predicting values (method B) than at learning patterns and discriminating similar curves that have been vertically distorted by PV (method A). Thereby the PV detection approach of Method B can be chosen as the best performing method. Moreover, the principle of this method is simple enough not to consider it completely a “black box”. Indeed, the neural network is used for net consumption forecasting and not for the detection of PV itself. This also

makes the algorithm more robust, especially regarding the yearly consumption increase. Moreover, the proposed detection approach operates separately for each node, thus, it does not need to consider grid topology.

Method B displays a high accuracy, which however depends on the considered period, the installed capacity, and the energy production of the installed PV generation. Thus, it is necessary to find a trade-off between the volume of available data and the required accuracy of the solution. It was found that the highest performances could be reached when considering the three sunniest months of the year. In this case, the algorithm can detect PV units in any of the 14 buses of the considered distribution grid, if the installed PV capacity is at least 6.2% of the maximal consumption of the same bus. Thus, it can be stated that Method B represents a good basis for PV detection in distribution grids, offering many opportunities to be expanded further upon in future work. However, it may also be interesting to investigate the performance of the tool with other load profile databases (e.g., with commercial buildings) to prove the generalization of the results.

3 Reinforcement Learning for Fast Voltage Control

3.1 Context of the study

One of the main tasks of distribution system operators (DSO) is to maintain voltage levels within specified limits, typically between 0.95 and 1.05 normalized per unit (p.u.) under normal conditions. DSOs can use different levers to regulate the voltage. For instance, conventional regulation consists of the control of On-Load-Tap-Changer (OLTC) or capacitor banks [80]. In recent years, new control capabilities emerged due to the increasing penetration of Distributed Energy Resources (DER) and demand-side management. Among those controls are shifted in prosumers consumption, regulation of solar and wind power plants, and charge and discharge of storage systems (fixed or mobile like electric vehicles) [81]. All these options denoted as "flexibilities", can be used for a wide range of grid operations apart from voltage regulation, such as loss minimization or peak load shaving. Typical control of those flexible resources relies on optimization/model-based approaches. These approaches can consist of looking ahead and/or near real-time decision-making processes. However, they require consumption values from the meters as well as grid data (i.e., topology and line impedance) to estimate the future state of the systems. Those data are typically integrated into model predictive control (MPC) strategies that rely on time series forecasts for the bus power (i.e., load/generation). However, optimization-based algorithms can require significant time to compute the controls, especially when considering uncertainties mitigation as a core feature, e.g., stochastic optimizations [82], or when greater numbers of resources shall be controlled.

Opposite to model-based approaches, data-driven approaches may then be interesting in some conditions, which are increasingly present in the evolving grids that we know. In this category, artificial intelligence (AI), and Neural Networks (NN) in particular, offer a relevant alternative to deal effectively and rapidly with such tasks as voltage control [42]-[44]. However, these data-based strategies require time and computational resources, for the training phase (a complete set of training examples can be processed hundreds or thousands of times [83]). In addition, NNs are supervised learning, i.e., they are biased similarly to MPC control, which is used to provide training examples (the process is shown in Figure 1.12). Thus, NN can encounter similar problems as MPC approaches and are not immune to parameter uncertainty or parameters drifting over time. Reinforcement learning (RL) algorithms can overcome this drawback, as they do not need the expected outputs of the controller, but only an assessment of the quality of the performed controls.

As presented in section 1.3.2, the algorithms presented in the literature mainly rely on bus power estimation (i.e., load and/or generation) or prediction from an external tool [45]-[51]. In this chapter, RL algorithms are implemented to handle the problem of voltage control by using only LV/MV node measurements or their estimations and limited flexibility levers. Being pre-trained, those algorithms require an extremely short time to compute the controls at every execution (<<1s). This can be advantageous for voltage control with a high proportion of RES that can rapidly change their output power.

A contribution of this chapter is to compare the efficiency of two different RL algorithms, Twin Delayed Deep Deterministic Policy Gradient (TD3PG) [84] and Proximal Policy Optimization (PPO) [85], that correspond to two main classes of RL algorithms, namely 'off-policy' and 'on-policy' methods. The main reason for the choice of these methods and a description of their principles are discussed first in this chapter. Then a small distribution system with DER is modeled and used to assess the performance of the different approaches. In particular, the sensitivity of the algorithms to their main hyperparameters is conducted. Finally, the results of the proposed voltage control over a full year are presented before concluding the chapter.

3.2 Reinforcement learning

3.2.1 Preliminary definitions

As already mentioned in Chapter 1, typical supervised learning relies on a training set of examples provided by a knowledgeable external supervisor [89]. Reinforcement learning, on the contrary, learns from its own experience [94] with the successive exchange of information with the controlled system. The main elements of RL training (Figure 3.1) are:

- the *Environment*: the considered system represented at time t by state variables stored in the vector s_t;
- *Agent*: the controller that takes actions a_t, depending on s_t, which creates the next state s_{t+1} once the actions are processed by the environment.
- *Reward* r_t: the controller objective metric calculated from s_{t+1}, to assess the relevance of the actions a_t taken.

Each transition sample defined as $[s_t; s_{t+1}; a_t; r_t]$ is then saved into a table ("replay buffer"). During the learning phase, the RL algorithm simultaneously uses several samples from this table. In off-policy algorithms, the replay buffer can store transition samples for a long time or even during the entire training (and later use a random set of these samples for the training). For on-policy algorithms, the replay buffer removes transition samples after every epoch (i.e., time horizon), i.e., on-policy algorithms use only the results of the last epoch. The final goal of the RL algorithms is to find an optimal policy π, which is a mapping between states and actions, $a_t = \pi(s_t)$. The policy is generated and adjusted along the training so that it maximizes the

cumulative reward over a time T, specified by the user. As discussed in Chapter 1, deep RL algorithms use NNs to approximate state and action spaces and thus they do not need the tables as in tabular RL algorithms. Nowadays, tabular RL algorithms are practically not used, except for small discrete states and action spaces.

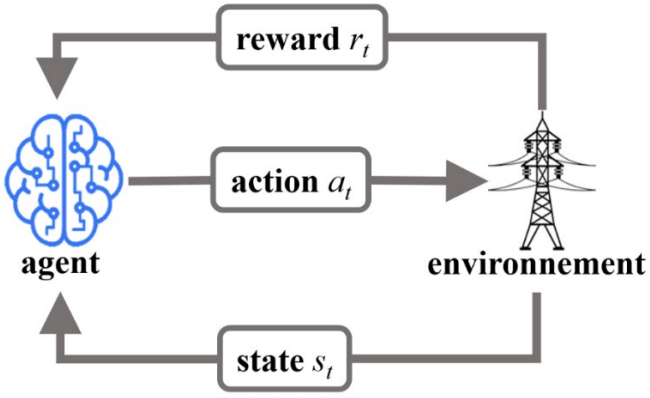


Figure 3.1. Reinforcement learning principle

The two main types of RL algorithms are:

i) on-policy learning (for instance Advantage Actor Critic (A2C), Trust Region Policy Optimization (TRPO), and Proximal Policy Optimization (PPO)), where a policy is trained based on the results of its actions.

ii) off-policy learning (for instance Deep Deterministic Policy Gradient (DDPG), Soft Actor-Critic (SAC), and Twin Delayed DDPG (TD3PG)), which learns about one policy (denoted π_1), while the reward and observations are generated with the actions sequence from another policy (denoted π_2).

Both types of RL algorithms have their advantages and disadvantages. Off-policy algorithms tend to use data more efficiently. Thanks to their replay buffer, they can use the same transition samples multiple times, thus they typically need fewer epochs and samples to train compared to on-policy algorithms. In addition, during the training phase, the algorithms can take random samples from large replay buffers, thus limiting correlations in the observations. That allows them to have a better generalization and lower the probability to get stuck in a local optimum during the training. The second advantage of off-policy algorithms lies in their principle – they do not require to choose actions according to their current policy to be trained but can learn from any other actions that could be randomly generated or inputted by a third party (e.g., user/operator). This is especially important in cases where incorrect actions can lead to dangerous situations (e.g., voltage drops. However, off-policy algorithms have their drawbacks. They are usually less accurate than on-policy algorithms if their policy is too different from the policies that were used to collect transition samples. They are also more often prone to divergence. Thus, the on-policy algorithms are preferred if stability during the training is a key feature imposed by the user or the problem to solve [86].

In this thesis, TD3PG is chosen as the baseline off-policy algorithm thanks to its technical improvements over DDPG [84] which are described more in detail in section 3.2.2.2.

Also, it can outperform SAC in higher dimensions due to its deterministic policy which is confirmed by OpenAI gym continuous control tasks tests [84]. As for on-policy algorithms, PPO is chosen as it is simpler to implement than TRPO, which has hard constraints to limit the size of the gradient step, i.e., that limits the divergence of the trained policy over successive iterations. PPO, on the contrary, implements soft constraints in the objective function, i.e., these constraints can be violated sometimes during the training process, but this allows us to get a good balance in the speed of the optimization. Also, PPO empirically seemed to perform at least as well as reference TRPO and A2C [85].

Both chosen algorithms are model-free, i.e., they don't try to create any model of the environment and learn directly a policy from their experience. Both algorithms are prepared in python by using a Python RL package, from the educational resource SpinningUp of Open AI [87]. To communicate the environment with RL algorithms, an open-source Python library Gym [88] was used. It provides a standard API (Application Programming Interface) to communicate between learning algorithms and environments. Gym implements the classic "agent-environment loop" introduced above. The agent performs actions in the environment, observes how the environment's state changes and obtains a reward. One such action-observation exchange is referred to as a 'timestep'. After some timesteps (an epoch), the environment may enter a terminal state. In that case, Gym resets the environment to a new initial state. The Gym also allows specifying the format and limits of valid actions and valid observations.

3.2.2 Learning algorithms

3.2.2.1 General principle

Both RL algorithms, TD3PG and PPO, contain two parts: an "Actor" part, which learns an optimal policy, and a "Critic" part, which is used to predict a return – i.e., a discounted sum of future rewards (as presented below). A discounted sum of future rewards lets the algorithm know that the reward in the future is less important than the same immediate reward values. Thanks to the "Critic" part, the "Actor" part can estimate how good its action will be. Both "Actor" and "Critic" parts correspond to neural networks (Figure 3.2), and the training consists in optimizing their parameters (i.e., weights and biases). However, note that the "Critic" part is different for the two algorithms.

- For TD3PG, the "Critic" part computes an action-value function (or shortly Q-function) $Q_t^{\pi}(s_t, a_t)$, which is defined as the expected future reward (algorithm estimation of all rewards in the future) starting at state s_t, and taking a sequence of actions a_t that follows a given policy π (Figure 3.2 (b)) (3.1) [89]:

$$Q_t^{\pi}(s_t, a_t) = \sum_{k=0}^{\infty} \mathrm{E}_{\pi}\left[\gamma^k \times r_{t+k+1} \mid s_t, a_t\right] \qquad \textbf{(3.1)}$$

where E_{π} is the expectation, γ is the discount factor that determines the importance of future rewards by decreasing it with each timestep (between 0 and

1) [88]. $\gamma = 1$ corresponds to an environment in which the reward is equally important regardless of the time step. $\gamma = 0$ corresponds to an environment in which only the reward for the first timestep is important. The discount factor of less than 1 allows using an infinite sum with a monotonic decrease of the weighted reward.

- In PPO, the "Critic" network computes a state-value function $V_t^{\pi}(s)$, which is defined as the expected future reward starting at state s_t and following a given policy π (Figure 3.2 (c)), (3.2) [89]:

$$V_t^{\pi}(s_t) = \sum_{k=0}^{\infty} \mathrm{E}_{\pi}\left[\gamma^k \times r_{t+k+1} \mid s_t\right] \qquad (3.2)$$

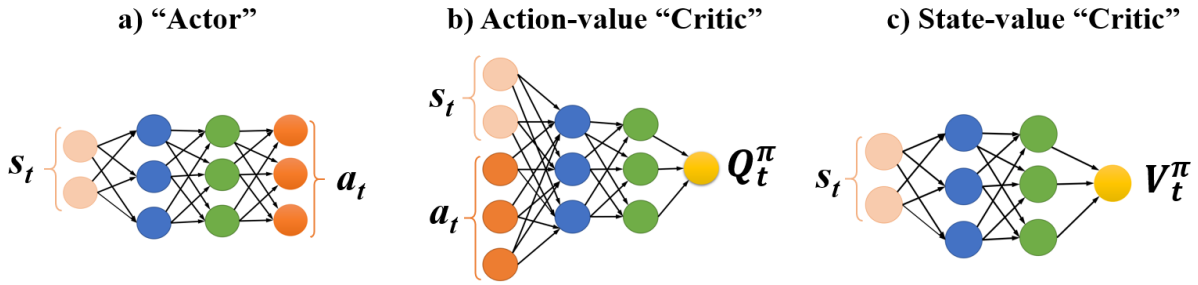


Figure 3.2. Main NN types in the considered RL algorithms – a) "Actor" b) "Critic" for action value function c) "Critic" for state value function

The "Actor" model computes actions a_t for a given observed state s_t. The training of this "Actor" consists of updates of its network parameters through gradient ascent to get output a_t which leads to a higher expected return, computed based on approximations of the "Critic" network. Thus, the algorithms concurrently learn a value function and a policy. They use the transitions obtained with the "Actor" policy to learn the value function, which is in turn used to learn the policy. Such interactions of "Actor" and "Critic" networks allow us to gradually get better results and reach a near-optimal policy π.

3.2.2.2 TD3PG algorithm

TD3PG is a sample-efficient, model-free, off-policy algorithm[84]. TD3PG works with six neural networks in total, four of them acting as "Critic" and two as "Actor". All six neural networks of the TD3PG algorithm can be divided into two categories with two "Critic" and one "Actor" in each: "Target" and "Model".

Overall, TD3PG uses its estimation of Q-function for timestep $t+1$ $Q_{t+1}^{\pi_{t\arg}}(s_{t+1}, a_{t+1})$ ("Critic target") to update its other estimation of Q-function for timestep t $Q_t^{\pi}(s_t, a_t)$ ("Critic model"), which is then used to partially update the initial $Q_{t+1}^{\pi_{t\arg}}(s_{t+1}, a_{t+1})$. This leads to instability, which is why TD3PG has two "Critic targets" instead of one (hence the term "twin")

and uses the smallest output of two to avoid the overestimation of action values and the subsequent policy breaking, which may happen with DDPG [84].

The first part of TD3PG, "Target", is shown in Figure 3.3, and the second part of TD3PG, "Target", is presented in Figure 3.4. At every iteration, TD3PG uses a batch of transition samples [s_t; s_{t+1}; a_t; r_t] extracted from the replay buffer. The first network, "Actor target", using state s_{t+1}, computes the actions $a_{t+1} = \pi_{targ}(s_{t+1})$. This value is inputted into two "Critic target" networks that compute two different action values (due to their random initialization at the beginning of the training) for the same timestep $t+1$: $Q_{t+1}^{\pi_{\text{targ1}}}(s_{t+1}, a_{t+1})$ and $Q_{t+1}^{\pi_{\text{targ2}}}(s_{t+1}, a_{t+1})$. Then, selecting the minimum between the two obtained action values and using the Bellman equation, the target action value for timestep t is calculated (3.3).

$$Q_t^{\text{targ}}(s_t, a_t) = r_t + \gamma \cdot \min(Q_{t+1}^{\pi_{\text{targ1}}}(s_{t+1}, a_{t+1}), Q_{t+1}^{\pi_{\text{targ2}}}(s_{t+1}, a_{t+1})) \quad (3.3)$$

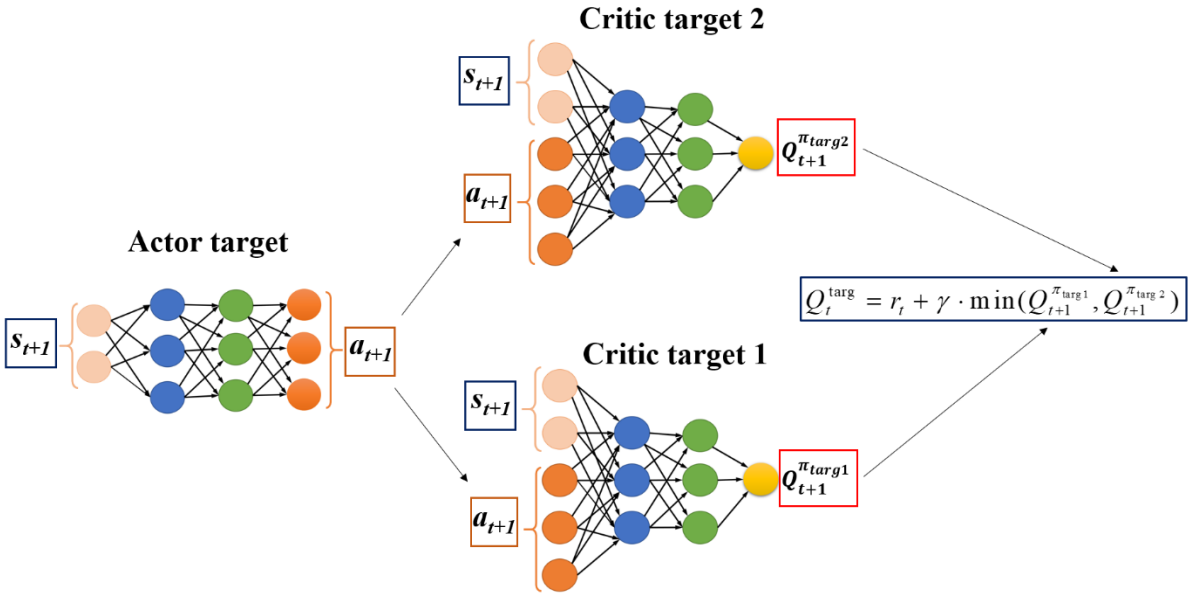


Figure 3.3. Principle scheme of the TD3PG "Target" part

Two other neural networks, denoted as "Critic models", compute the action-values for the timestep t: $Q_t^{\pi_1}(s_t, a_t)$ and $Q_t^{\pi_2}(s_t, a_t)$. Then the loss function L is computed as a mean-square error (MSE) between the obtained action values and the target action value (3.4). This loss value L is backpropagated from the NN output to their inputs (i.e., through all the hidden layers) to update the weights of the "Critic model" neural networks, which are denoted by the matrix $\boldsymbol{\theta}_{\text{model}}$.

$$L = MSE(Q_t^{\pi_1}(s_t, a_t), Q_t^{\text{targ}}(s_t, a_t)) + MSE(Q_t^{\pi_2}(s_t, a_t), Q_t^{\text{targ}}(s_t, a_t)) \quad (3.4)$$

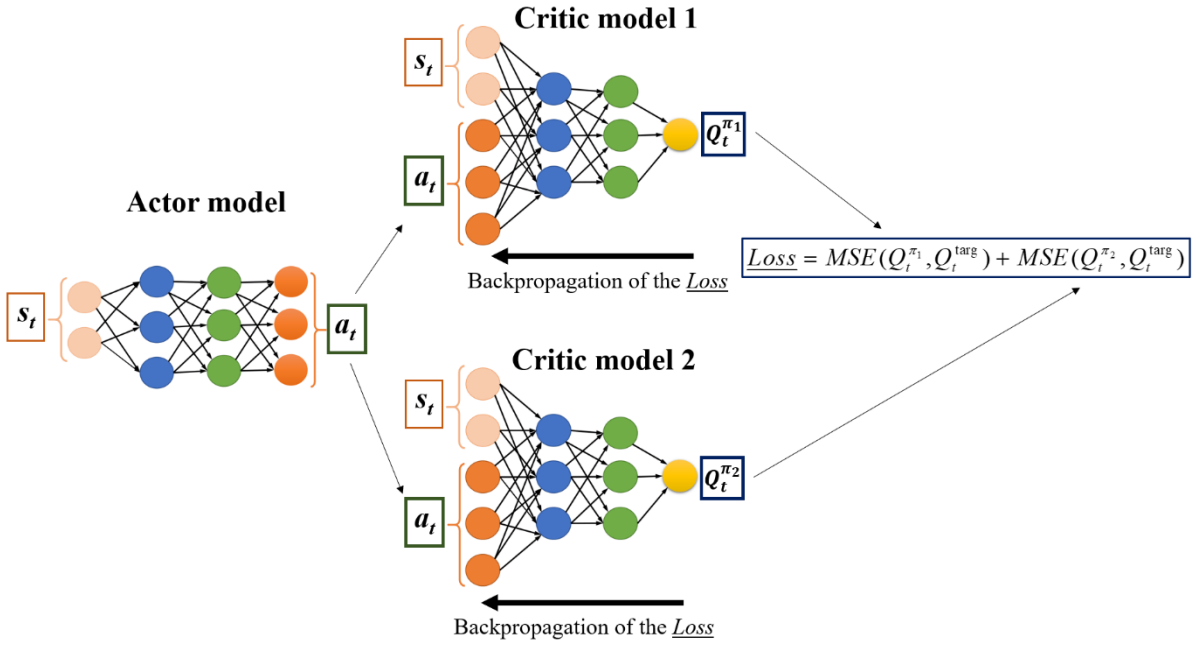


Figure 3.4. Principle scheme of TD3PG "Model" part

The "Critic target" models have "target" in their name because the algorithm tries by backpropagation (i.e. minimization of the loss L) to make the Q-function of the "Critic Model" more like this target. As the target also depends on this same value estimation network, it is not fixed but moves with each update of the network, which makes the convergence unstable. To solve this problem, TD3PG uses a "soft" update of the weights of the "Critic target" $\boldsymbol{\theta}_{\text{targ}}$ that come close to the weights of the "Critic models" $\boldsymbol{\theta}_{\text{model}}$, but with some delay. For this, the weights of the "Critic target" $\boldsymbol{\theta}_{\text{targ}}$ are updated through Polyak averaging (3.5) [84].

$$\boldsymbol{\theta}_{\text{targ}} = \boldsymbol{\theta}_{\text{targ}} \cdot \rho + (1 - \rho)\boldsymbol{\theta}_{\text{model}} \quad (3.5)$$

Where ρ is an interpolation factor, usually close to 1.

Finally, the "Actor model" is updated by performing a gradient ascent on the output of the first "Critic model". Then, once every specified number of iterations, the weights of the "Actor target", similarly to the weights of the "Critic target", are updated using Polyak averaging.

3.2.2.3 PPO algorithm

PPO is an on-policy model-free RL algorithm, which relies on a clipped objective function (i.e., saturation function) that prevents the policy to get too far after each update. In other words, it defines the maximal difference between the "old" probability distribution of each action $\pi_{old}(a_t \mid s_t)$ and a new one $\pi(a_t \mid s_t)$ along the training iteration. The main idea is to avoid instability in the policies generated in the course of the training. PPO maximizes the following objective function (3.6):

$$L = \min\left(\frac{\pi(a_t \mid s_t)}{\pi_{old}(a_t \mid s_t)} A^{\pi_{old}}(s_t, a_t), clip(\frac{\pi(a_t \mid s_t)}{\pi_{old}(a_t \mid s_t)}, 1-\varepsilon, 1+\varepsilon) A^{\pi_{old}}(s_t, a_t) \right) \quad (3.6)$$

Where $\pi_\theta(a_t \mid s_t)$ and $\pi_{\theta_{old}}(a_t \mid s_t)$ are the probabilities of actions considering the current and the previous ('old') policies respectively;

Clip denotes the clipping operation (limitation between two set values)

ε is the clipping parameter which defines how far the new policy can go from the old policy;

$A^{\pi_{old}}(s_t, a_t)$ - advantage function under the previous policy. It represents the advantage of selecting a certain action a_t for a certain state s_t. It can be defined as a difference between action value and state value (3.7). The use of the advantage function instead of the expected reward (as in TD3PG) reduces the variance of the estimation.

$$A^\pi(s_t, a_t) = Q_t^\pi(s_t, a_t) - V_t^\pi(s_t) \quad (3.7)$$

The objective function (3.6) is the main distinction between the algorithms PPO and other on-policy algorithms TRPO. Instead of a hard constraint on the probability ratio $\frac{\pi(a_t \mid s_t)}{\pi_{old}(a_t \mid s_t)}$ in the TRPO optimization problem (that significantly increases the computational complexity of the second-order optimization), PPO uses clipping to remove incentives for the new policy to go too far from the old policy. This can be optimized by a first-order optimizer (first-order derivatives method), and in combination with the advantage function, this allows almost monotonous performance improvements. The first-order derivatives method uses gradient information to construct the next training iteration which is simpler compared to second-order derivatives that compute the iteration based on the optimization trajectory [90].

3.3 Case study and operation principle

RL algorithms must be trained before being used in operation. However, it is not realistic to train any AI-based controller on an actual grid at the risk to send controls that could endanger the system's safety. Indeed, similarly to many metaheuristics, all training algorithms need to investigate a wide range of different actions before converging to near-optimal policies. The training is then conducted in an offline mode, on a simulated grid, a model of the system, before the controller is tested on an actual network (which will be simulated in this work). The case study consists of a medium/low voltage grid simulated using the Pandapower package in Python. Pandapower combines the "pandas" data analysis library and the "pypower" power flow solver to create a grid calculation program aimed at automated analysis and optimization of electric power systems [74]. This tool allows different scenarios to be simulated and thus testing the effectiveness of the developed solutions. The considered grid (shown in Figure 3.5) comprises eleven nodes ($n \in N$), six loads, three PV, and four flexibilities, ($f \in F$) in the form of storage systems that can regulate their active P_t^f and reactive Q_t^f power flows at each timestep t. Those are the actual degrees of freedom for voltage regulation purposes. The objective is to maintain as much as possible the voltage at each node V_t^n within specified limits [0.95 p.u., 1.05 p.u.].

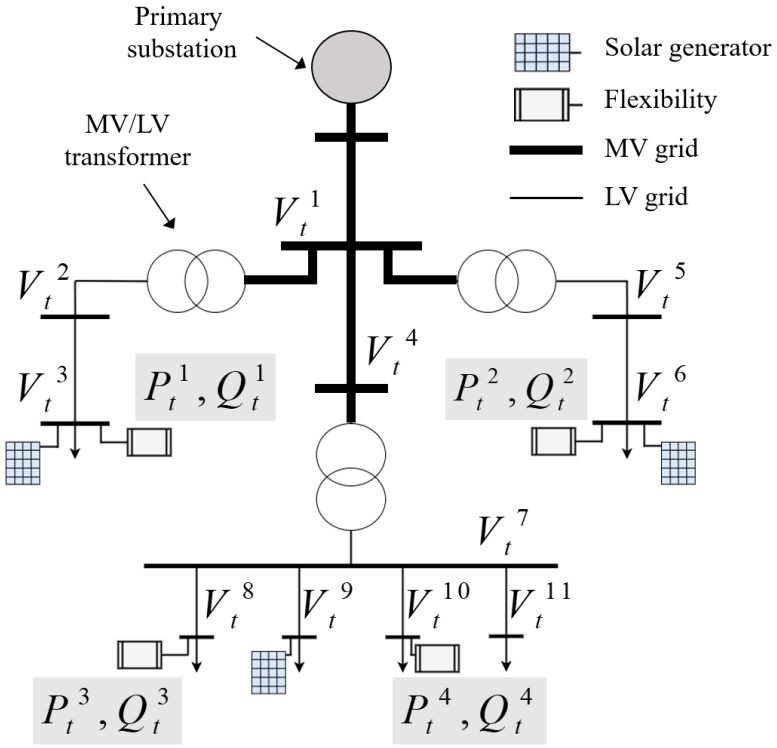


Figure 3.5. Considered MV/LV grid, including local storage and production in selected nodes

Traditional discrete tools are not suitable for frequent regulation, which can lead to the faster aging of the components [39]. Moreover, their response time may be insufficient compared to the power changes of RES. Thus, the flexibilities are here chosen as leverage for voltage regulation, considering the constant growth of their installed capacity in medium and

low voltage grids. Each of the four flexibilities considered in the use case consists of an energy storage system (i.e., battery + inverter). Batteries can inject and consume active power until they are completely depleted or fully charged, respectively. A 90% efficiency is assumed for both charging and discharging (i.e., 0.81 round trip efficiency). They have a saturation block that prevents charging or discharging beyond 0 and 100% of SOC, a capacity of 500 kWh, and a rated power of 200 kW. Inverters can regulate reactive power independently of the batteries in the range between -300 and 300 kVar and have no losses during operation. Thus, the maximum apparent power is 360 kVA. The capacity and the rated power are chosen as sufficient enough to maintain the voltage in the considered grid in the given limits and based on the preliminary tests.

Consumption profiles from the "Smart meters in London" project [69] are used to populate the simulated system while assigning load to every consumption node on a 30-minutes basis. Following a pre-processing step of the raw data, each profile aggregates the consumption of a distribution grid feeder with around 50 households. To model the solar generation, representative PV profiles from NREL [71] with 5-minute timesteps are used and resampled at a 30-minutes – which is the time resolution of the control here. As a result, six different load profiles (Figure 3.6 (a)) and three PV profiles (Figure 3.6 (b)) are ultimately prepared and scaled according to the modeled grid leading to a baseline case (with no flexibility control) that displays both undervoltages and overvoltages throughout the year.

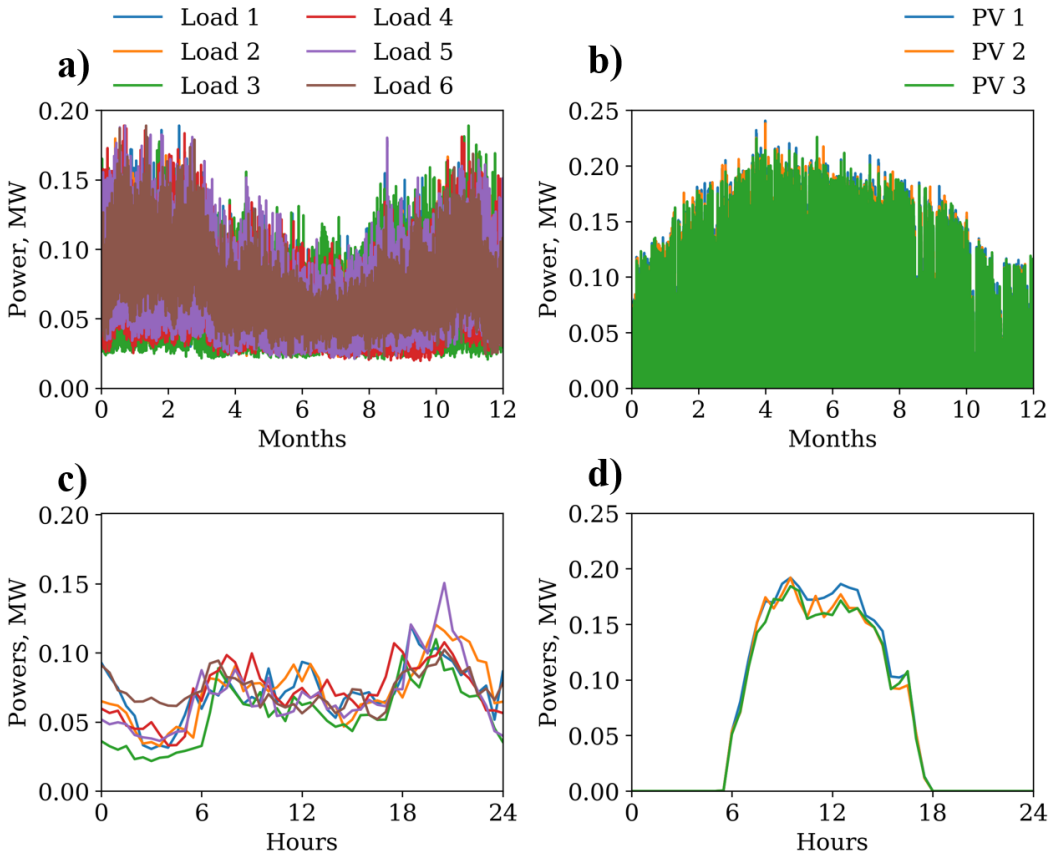


Figure 3.6. Load consumption (a) and PV generation (b) profiles of the modeled grid for a year and one day (c and d)

To train, the model of the test system (Figure 3.5) is connected to the RL algorithm for actions and observation exchanges, as presented in Figure 3.7. State s_t is represented by the vector expressed in (3.8):

$$s_t = \left[\mathbf{V_t^n}, \mathbf{P_t^f}, \mathbf{Q_t^f}, \mathbf{SOC_t^f}, t \right] \quad (3.8)$$

$$a_t = \left[\mathbf{P_{t+1}^f}, \mathbf{Q_{t+1}^f} \right] \quad (3.9)$$

where t denotes the current timestep, $\mathbf{V_t^n}$ the vector of voltages for all n nodes of the grid, $\mathbf{P_t^f}$ the active power of the batteries, $\mathbf{Q_t^f}$ the reactive power of inverters, and $\mathbf{SOC_t^f}$ the state of charge of the batteries (SOC) - at time t. Note that, the algorithm does not require access to the node consumption/production data but only the voltage measurements or their estimation. From these observations, the RL algorithm computes the actions a_t as the references setpoints (active/reactive) for the flexibilities at the next timestep $t+1$ (3.9), based on the policy that is adjusted along the training process. Thus, the RL algorithm implicitly accounts for built-in load and local generation forecast for the next timestep based on the voltage measurements at the current timestep. For this reason, the values of $\mathbf{P_t^f}$ and $\mathbf{Q_t^f}$ are known for state s_t, because they represent the actions from previous timestep $t-1$.

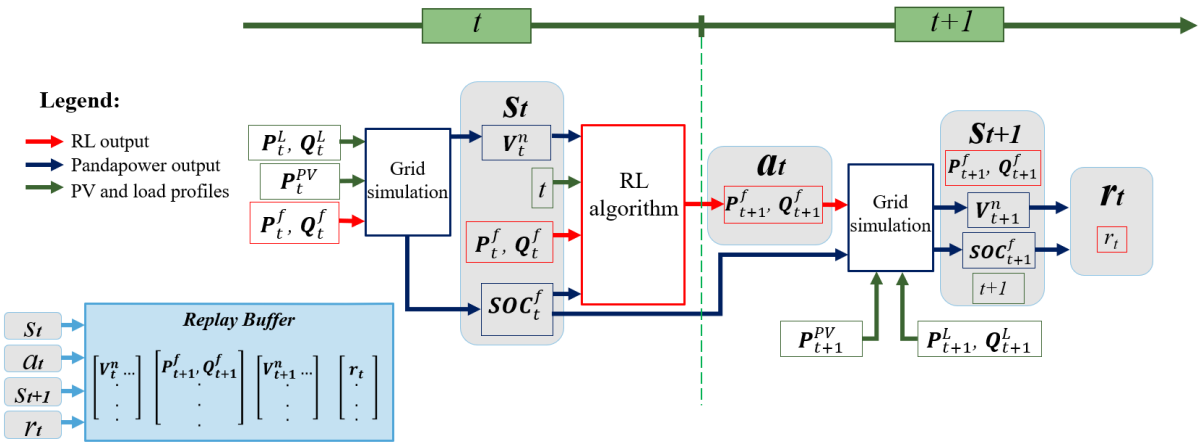


Figure 3.7. Proposed RL algorithms training and interaction with the study case

Based on the flexibilities setpoints and PV/load values at timestep $t+1$, the simulation updates the state variable values once the controls are applied to the environment, thus providing the observation s_{t+1} to the RL algorithm at the next step. From this observation, the algorithm also calculates the corresponding reward r_t whose value allows the training process to tune the policy. As expressed in (3.10), the reward penalizes any voltage deviations outside the boundaries of the normal operations, set between 0.95 p.u. and 1.05 p.u. Note that depending on the system layout, the load/generation profiles, and the installed storage capacities, it may not be possible to avoid the totality of the deviations. Similar to an optimization/model-based controller, the objective here to is minimize as much as possible the

occurrences of those deviations. The second term in the reward function tends to preserve storage availability. Especially, an exponential contribution to the reward penalizes SOC values as a function of their deviation from 50% (0.5 in the exponent) with a coefficient ω. Finally, coefficients α and β allow arbitrage between the two objectives - voltage and SOC deviations (sensitivity analysis of reward function is presented further in section 3.4.2.2).

$$r_t = -\alpha \sum_{n=1}^{N} \frac{(max(0,(V_{t+1}^n - 1.05))^2 + max(0,(0.95 - V_{t+1}^n))^2)}{N} - \frac{1}{\beta} \sum_{f=1}^{F} \frac{exp(\omega \cdot |SOC_{t+1}^f - 0.5|)}{F} \quad \textbf{(3.10)}$$

Note that different functions are considered to penalize voltage violations (quadratic) and SOC oscillations (exponential). This is due to different requirements for these objectives. The penalty for small SOC deviations around 0.5 should be significantly less penalized compared to small voltage violations, enforcing the algorithm to charge or discharge the battery to maintain voltage. However, for SOC values below 0.2 or above 0.8, the penalty for SOC deviations should increase rapidly (because based on the given capacity and rated power of the batteries, they can change their SOC maximum by 20% in one 30-minute timestep) to avoid complete charge or discharge. In this way, the batteries with the values of SOC beyond these limits will be used only for the most severe voltage violations. An exponential function (with correctly chosen coefficients ω and β) allows the policy to tend toward the desired behavior of flexibilities. An example of a quadratic function (for voltage violations ΔV) and several exponential functions (for SOC deviations) is presented in Figure 3.8.

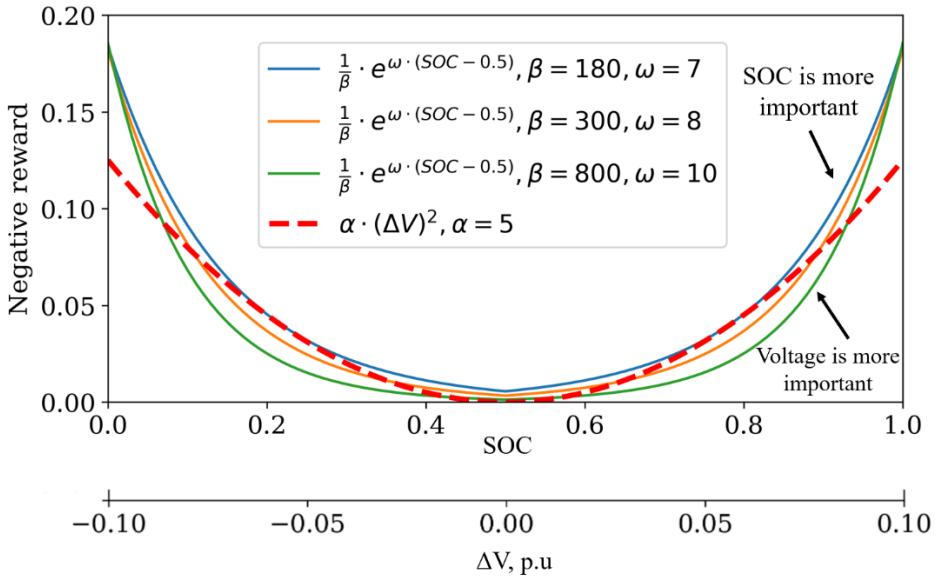


Figure 3.8. Example of quadratic and exponential functions comparison

It can be seen, that the exponential functions increase more slowly compared to the quadratic function at the beginning, and significantly faster at extremes. Moreover, by changing coefficients β and ω, it is possible to obtain similar values of the exponential functions at extremes, but different ones at intermediates points, thereby adjusting the behavior of the controller – values below the quadratic function mean that the voltage regulation is currently

more important, and values above quadratic function mean that the most important term of the reward now is SOC penalization.

As explained in the previous section, the objective of any RL algorithm is to maximize the sum of rewards over a time horizon (or minimize the negative sum of the rewards in this case). Empirically chosen coefficients (α, β, and ω) allow training of the RL algorithm to maintain the voltage within given limits while avoiding complete depletion or overcharging of batteries. Each transition [s_t, s_{t+1}, a_t, r_t] is later used in the training (weights/biases updates), allowing the algorithm to gradually learn the behavior of the simulated grid and adjust the control policy along the training process (i.e., iterations/epochs). Note that the original training set consists of consumption and production data over a full year. However, during the training process, the algorithm can run multiple "theoretical" simulations over the same year while choosing different actions for the same states to converge toward a near-optimal policy.

A test phase is run following the training while assessing the controller performances with consumption and local generation profiles for another year. In practice, the controller trained offline is deployed on the system (based on grid simulation here). Thus, the testing set shall be independent of the training data. The proposed in this chapter operation principle presents two main advantages:

1. The RL algorithms do not require an external forecast and implicitly embed a prediction of the bus power to compute the controls;

2. The RL algorithms do not need potentially sensitive private data (such as bus load and/or generation) are needed, as the controls are only computed based on the voltage values;

3.4 Training results and sensitivity analysis to hyperparameters

3.4.1 Performance metric

To assess the performance of the implemented RL-based controllers, a voltage performance index (VPI) is defined in (3.11). It enumerates the occurrences where the node voltage is outside of specified limits - an average over all the buses in the grid and over the simulated period (T).

$$\mathrm{VPI} = \frac{\sum_{n=1}^{N}\sum_{t=1}^{T}\delta_t^n}{|N|\cdot|T|}\cdot 100\% \begin{cases} \delta_t^n = 0 & if\ 0.95 \le V_t^n \le 1.05 \\ \delta_t^n = 1 & otherwise \end{cases} \qquad (3.11)$$

As previously mentioned, one-year generation/load profiles are considered for the offline training and another year's profiles for the testing. The VPI over the test year (i.e. T=17520) without any control is significant (19.5%) with both under and overvoltages occurring along the simulation period. Only four representative weeks of the year (one per

season) are used to calculate the yearly VPI (T=1344), because of the long optimization time (around 5 hours for 16 cores, 96Gb machine) of the benchmark algorithm, which is later used for comparison with RL results (it will be presented in Chapter 4). This allowed us to reduce the computational time (notably considering the need for multiple runs) while providing consistent results - VPI is 18.7% over the four representative weeks compared to 19.5% for the whole year without any flexibility control.

3.4.2 Training settings and sensitivity analysis

3.4.2.1 Training settings and test validation

One identified drawback of Machine Learning methods and RL algorithms, in particular, is that their training is very sensitive to hyperparameters. The speed of convergence with the same initial set of random transitions [s_t; s_{t+1}; a_t; r_t] can differ significantly depending on the initialization and algorithm parameters settings. Those random transitions are generated by random actions in a preliminary step of the training. Similarly, the same initialization and hyperparameters cannot guarantee to converge toward similar results (in terms of the controller) due to the stochastic nature of the training algorithms. Thus, a combination of correctly selected hyperparameter values (e.g., by sensitivity analysis) and fixed initialization is necessary to obtain close to optimum control strategies. A sensitivity analysis over 1500 runs was performed to choose the best set of hyperparameters for both TD3PG and PPO. Values were selected to provide the highest performance after the training phase - i.e., the lowest VPI of the trained RL algorithm over the four weeks of the test dataset.

A set of preliminary tests was performed to validate the representativeness of the four chosen weeks. Four controllers were then taken from the 1500 runs performed (two with a low VPI and two with a high VPI) and executed on the whole test year. Their VPI was calculated and then compared with the corresponding VPI over the four test weeks. The results, displayed in Table 3.1, show very close performances of the model in both periods, thus validating the conservativeness of the results over the four representative weeks – i.e., if a controller is less performant over the whole year, it is also less performant over the representative period. Thus, a four-week test can be used to estimate the efficiency of RL-based control for the whole year as it will be done in the remaining of this thesis

Table 3.1

COMPARISON OF VPI FOR 4 WEEKS AND 1 YEAR

Model N°	1	2	3	4
4 weeks	2.9%	3.1%	19,1%	18.6%
1 year	3.6%	3.5%	19.0%	19.1%

For the sake of simplicity, the size of each training epoch (time horizon) was set to 48 steps (i.e., one day at 30 min resolution), where each step corresponds to one interaction between the controller and its environment for the next 30 minutes. After preliminary tests, the total number of epochs for the training was set for the two considered algorithms:

- For TD3PG: 4000 epochs with update (adjustment of weights) every 8 steps for TD3PG;
- For PPO: 30000 with 20 updates at the end of each epoch for PPO.

The Adam optimizer [93] was used to implement the stochastic gradient descent to train the networks (i.e., to update the weights and biases).

3.4.2.2 Sensitivity analysis of the performance of the main hyperparameters

The sensitivity analysis consists in finding the best set of hyperparameters to get a control that reduces the VPI values as much as possible. To do so, a greedy method is implemented where the parameters are adjusted one at a time toward the direction of a lower VPI until there is no improvement (dynamic tuning). In other words, all hyperparameters are initially set to their typical values. From this initial set of hyperparameters values, every parameter is then investigated individually, and a grid search can be performed. By varying the studied parameters, we can find their best values and fix them for further analysis of other parameters. The results of the sensitivity analysis of the common parameters for both RL algorithms are presented further in this subsection and illustrated in the TD3PG example. For each value of the hyperparameter, five different pieces of training were executed to partially mitigate the stochasticity of RL training. The number of trainings was chosen as a trade-off between the calculation power of the server used, the required training time, and the number of different hyperparameters' values to test. Further in this thesis, the best VPI value among five results of trained RL algorithms is presented (while testing the corresponding controller of the four representative weeks).

- Number of nodes and layers in the neural networks

At first, the impact of the model's structures and the number of neurons (for both, "Actor" and "Critic" networks) were assessed. To ensure the same initial conditions for a fair comparison, a common random seed (i.e., initial configuration) was chosen for the investigated neural structures. That allowed having the same initialization of all network weights and the same direction of the gradient at the beginning of the training for a fair comparison of the results. Figure 3.9 shows the VPI obtained with different network architectures over the 4000 training epochs. These VPI curves are obtained by stopping training every 50 epochs, running the currently trained algorithm over the four test weeks, and getting the corresponding VPI, before resuming the training. This allows getting a visualization of the training dynamic of the algorithm, which does not depend on which days the algorithm is trained at each given moment.

In the beginning, the training returns bad VPI values, even worse than the base case without any flexibility to control (18.7%) - reaching 100% of VPI for all structures before 800 epochs. This can be interpreted as the necessary number of iterations needed by the RL algorithm for a good understanding of its environment and mission (exploration phase) before it can adjust its policy from random actions toward a meaningful control (exploitation phase). Among the configurations tested, the structure with 20 neurons and one hidden layer displays the worst results with a VPI still above 30% after 4000 epochs. The neural network with one hidden layer of 32 neurons achieves a VPI of 4.1% at the end of the training. It is better than the structure with two hidden layers of 50 neurons each, but still worse than the 32x32 structure,

which shows VPI of 4.5% and 2.9% at 3700 and 3300 epochs, respectively. It can be concluded that one hidden layer is not enough to capture all dependencies, and a structure of 50x50 neurons needs too many samples to converge towards a better solution. Thus, the higher complexity of the network structure does not systematically return better performances, and a 32x32 structure is chosen, which represents the trade-off and gives the best results. As can be seen in Figure 3.9, the best VPI value along the training does not necessarily correspond to the last epoch, which is further discussed in section 3.5.1.

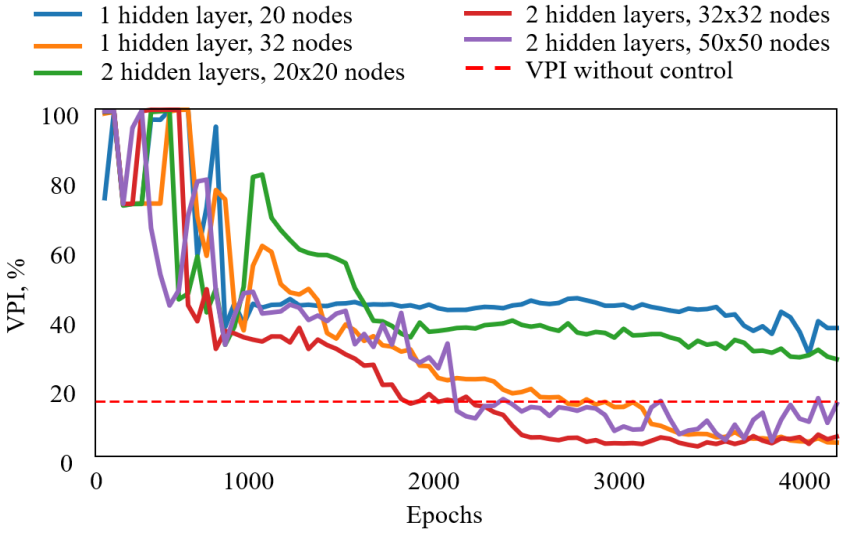


Figure 3.9. Learning curves of TD3PG for different numbers of nodes and layers in the neural networks

- Reward coefficients α, β and ω

Next, the dependency of the controllers' performances on the coefficients α, β, and ω of the reward function expressed in (3.10) is investigated. The training is executed over 4000 epochs as the algorithm did not show any improvement with longer training in the previous tests. Note that, for each set of α, β and ω values, the best VPI obtained along the training was kept, which did not necessarily correspond to the value obtained at the end of training.

As was shown in Figure 3.8, by changing β and ω it is possible to get the same reward for SOC oscillations at extremes (SOC=0 and SOC = 1), but a different reward for intermediate SOC values. Thereby, different couples of β and ω were chosen in such a way that they have the same reward at extreme points. The best VPI results for these (β, ω) couples and different values of α coefficient are presented in Figure 3.10. The coefficient α adjusts the importance of the voltage regulation objective. As expected, when α is too high, the algorithm neglects the SOC value and tends to overuse the battery. That can lead to situations where batteries are fully charged or discharged and cannot mitigate severe voltage oscillations. Similarly, when α is too low, the algorithm focuses mainly on keeping the SOC close to 0.5 while ignoring the voltage deviations. In that case, flexibilities are under-used.

Higher β corresponds to higher ω, which leads to a significant reward for values close to the extremes (0 and 1) while leaving a relatively low reward for values close to 0.5. At the

same time, too steep characteristics should be avoided, as it complicates the training of the algorithm and will not motivate the agent to return to the target SOC value of 0.5 without violating the voltage limits.

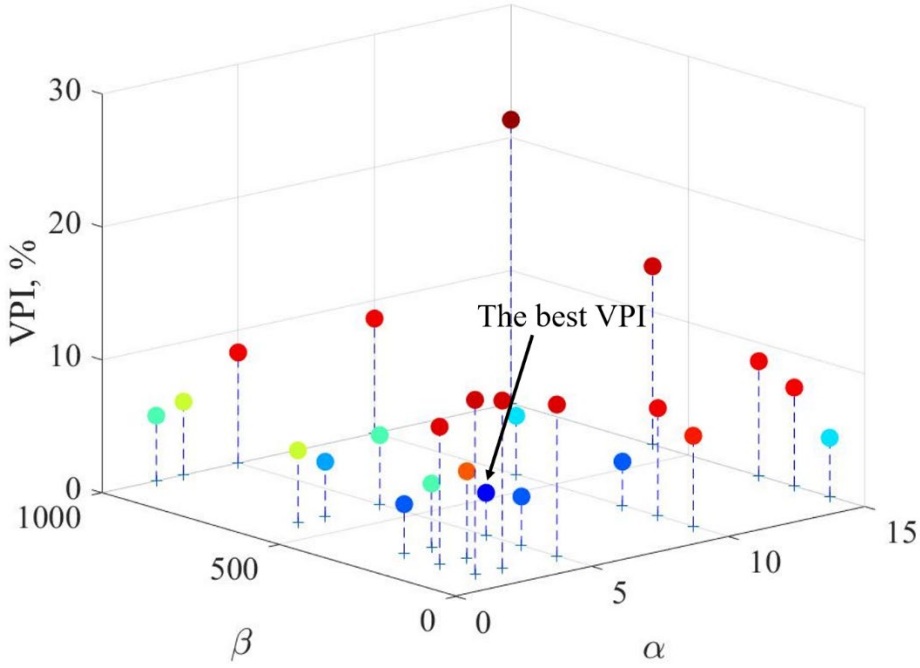


Figure 3.10. Best VPI of TD3PG training for different coefficients α and (β, ω)

Figure 3.10 shows the increase in performance as β decreases to 300, amplifying the importance of small SOC deviations. But as β decreases further, performances drop because the algorithm starts to neglect the voltage violations even for relatively small SOC oscillations. Thus, β=300 and corresponding ω=8 result in the best slope of the reward function, which together with α=5 leads to the lowest VPI. As a result, these values are chosen for the simulations discussed in Chapters 3 and 4.

- Activations functions

The impact of the activation functions of the neurons in the hidden layer that was discussed in the first chapter (Figure 1.5) is presented in Table 3.2.

Table 3.2

COMPARISON OF BEST VPI FOR DIFFERENT ACTIVATION FUNCTIONS

Sigmoid	Tanh	Softmax	ReLU	ELU	LogSigmoid
7.04%	2.94%	3.67%	2.95%	3.47%	4.12%

Tanh and ReLU functions show the best VPI, whereas neural networks with the Sigmoid activation function have the lowest performance once the trained controller is tested on the four representative weeks. As a result, Tanh is chosen for the "Actor" network and the ReLU function is chosen for the "Critic" network.

3.4.2.3 Sensitivity analysis to TD3PG specific hyperparameters

- Coefficient η

This subsection gives examples of hyperparameters' impact specific to TD3PG. Figure 3.11 illustrates the results of a sensitivity analysis on the convergence to the learning rate η for both "Actor" and "Critic" neural networks.

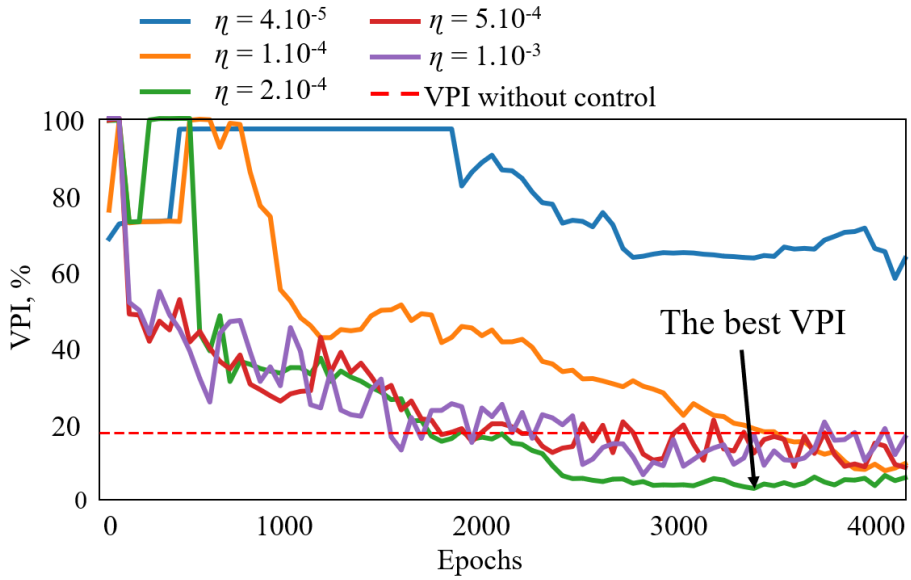


Figure 3.11. Learning curves of TD3PG algorithm for different learning rates

Large learning rates (e.g., $5e^{-4}$ and $1e^{-3}$) allow faster convergence at the beginning of the training process but do not lead to the best results. Indeed, large learning rates take too large steps at each update and may miss optimum solutions. After 500 epochs, smaller learning rates (e.g., $2e^{-4}$) lead to a monotonic improvement, and after 3200 epochs reach a better VPI (2.8%). However, with too small learning rate values, the networks can get stuck into less relevant local optimum. Thus, a medium value for the learning rate ($2e^{-4}$ in our case) presents a good trade-off in terms of the VPI metric on the four test weeks

- Batch size and replay buffer size

The dependency of the training results on the batch size (i.e., the number of samples used for each update of the neural networks) and on the replay buffer size is presented in Figure 3.12.

The size of the replay buffer directly affects the training process. Small replay buffers are the most suitable for a non-stationary environment. Stationary environments do not change without the action of the Agent (e.g., chess board), non-stationary environments change over time (e.g., in a traffic lights control problem, traffic patterns vary over time). For small buffers, only the recent interactions' results are considered at each update. This also helps converge faster to the current local optimum. On the contrary, bigger replay buffers allow better generalization and smooth changes in the data distribution as random samples over a long period are considered in the computation of the training metrics (e.g., loss function). An

incorrectly sized replay buffer will slow down the learning of the correct value function or on the contrary, make the network easily "unlearn" good behavior. Thus, similarly to the learning rate, a trade-off has then to be found which justifies the analysis discussed here [91].

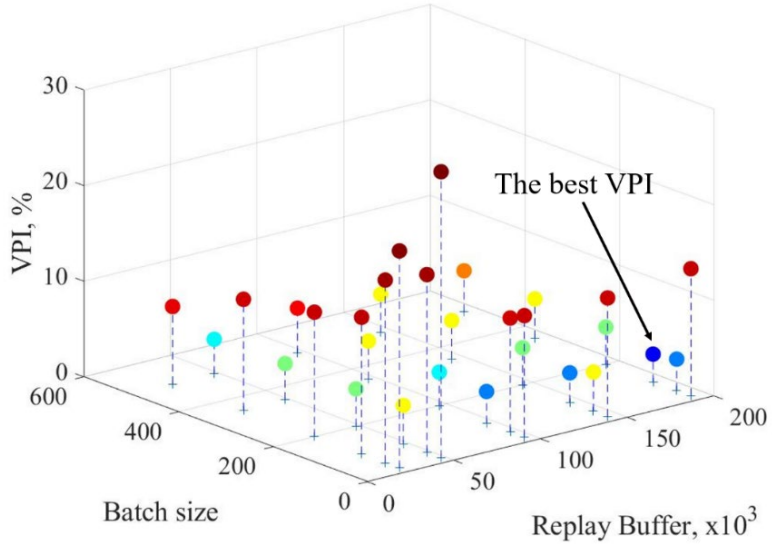


Figure 3.12. VPI of TD3PG for different batch and replay buffer sizes

According to [92], small batch sizes tend to converge to flat local optimum that vary only slightly around this optimum, whereas large batch sizes converge to sharp local optimum. Flat optimums tend to generalize better since they are more robust to any changes between the training and test sets. However, small batch size training finds minimizers further away from the initial weights, compared to large batch size training [92]. As can be seen in Figure 3.12, for a batch size of 500, the best size of the replay buffer is 48000. However, as the batch size decreases, the size of the best replay buffer tends to grow. Thereby, a batch size of 100 with a replay buffer capable of saving all interactions during 4000 epochs showed the lowest VPI and was selected for the rest of the work as default values for those hyperparameters.

- Number of random actions at the beginning and noise of the actor model

Finally, the dependency of the performance of TD3PG to the number of random actions at the beginning of the training N_{rand} and the noise of the "Actor model" network Δ_{act} is presented in Figure 3.13 (a) and Figure 3.13 (b) respectively. The noise is a random component added to each action element. Both parameters, N_{rand} and Δ_{act}, are used to balance the exploration and exploitation of the algorithm. TD3PG relies on a deterministic policy. Thus, at the beginning of the training, it performs a search space exploration by executing a given number of random actions.

The more random actions TD3PG tests at the beginning, the more time it generally takes at the beginning to adjust the policy. This is because the agent cannot check the results of the actions proposed by its policy to assess their effectiveness. Instead, to estimate how good can be the proposed actions, the algorithm uses the reward received for performing other random actions. For this reason, the algorithm with 2000 epochs of random actions presents the worst

VPI at the end of the random phase compared to other algorithms at the same moment (as seen in Figure 3.13 (a)). However, a bigger number of random actions allows us to get a better generalization of the Critic network. Thus, over the next 2000 epochs, the algorithm with 2000 initial epochs of random actions achieves the same VPI as the algorithms with random actions over 1000 and 500 epochs (VPI = 2.8%). Algorithms with a small exploration phase (100 and 250 epochs), on the contrary, reach the best VPI among all algorithms after 500 epochs, but then do not converge as well as the others and reach only a VPI of 6.51% and 5.79% respectively. These results are due to the lack of diversified actions which leads to a poorer generalization of the neural networks. Thus, random actions over 500 epochs showed the best results in terms of trade-offs between suboptimal solutions and excessive exploration.

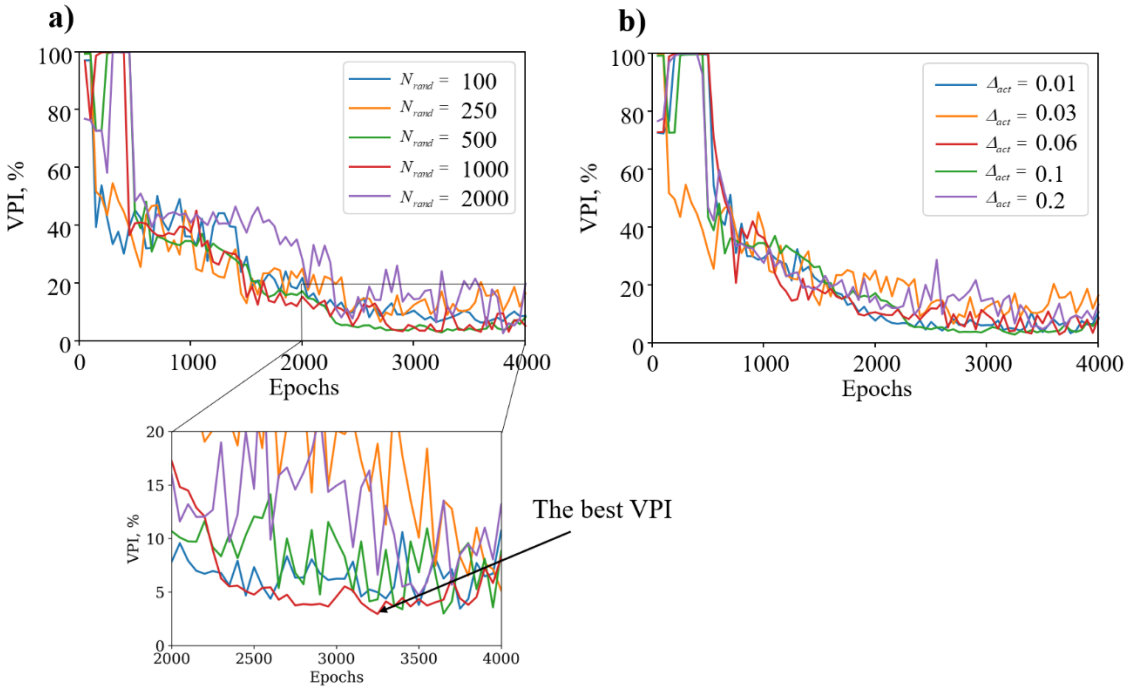


Figure 3.13. VPI of TD3PG – a) for different numbers of epochs with random actions at the beginning of training N_{rand} b) for different Actor noises Δ_{act}

"Actor model" noise is the second parameter that affects the capacity of the training to arbitrage between exploration and exploitation. In particular, it impacts the second phase of the training once the exploration is done with random actions. The value of the noise is taken from a normal distribution with a given standard deviation and is added to each action element. Unlike PPO, the standard deviation is constant, and the noise is limited in such a way that the action with noise does not exceed the limits set for the actions (i.e., saturation block on the action). The results presented in Figure 3.13 (b) do not show a high dependency in terms of VPI. This is probably due to a large enough exploration phase at the beginning and a large replay buffer that contains all the transitions which allow a good generalization during the entire training, no matter the noise value. However, despite the small impact observed, a standard deviation of 0.1 gives a slightly better VPI, thus this value is used further.

3.4.2.4 Sensitivity analysis to PPO specific hyperparameters

- Optimizer learning rates of value function and policy

The results of the sensitivity analysis of the PPO algorithm control to the optimizer learning rates of value function and policy are given in Figure 3.14 (a) and Figure 3.14 (b) respectively. Unlike TD3PG, PPO optimizes for the "Critic" network the state-value function $V_t^{\pi}(s)$ (3.2) instead of the action-value function $Q(s_t,a_t)$ (3.1). PPO, contrary to TD3PG, uses a stochastic policy, where each action is taken from a generated probability distribution. The standard deviation will then vary from a large value at the start of the training, to a smaller value as the algorithm converges to the optimal solution. PPO thus automatically encourages exploration at the beginning of the training and prioritizes exploitation at the end. Also, PPO converges to a policy π, which maximizes the advantage function $A^{\theta}(s_t,a_t)$ instead of the expected reward, which allows reducing the variance of the estimation

Figure 3.14 (a) displays training results obtained when varying the learning rate of the value function and no significant impact can be observed. That is explained mostly by the use of an advantage function (difference between action value and state value functions) during the optimization. The optimizer uses an incremental reward, which can be expressed as the difference between the action value and the state value (3.7). Thus, the objective of the algorithm is to determine whether the chosen action for the last timestep provided a higher reward compared to the value (expectation) of the state-value function for the same timestep. If yes, PPO increases the probability of such actions in cases where similar states are observed in the future. This way, PPO does not need an action-value function, but only a state-value function.

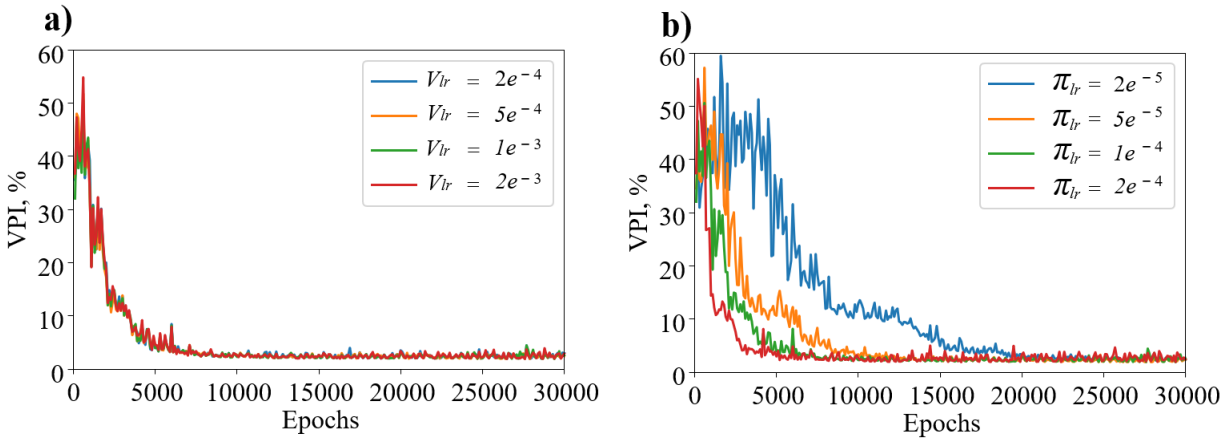


Figure 3.14. VPI of PPO – a) for different learning rates of policy optimizer b) for different learning rates of value function optimizer

The policy learning rate notably affects the convergence process with a speed of convergence that increases as the learning rate increases. A learning rate of $2e^{-4}$ allows for reaching a VPI of 5% after 3500 epochs, whereas training with a learning rate of only $1e^{-4}$ needs 4800 epochs, and a learning rate of $2e^{-5}$ needs almost 15000 epochs for the same accuracy. The

final performances are very close no matter the learning rate value. However, a value of $1.e^{-4}$ returns slightly better results (VPI of 1.85%) compared to the tests with other learning rates. The learning rate of $1e^{-4}$ is thus selected for the rest of the work. Such close results for different learning rates are due to the trust region optimization method of PPO compared to line search methods, such as gradient descent [95]. This allows for avoiding significant drops in accuracy during the training even with relatively big learning rates.

- Clipping parameter ε and the KL-divergence coefficient

The impact of the clipping parameter ε and the KL-divergence coefficient on the VPI are presented in Figure 3.15 (a) and Figure 3.15 (b) respectively.

Both of these parameters limit how much the new policy (after update) can be different from the old one. Unlike the clipping parameter ε, which does not have a noticeable effect on the convergence rate, the bigger KL-divergence parameter allows a substantially speed up of the training – a VPI of 5% is achieved after 3600 epochs for KL=0.03 and after 8200 epochs for KL=0.003. However, too large parameter values increase the variance of policy, thus the best performance corresponds here to a medium-sized KL=0.02 and ε=0.01.

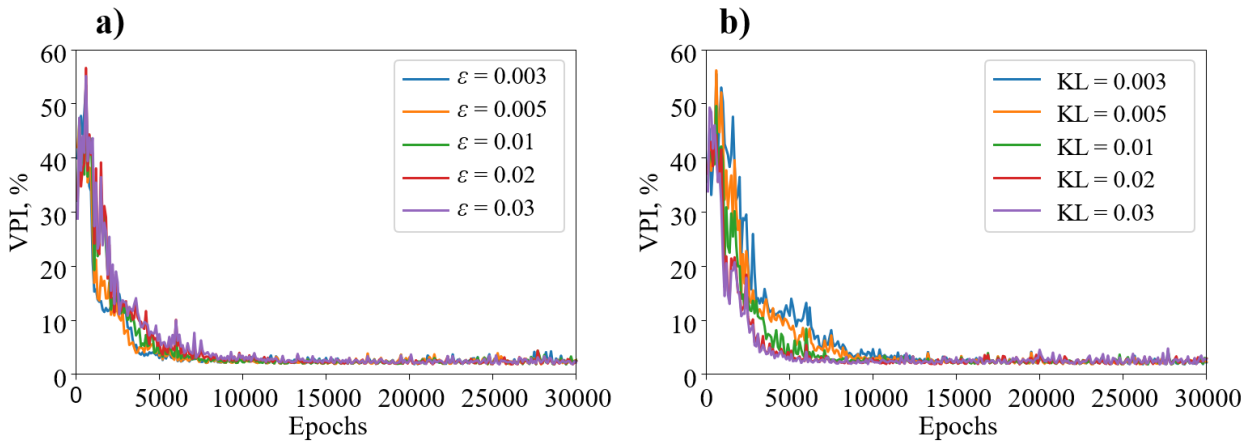


Figure 3.15. VPI of PPO – a) for different clipping parameters ε b) for different KL-divergence values

3.4.2.5 Optimal parameters values

Based on the main results of the sensitivity analysis presented above, the values of the most important parameters were selected. A summary is presented in Table 3.3. Those parameters were used to get all the results furtherly presented in this chapter.

Table 3.3

PARAMETERS OF THE ALGORITHMS			
Parameter	**Value**	**Parameter**	**Value**
Hidden layers	32x32	α	5
ω	8	β	300
Activation function ("Actor" network)	Tanh	Activation function ("Critic" network)	ReLU
SPECIFIC PARAMETERS FOR THE PPO ALGORITHM			
Nb. of gradient descent steps for policy/value function	20/20	Nb. of steps between networks update	48
Clipping parameter ε	0.1	KL-divergence	0.02
Learning rate for policy optimizer	$1e^{-4}$	Learning rate for value function	$2e^{-4}$
SPECIFIC PARAMETERS FOR THE TD3PG ALGORITHM			
Replay size	192000	Policy learning rate	$2e^{-4}$
Batch size	100	Q-function learning rate	$2e^{-4}$
Noise of the "Actor model" Δ_{act}	0.1	Noise of the "Actor target"	0.2
Nb. of random actions for initial training N_{rand}	24000	Nb. of steps between networks update	8

3.5 Analysis of trained algorithms

The best values of the hyperparameters following the sensitivity analysis, summarized in Table 3.3, are used to train RL algorithms. Their learned policies are examined and the efficiency of corresponding RL-based controllers for voltage regulation is investigated.

3.5.1 Controller performances in the training period

As an off-policy RL algorithm, TD3PG allows multiple reuses of transition samples [s_t; s_{t+1}; a_t; r_t]. For this reason, TD3PG presents a faster convergence than PPO, reaching a VPI of 2.8% after 3600 epochs, while PPO needs 9000 epochs to achieve the same accuracy (Figure 3.16).

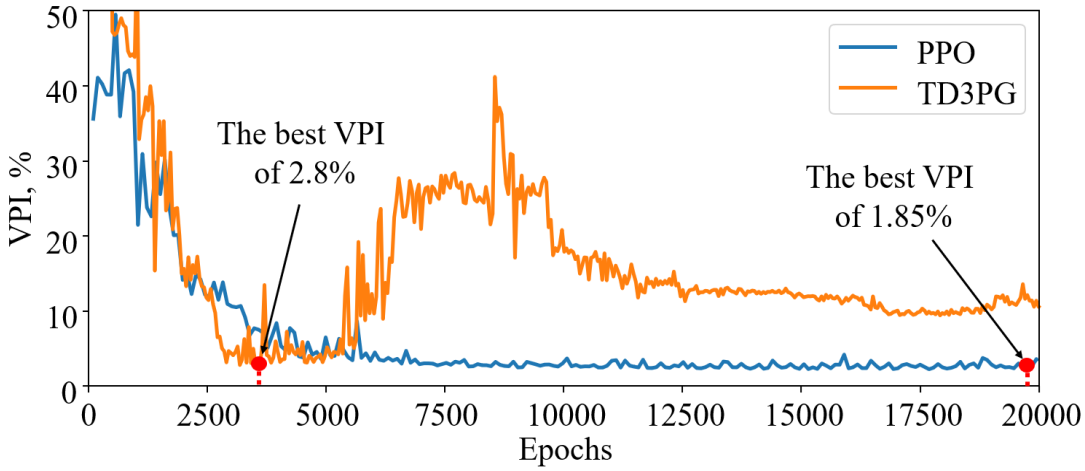


Figure 3.16. Learning curves of TD3PG and PPO

However, TD3PG may encounter problems with further convergence due to the "Deadly Triad" phenomenon, which is common for off-policy algorithms with function approximations in the form of neural networks [96]. This phenomenon occurs if s_t and s_{t+1} are sufficiently similar, which is the case for the voltage control over two successive 30 minutes periods. As a result, the value function suffers from instabilities and/or divergence. PPO, on the contrary, thanks to its constraints on the newly learned policy, avoids divergence and allows an almost monotonic slow improvement [85], reaching a VPI of 1.85% after around 20000 epochs (i.e., it needs 5.6 times more training epochs than TD3PG).

3.5.2 Learned policies

While the action-value function helps to roughly estimate how good the action will be (the results of an action-value function study are presented in Appendix 4), the policy function is the most important part of the RL algorithm. It determines which control will be computed based on state measurements. To explain the difference between voltage control results of the two RL algorithms (VPI of TD3PG is 2.8% and VPI of PPO is 1.85%), the actions (i.e., active/reactive setpoints) of flexibilities 1 (P_t^1, Q_t^1) and 2 (P_t^2, Q_t^2) under different voltage values at nodes 2, 3, 5 and 6 (V_t^2, V_t^3, V_t^5 and V_t^6) are presented in Figure 3.17 for the TD3PG control and in Figure 3.18 for the PPO control. Generation (i.e., discharge) is denoted by positive values, and consumption (i.e., charge) by negative values. The figure is obtained while inputting manually predefined states to the controller and analyzing its outputs in terms of flexibility powers. The objective is to check the physical validity of the trained controllers.

Due to limitations of the 3D representation, only 2 variables vary when calculating the action-value function. To implement this, several unrealistic approximations are done. The surfaces are plotted for a single timestep t (e.g. fixed at noon), the voltages at all nodes except 2, 3, 5, and 6 are fixed to 1 p.u., the SOC of the batteries $\mathbf{SOC_t^f}$ are equal to 0.5, and the powers of all batteries and inverters $\mathbf{P_t^f}, \mathbf{Q_t^f}$ are equal to zero. The first variable is the voltages at nodes 2 and 3 (which are approximated to be equal), and the second variable is the voltages at nodes

5 and 6 (which are also approximated to be equal). It is logical to expect that the algorithm will try to increase active and reactive power of flexibilities for undervoltage and decrease them in case of overvoltage. In the considered grid, flexibility 1 is located at node 3, so it has the greatest effect on the voltage at nodes 2 and 3 (connected through line 1), and flexibility 2 is located at node 6, so it has the greatest effect on nodes 5 and 6 (connected through line 2).

However, the trained TD3PG displays a counterintuitive policy for the considered study case (which is expected due to its worse performance with a VPI of 2.8% compared to 1.85% for PPO). First, the active powers of both batteries (1 and 2) and the reactive power of inverter 1 are approximately equally dependent on the voltages of the nodes in both lines. And the reactive power of inverter 2 (Figure 3.17 (d)) depends to a greater extent on the voltages of nodes 2 and 3 of the opposite line. Second, the reactive power of inverter 1 (Figure 3.17 (c)) behaves oppositely to the battery in the same node, decreasing as the voltage drops, and producing a negative value no matter the voltage levels measured.

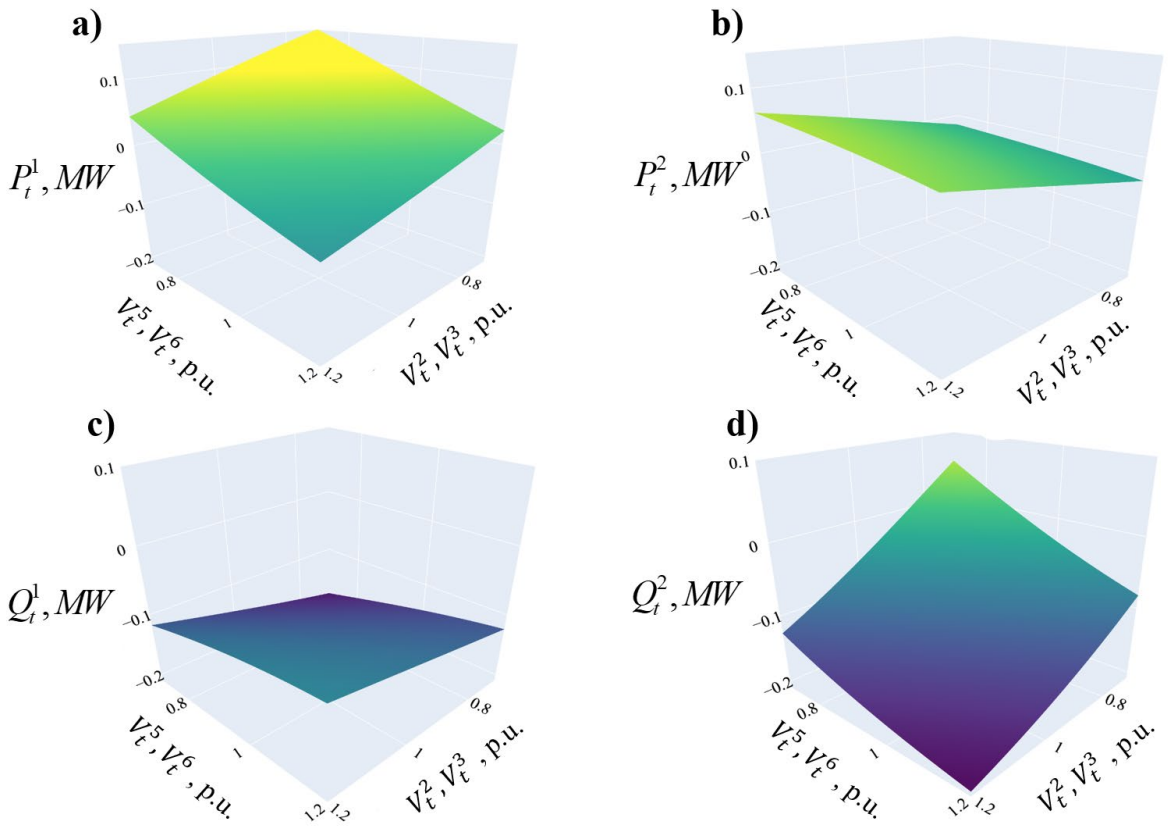


Figure 3.17. TD3PG control from different voltages of: a) Active power of battery 1 b) Active power of battery 2 c) Reactive power of inverter 1 d) Reactive power of inverter 2

The active power of battery 1 (Figure 3.17 (a)) discharges for the majority of measured voltage values, and because the active resistance of the line is twice as large as the reactance (0.59 Ohm versus 0.3 Ohm for line 48-AL1/8-ST1A 0.4 of the Pandapower library [97]), the total effect of the battery and the inverter on the node leads to the correct voltage regulation.

However, the use of such a policy by the TD3PG algorithm, where the battery and the inverter partially mitigate each other to produce the necessary control, does not seem to be optimal.

The trained PPO policy shows a more logical control behavior. All powers change in the expected direction depending on the measured voltage variations. Moreover, the reactive power of the inverters changes more than the powers of batteries, which is logical, because they do not incur any actual charge or discharge of the storage systems. In addition, flexibilities are more responsive to voltage changes at the buses where they are located. Those observations tend to confirm the better results obtained with the PPO-based controller even though TD3PG remains also acceptable compared to a base case with no control (VPI of 18.7%).

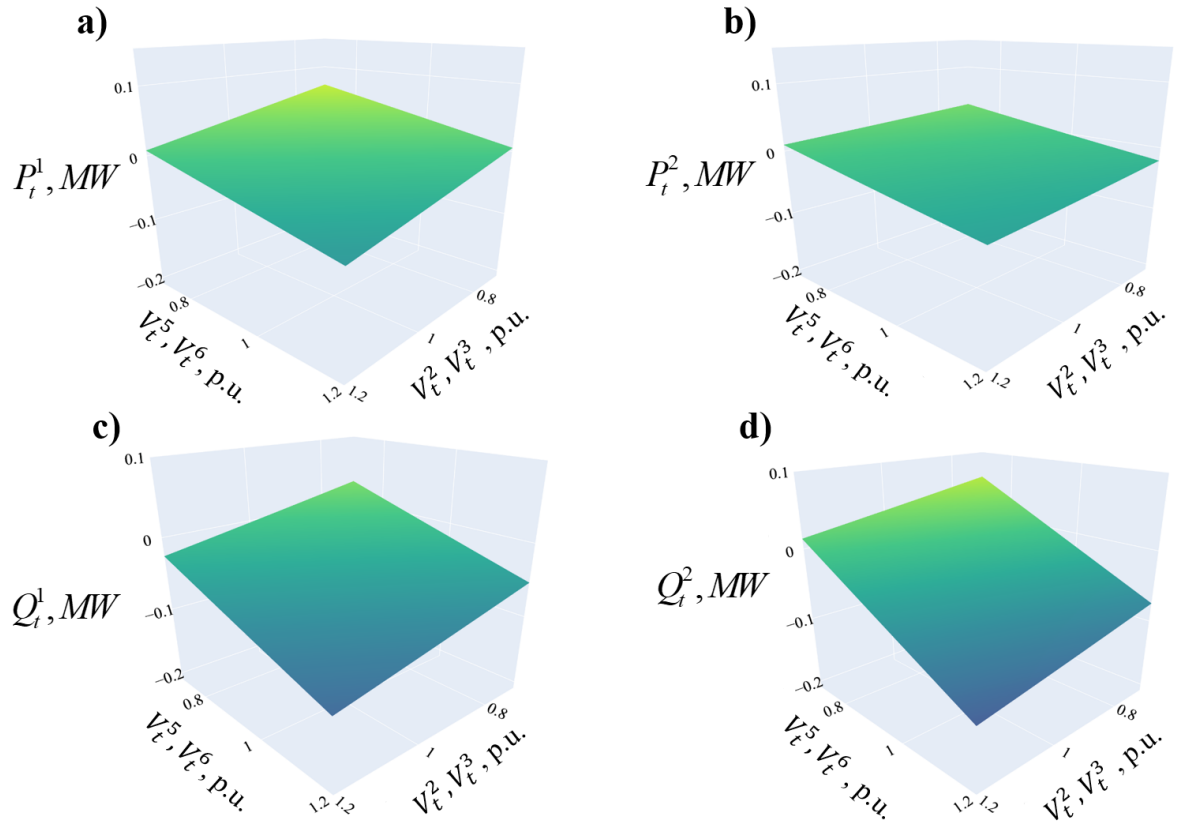


Figure 3.18. PPO control from different voltages of: a) Active power of battery 1 b) Active power of battery 2 c) Reactive power of inverter 1 d) Reactive power of inverter 2

It also should be considered that the 3D representation is limited in terms of appreciation of the results. Indeed, it allows only a rough estimate of the behavior of the controllers, since in reality, the voltages at nodes 2-3 and 5-6, although close, are never equal due to the voltage drop in the lines connecting those nodes. Therefore, RL algorithms trained to solve problems in grids with real voltage distribution may have a slightly more adequate and accurate policy compared to the behaviors displayed on the 3D representations, obtained with unrealistic assumptions. This explains why their results are very close to the optimal (VPI are 2.8% and 1.85% vs 1.29% of benchmark optimization). The most important feature of the proposed

control scheme is that the policy implicitly copes with load and generation uncertainties in the next step.

3.6 Controller performances in the test period

Due to the higher final accuracy of PPO compared to TD3PG, the PPO-based controller is considered in this section. Its performance for voltage control is assessed on the validation (test) year. The study case is the one from Section 3.3, presented in Figure 3.5, where both load and generation levels were tuned so that we can have overvoltage and undervoltage problems to solve (and thus validate the controller's performances). This case study does not represent any actual system. As an illustration, the voltage profiles of three nodes are presented, where the first has both flexibility and PV, the second has none of them, and the third has only PV. For visibility purposes, only one representative day of each season is plotted in the following figures.

The voltage profile of node 3, which contains both PV and controlled flexibility is displayed in Figure 3.19. The voltage profile before optimization deviates from 0.88 p.u. to 1.12 p.u. during the year. Undervoltages occur due to high consumption from autumn to spring while overvoltages can be seen mostly during summer days due to greater levels of solar generation. However, thanks to the controlled flexibility at the same node, the voltage is mostly maintained within the given limits once the RL-based control is applied.

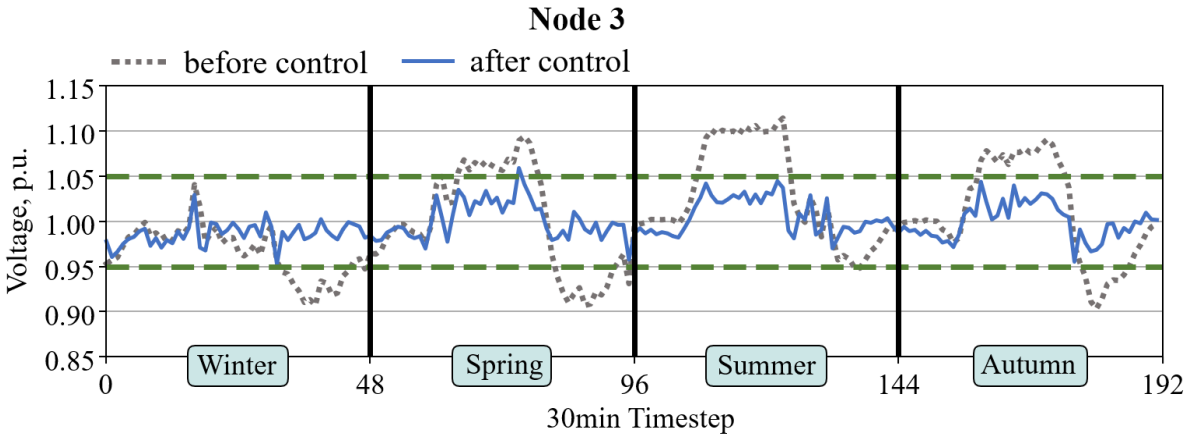


Figure 3.19. Voltage profile at node 3 of the grid presented in Figure 3.5 for four representative days of the test year with constant impedances and PPO control

Node 11 has neither PV, nor flexibility, and displays only undervoltage for the considered grid (Figure 3.20). However, its maximal undervoltage is deeper than the one observed at previous node 3 (which has a local generation of PV) and reaches 0.86 p.u. The PPO-based controller increases the voltage to the required level by injecting proper active and reactive power from neighboring nodes 8 and 10, which have flexibilities.

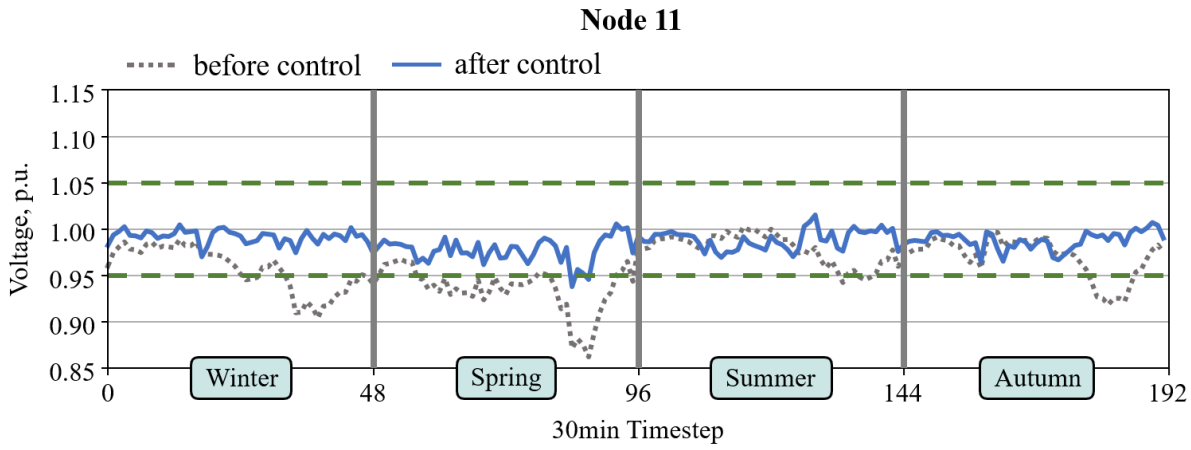


Figure 3.20. Voltage profile at node 11 for four representative days of the test year with constant impedances and PPO control

Figure 3.21 displays the voltage profiles at bus 9, which contains PV, but no flexibility. Similar to the results of bus 11, the RL-based controller mostly handles the problem of undervoltage thanks to flexibilities from neighboring nodes. However, overvoltage cannot be as efficiently mitigated as undervoltage. This is explained by the lack of flexibility in this node and the overall system layout. Moreover, the controller cannot lower the voltage of node 9 using flexibilities in nodes 8 and 10, because it would lead to a significant voltage drop in node 11 below 0.95 p.u. This is mainly related to the system topology, flexibility locations, and dynamics of load/generation profiles.

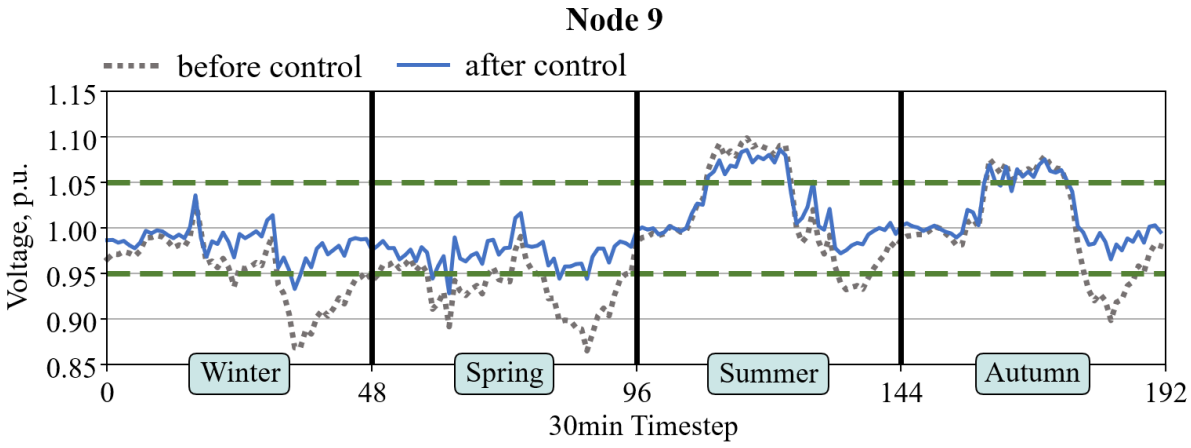


Figure 3.21. Voltage profile at node 9 for four representative days of the test year with constant impedances and PPO control.

Figure 3.22 displays the injection of reactive power from inverters 1 to 4, connected at nodes 3, 6, 8, and 10 respectively. It can be seen that injections from inverters 3 and 4 are higher than from inverters 1 and 2. This can be explained by the fact that they increase the voltage also in nodes 9 and 11, and have to compensate for the lack of flexibility on those buses. In short, the contribution of each flexibility depends on its location relative to the dynamics of consumption/generation profiles and impedances of the neighboring lines. Voltage profiles of

all 11 nodes before and after the PPO control for four representative weeks of the test year are presented in Appendix 2.

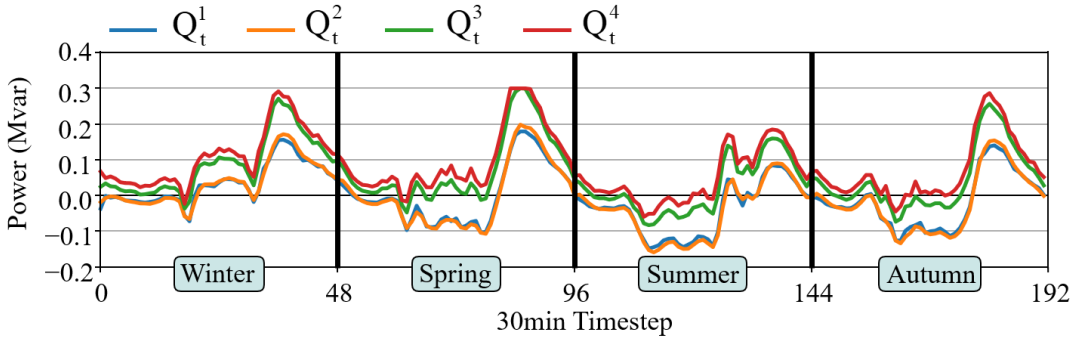


Figure 3.22. Inverter powers for four representative days of the test year with constant impedances and PPO control

The state of charge profiles of the batteries are shown for the four representative days in Figure 3.23.

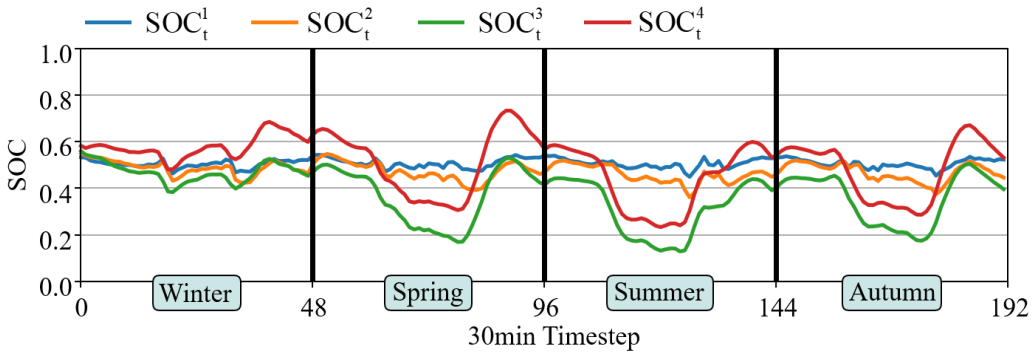


Figure 3.23. State of charge of each battery for four representative days of the test year with constant impedances and PPO control

The SOC displays daily variations in the range of 0.12 (for battery 3) to 0.75 (for battery 4). Batteries are then never completely charged or discharged, thus preserving their availability, thanks to the considered reward function that maintains SOC deviations around 0.5.

3.7 Conclusion

AI-based reinforcement learning methods using Twin Delayed Deep Deterministic Policy Gradient (TD3PG) and Proximal Policy Optimization (PPO) are proposed to control voltage profiles in a distribution grid. While adjusting the voltages, the algorithm also accounts for the battery's SOC, preventing them from being completely discharged or charged to preserve storage availability. A sensitivity analysis of the main hyperparameters was presented, which showed that TD3PG was much more dependent on the correct choice of hyperparameters than PPO. Selected hyperparameters' values allowed to decrease voltage violations from 19.5%

to 2.8% of the time for the TD3PG-based controller and to 1.85% of the time for the PPO-based controller. However, the TD3PG algorithm needed almost six times fewer epochs to be trained.

It is necessary to note, that the results of voltage regulation are highly stochastic because the training of RL algorithms depends on plenty of factors, such as the initialization of the algorithm and the environment and the random choice of actions by the RL algorithm during the exploration phase. Due to this, each new training is not identical to others, even with fixed hyperparameters. Therefore, a particular choice of hyperparameters cannot guarantee a particular accuracy. However, it can be stated, that the average result of trained algorithms with correctly chosen hyperparameters will be higher than that of other algorithms (the probability of this will be higher with an increase in the number of trained algorithms).

Learned by RL algorithms, the action-value function showed good consistency with the built reward function, making amends that it also takes into account load and generation uncertainty for the next timestep and doesn't need exact values for physically infeasible voltage operating conditions. Moreover, learned by RL algorithms policy functions were investigated and compared for both, TD3PG and PPO algorithms. The results showed that PPO uses flexibilities more efficiently, while TD3PG policy uses flexibilities too much, as a result partially compensating effect of each other.

It is important to remind that the algorithms do not need consumption/generation data thus respecting the privacy-by-design principle. They also implicitly embed a predictor part, by computing the next action based only on voltage measurements obtained 30 min before, i.e., no external forecast tools such as conventional model predictive approaches are needed. Thus, ideally, the result of the RL-based controller should be compared with what will be the best action of the expert in the same particular state, (if the expert has the same data only about this particular timestep, i.e., without knowing what exactly will happen in the future).

The proposed control scheme also presents the interest of being versatile in terms of control application. The change of reward function allows other considered objectives to be performed for grid management operations (as presented in Chapter 5). One of the main interests of the proposed scheme is its performance under uncertainties, which can be compared with a more traditional optimization algorithm. This will be covered in the following chapter.

4 Reinforcement Learning for Voltage Control Under Impedance Uncertainties

4.1 Introduction

Optimization-based voltage controllers compute the setpoints (e.g. battery charge/discharge) from simulations of the controlled system and with a prediction of its future states. In particular, their performances depend partly on their ability to mitigate uncertainties (Figure 4.1), such as forecast errors in consumption and production. However, even with perfect predictions, model-based optimization controllers' performances can be degraded due to model approximations (e.g., relaxation of load flow equations) and/or parameters that cannot be perfectly known. It is specifically the case for line impedances whose values can drift over time due to aging and weather conditions (continuous change) or by lack of knowledge of the grid, especially in low voltage grids and old facilities (abrupt change). This directly impacts the accuracy of the system equations embedded in traditional model predictive control (MPC) architectures. Thus, potentially significant errors may occur in the grid state estimation within the controllers, which consequently reduces the quality of the voltage regulation.

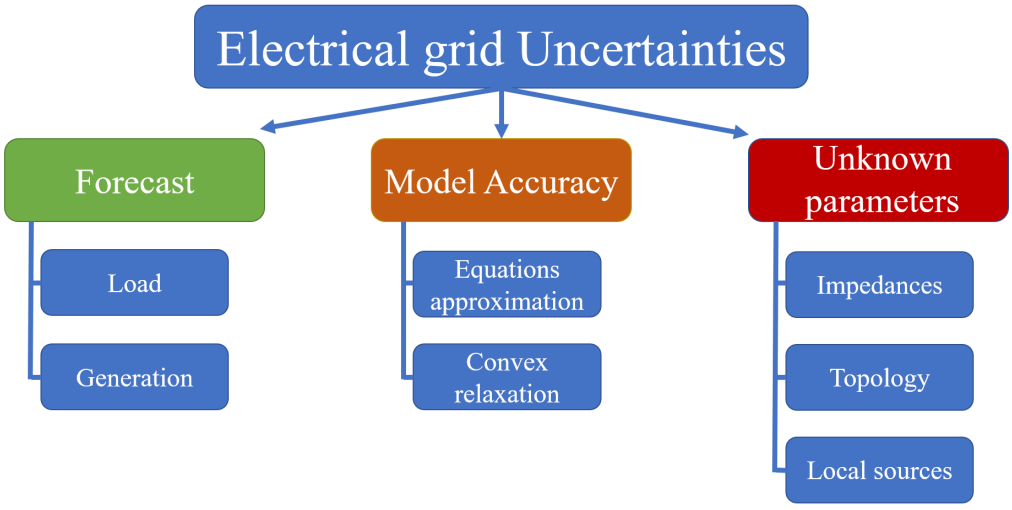


Figure 4.1. Different types of uncertainties

AI, and more precisely RL, after initial training (which may take a long time depending on the objective), is not only significantly faster than optimization-based controllers in the operational phase but can also be a solution to mitigate parameter uncertainties. In the literature,

uncertainties mitigation with RL-based controller, when considered, mostly refers to the prediction for the bus power (i.e., load and/or generation), while neglecting the lack of knowledge on the grid itself, notably impedances [45]-[51]. Thus, the study of the robustness of the control to changes in the impedance of the lines is deemed interesting and is the main motivation for the concepts developed in this chapter 4.

This chapter then presents the further application of the algorithms discussed in Chapter 3. A comparison of trained PPO and TD3PG algorithms and the conventional optimization-based approach to the robustness of the grid impedance uncertainties is conducted. Especially for cases where the impedances of the actual system differ from the simulation environment considered in the training phase. A two-stage training of PPO is then proposed and consists of offline training followed by an online adjustment of the controller. In the first phase, the RL algorithm is trained offline on the model of the grid with nominal impedances (similar to what was presented in Chapter 3). In the second stage, the RL-based controller is then connected to an actual grid (simulated for the tests here) that may display different impedance values. The controllers keep training online while exchanging actions and states with the actual grid. Such two-stage training allows for better coping with impedance uncertainties (thanks to the online phase) and avoids dangerous voltage values in the distribution grid during the online training (thanks to the offline phase). The online learning phase of PPO is verified in two cases: i) abrupt impedance change compared to the training model (original system) due to the lack of knowledge about impedance values, and ii) continual drifting of impedance values due to aging over years. Finally, the scalability of the RL algorithms is investigated by implementing the control on a distribution grid that is five times larger than the original test grid from Chapter 3.

4.2 Proposed framework

4.2.1 Baseline: conventional optimization-based voltage control

A reference optimization-based control, adapted from [98], is considered a baseline for comparison with RL algorithms. This controller is formulated as a multi-objective optimal power flow (OPF) problem with a second-order conic relaxation to account for the grid model. The main objective is to minimize the total voltage deviations over a predefined time horizon T (4.1). The problem is subject to grid and battery constraints that are not represented here for the sake of clarity but are described in [98]. The losses are integrated into the objective function so that the conic relaxation of the power flow constraint is valid. Thus, careful tuning of the objective function loss weight C_{loss} shall be conducted so that priority is given to the penalization of the voltage deviations ΔV_t^n beyond the acceptable limits [98].

$$\min\left(\sum_{t=1}^{T}\sum_{n=1}^{N}\Delta V_t^n + C_{loss}\sum_{t=1}^{T}\sum_{l=1}^{L}P_{loss\,t}^{\ l}\right) \qquad (4.1)$$

Where L is the set of lines and $P_{loss\,t}^{\ l}$ the losses in line l at time t.

In practice, and as already mentioned, such a controller relies on prediction for the bus power (i.e., load and generation) quantities. It will not be considered here, and the model-based control assumes a perfect forecast. In another word, the performances of such a controller can be used as a reference in case the parameters of the system are perfectly known. This reference can furtherly be used to assess the accuracy of the proposed control approaches. As was mentioned in section 3.4.1, only four representative weeks of the year (one per season) are used to calculate the yearly VPI, because of the long optimization time of this algorithm (around 5 hours for 16 cores, 96Gb machine).

4.2.2 Two-stage offline and online training

A two-stage training scheme (Figure 4.2) is proposed which allows, thanks to the online stage, coping with the impedance deviations of a real electrical grid by gradually adapting the controller policy to the actual impedances of this grid. Moreover, such training avoids, thanks to the offline phase, the occurrence of dangerous operating conditions in the power grid during the online training.

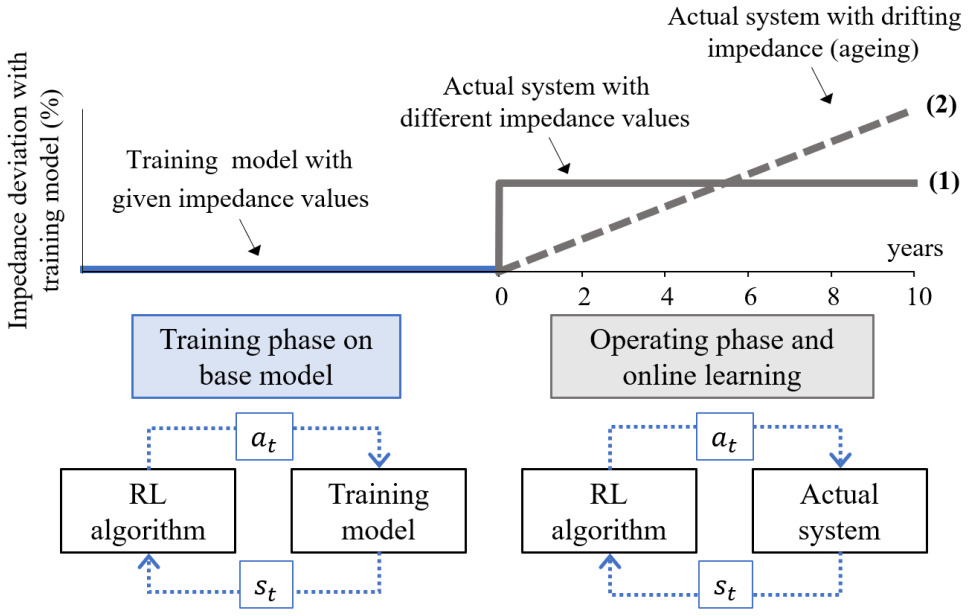


Figure 4.2. Offline and Online training for impedance uncertainties mitigation. Step variation of impedances in scenario (1) and gradual variation in scenario (2)

As already mentioned, the idea is to first train the algorithm in an offline mode on a target grid model that uses nominal datasheet impedance values. Similar to what was done in the previous chapter, the RL algorithm uses consumption and local production data over a full

year. It can run multiple simulations over the same year, testing different actions for the same states. Controller performances are assessed on test simulations with consumption and local generation profiles for a second year (over 4 representative weeks). The second step is online mode - the controller pre-trained offline is connected to the actual electrical grid.

The online training is similar to the offline phase in the sense that it uses the learning approach detailed in chapter 3. But instead of being randomly initialized at the beginning of the training, "Actor" and "Critic" neural networks (Figure 3.2) of the RL algorithm start the online training phase by being pre-trained offline on the model of the electrical grid. This grid resembles a real one, the more, the less the impedance difference. Thus, the RL-based controller should just adjust its policy instead of learning it from scratch. Trained preliminary offline-only RL-based controller connects to a real grid and starts voltage regulation using its current policy, similar to the process shown in Figure 3.7. The policy of the controller is unchanged during a fixed number of timesteps (in the considered case, 48 timesteps with a 30-min resolution - i.e., a daily update of the controller). During voltage regulation, similar to the offline phase, the RL algorithm after each timestep obtains a transition sample $[s_t; s_{t+1}; a_t; r_t]$, which is stored in the Replay Buffer. After regulating the voltage for a fixed number of time steps, the RL algorithm uses its filled Replay Buffer to slightly update the weights of "Actor" and "Critic" neural networks. The Replay Buffer is then cleared, and the RL-based controller continues voltage regulation with the updated policy. The overall process is repeated, so the RL-based controller gradually adapts the controller policy to the actual impedances of the grid, once a day in the considered study case. Two scenarios of a "real" system are considered about the impedances' uncertainties (as shown in Figure 4.2):

Scenario (1): – a significant difference between the expected impedance values (used during offline training) and the real ones (abrupt deviation). This scenario simulates a lack of knowledge, a priori, of the system, which is realistic for MV/LV networks.

Scenario (2): the impedances are close to the ones used in the grid model for the offline training, but their values continually drift over time in the operational phase due to exploitation (continual deviation). This simulates potential aging effects.

Online training depends only on the data coming from the interaction with the real system and no more data coming from an initial model that is no longer correct.

Preliminary offline training of the algorithm prepares a policy, which excludes actions that could lead to dangerous voltage values in the actual operational phase. By exchanging actions and observations with the real grid in the online phase, the algorithm can gradually adapt itself to any impedance changes. This way it is expected to improve the results compared to the ones obtained with a controller trained offline only.

4.3 Offline control performance under impedance uncertainties

In this section, the MV/LV grid with 11 nodes (Figure 3.5) and the performance metric VPI (3.11), which were previously introduced in Chapter 3, are used to assess the efficiency of the considered algorithms in the testing phase. It is to be reminded that proposed RL-based controllers are completely immune to any load forecasting errors because they rely only on measurements of voltage and power flexibilities. The optimization-based control results, on the contrary, directly depend on errors in load forecast (Figure 4.3). It has a VPI of 1.29% for 0% error (when all loads are known), but it rises to 9% already for 5% of power forecast error.

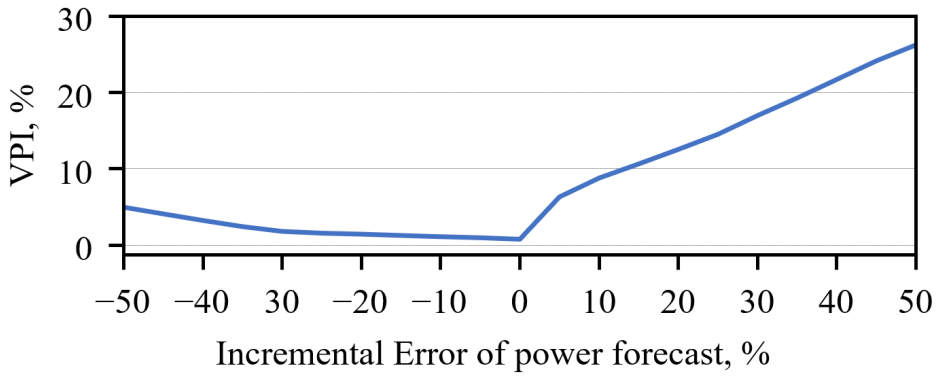


Figure 4.3. VPI of the optimization-based algorithm in the case of power uncertainties

As mentioned above, the main contribution of this chapter is to mitigate potential uncertainties arising from the lack of perfect knowledge of line impedances. To study the impact of those uncertainties on the controllers' performance, an actual MV/LV grid was simulated while varying its impedances around their original values (i.e., the ones used in the training system) with standard deviations ranging from 5% to 40% (scenario (1) in Figure 4.2). Changing the impedance values in the grid simulation tool Pandapower allowed multiple deviation scenarios to be tested, which is not feasible with a real electrical grid. Different tests were executed over T=1344 timesteps of 30 minutes corresponding to the four representative test weeks of the year while changing the environment parameters around the original values with a normal law and a standard deviation of σ% (refer to Figure 4.4 for the synoptic). For each value of standard deviations, the VPI is calculated as the average VPI over K=30 independent samples - for every k^{th} sample test a new actual system was considered with impedances of the line $Zij^{(k)}$ modified compared to the original grid embedded in the training phase.

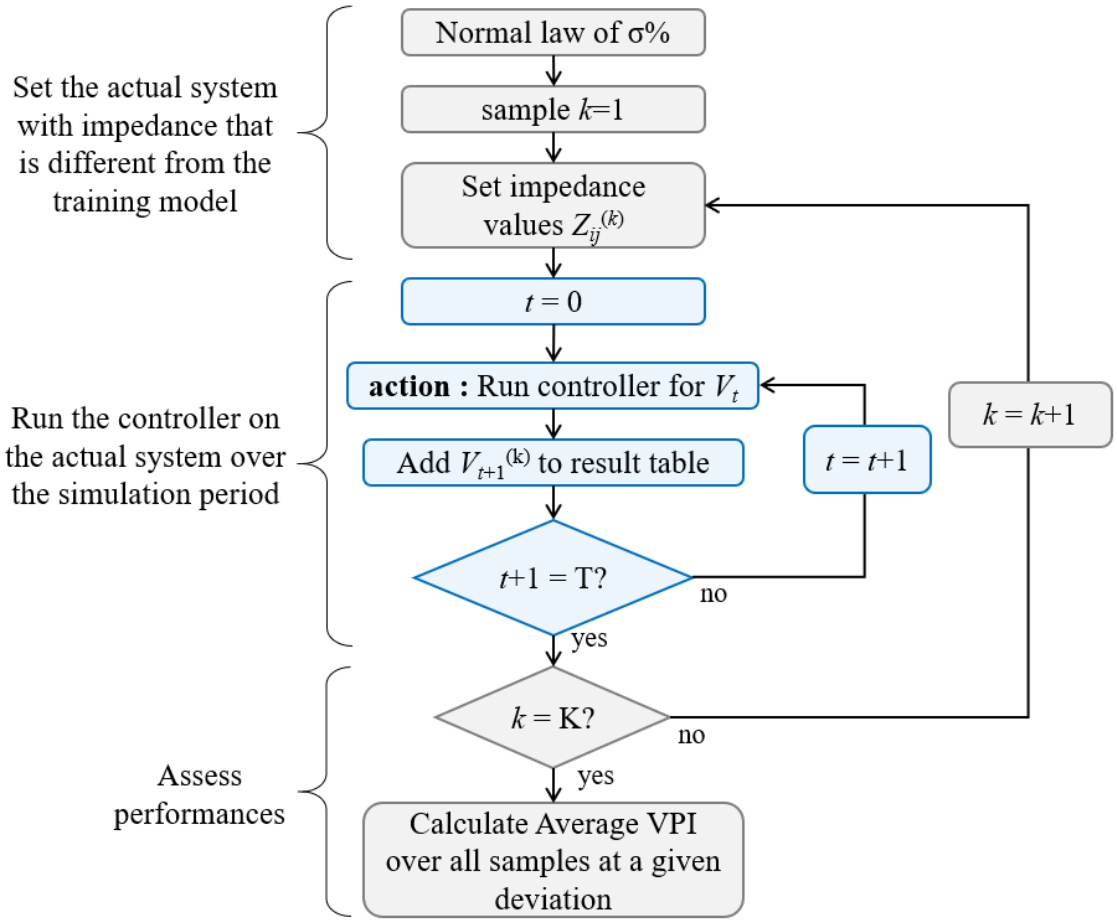


Figure 4.4. Simulation framework to estimate the impact of impedances uncertainties

To compare the optimization and the AI-based controllers, nominal impedance values are given to the optimization-based control algorithm. Its control outputs are then sent to the system with distorted loads or impedances. Similarly, the RL algorithms were trained on the model of the grid with the nominal impedances, but their control setpoints are applied to the grid with deviated values, thus modeling the unawareness of the exact values of the line parameters.

The comparison of the VPI results obtained with the offline trained controller (TD3PG, PPO) and the optimization-based approach for different abrupt impedance deviations are presented in Figure 4.5. In this figure, the "best possible VPI" (denoted by the dashed line) represents the reference optimum that can be computed with a model-based controller that integrates accurate line impedances (i.e., perfect knowledge). Confidence interval *CI* [99] is calculated according to formula (4.2) over K samples for impedance standard deviation.

$$CI = \overline{VPI} \pm Z_{crit} \frac{\sigma_{VPI}}{\sqrt{K}} \qquad (4.2)$$

Where $\overline{VPI}$ – average VPI for *n* samples,

Z_{crit} - Critical value, 1.96 for 95% confidence interval,

σ_{VPI} – standard deviation of VPI results over K independent runs with different "test systems".

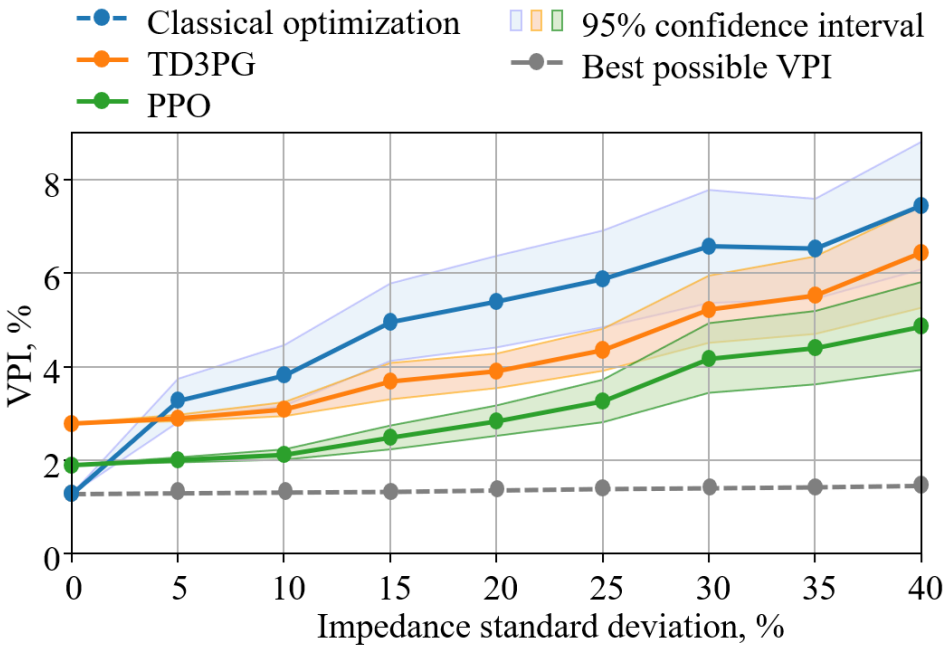


Figure 4.5. VPI of offline trained PPO, TD3PG, and optimization-based algorithms in the case of impedance step variations – average values and corresponding confidence intervals of 95 %

The optimization-based control performs better when there is no impedance deviation, i.e., when impedances are completely known (same impedance values in the model-based controller and the actual environment). Its VPI is 1.29% (vs 1.85% for PPO and 2.8% for TD3PG) and corresponds then to the theoretical optimum with perfect knowledge. However, starting from small degrees of uncertainties (from 5% of standard deviation on line impedance between training model and actual system), the RL algorithms perform better, and the improvements are maintained with an increasing lack of knowledge. Such results are explained by the fact that the proposed RL algorithms do not rely on impedance data directly or any grid model but on voltage measurements only. Those measurements are then enough to capture the system dynamic about different levels of bus power (injection/load) and flexibilities controls. It can also be noted that PPO displays conservative performances relative to the TD3PG – PPO outperformed TD3PG in the offline phase (training in Chapter 3) and for all the uncertainties on line impedances investigated here (mainly due to a more successful offline training).

It is worth reminding that the implementation of the optimization is conducted with some advantageous conditions compared to TD3PG and PPO. Indeed, the optimization relies on perfect predictions of all loads and generations for the whole considered period. As already mentioned, RL algorithms, on the contrary, make their decisions for the next timestep based on the data from the previous timestep only. Thus, TD3PG and PPO implicitly embed load and PV

generation predictors for the next timestep, and this uncertainty contributes to the final control error. For instance, this error could be reduced by decreasing the timestep due to the persistence of both load/solar profiles. However, this aspect has not been investigated here due to the lack of open-access two-year PV and consumption profiles for distribution grids with a higher time resolution.

The proposed algorithms also significantly outperform the optimization in terms of execution time once the controllers are trained offline. The computation is almost instantaneous as the optimal control are ultimately linear combinations of the input passed through activation functions. Optimization-based controllers need to package/write the problem based on real-time measurements before solving them relying on toolboxes or embedded software that can be time-consuming (even for convex problems).

4.4 Online training

4.4.1 Scenario 1: Abrupt line impedances deviation (lack of knowledge)

As previously introduced, the main objective of the work carried out in Chapter 4 is to cope with impedance uncertainties thanks to the use of the second stage of the training (online), introduced in section 4.2.2. PPO is more suitable for online training, among the two considered RL algorithms, due to the absence of the deadly triad problem that exists for TD3PG (introduced in section 3.5.1) - this problem can lead to divergence of TD3PG during the training [96]. Moreover, the on-policy approach is in line with the objective since the training depends only on the data coming from the interaction with the real system and no more data coming from an initial model that is no longer correct (thus, a very small replay buffer of PPO allows "forgetting" previous incorrect values of impedances from offline phase). First, the online training of the PPO algorithm after an abrupt deviation (scenario (1) in Figure 4.2) is investigated to demonstrate the effectiveness of the proposed two-stage method for compensating for the lack of knowledge of line impedances in the training phase. For instance, a controller trained offline only is applied to a network that displays impedances deviations of 40% (standard deviation) compared to the values used in the offline model (Figure 4.5). The deviations of line impedances for this sample (resistance R and reactance X) are presented in Table 4.1.

Table 4.1.

LINE IMPEDANCE DEVIATIONS FROM NOMINAL VALUES AT THE END OF 10 YEARS

	Line 1	**Line 2**	**Line 3**	**Line 4**	**Line 5**	**Line 6**	**Line 7**	**Line 8**
R, %	55.8	5.5	4.9	7.3	33.7	9.4	17.4	10.7
X, %	31.0	22.2	16.1	56.0	116.0	62.0	10.6	19.3

Then the controller trained offline is connected to the network and adapted online on daily basis over 10 years. The monthly VPI for this online PPO-based controller is calculated

according to equation (3.11) with T=1460 timesteps at 30 min resolution (for 1 month). It is compared with a VPI for an optimization-based controller (which is calculated in the same way), and a controller based on a PPO trained offline only. The results are presented in Figure 4.6, where the "best possible values" represent the theoretical best results obtained by optimization in the case where all impedances and powers would be perfectly known. These values cannot be reached in real life due to lack of knowledge but can be considered as a reference to assess the accuracy of other control strategies

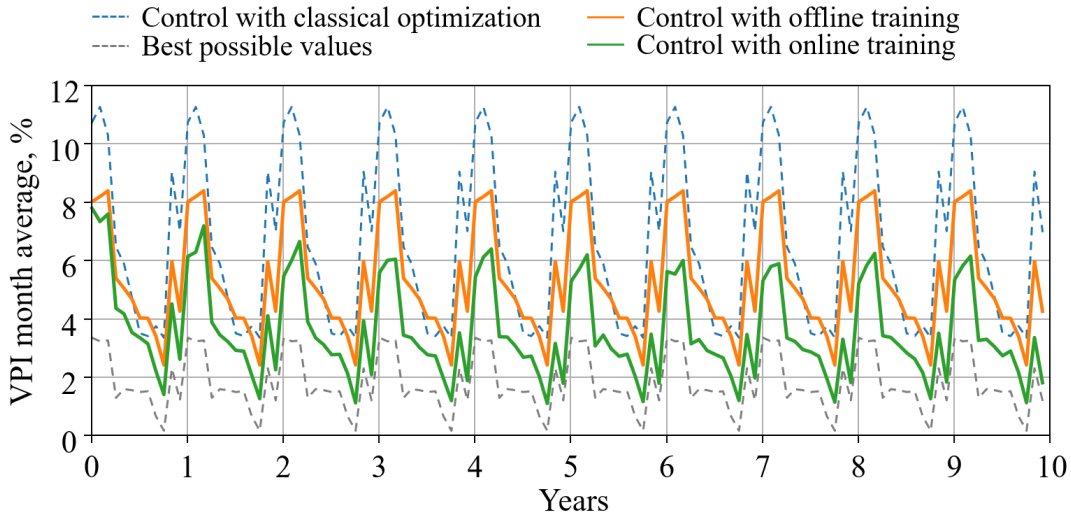


Figure 4.6. Average monthly VPI over 10 years with abrupt impedance deviation

Both optimization-based control and PPO-based controller with offline training only do not adapt and their performances remain the same over the years (same load/generation profiles are considered). At the end of the first month of the online training, the adaptive controller does not significantly improve the performance (the monthly VPI is 7.9%). However, the online training allows the controller to reach a yearly VPI of 3.48% after three years, and then fluctuate around this value. Fluctuations occur due to the daily update of the controller, which leads to the convergence of the algorithm to the local optimum (the best control for a given day) and may temporarily worsen the control for subsequent days.

For illustration, the results of controller operations over the fourth simulated year are presented in Figure 4.7 and are compared with the average weekly VPI values obtained with the benchmark optimization and the controller trained offline (PPO). The online training returns the best performances – i.e., the lowest VPI values. Such results are obtained after training on 52560 interactions, i.e., after 3 years (at 30 min resolution). The training can be accelerated by using smaller timesteps, e.g., a control with a resolution of 10 minutes would provide the same number of interactions for online adaptation after one year only.

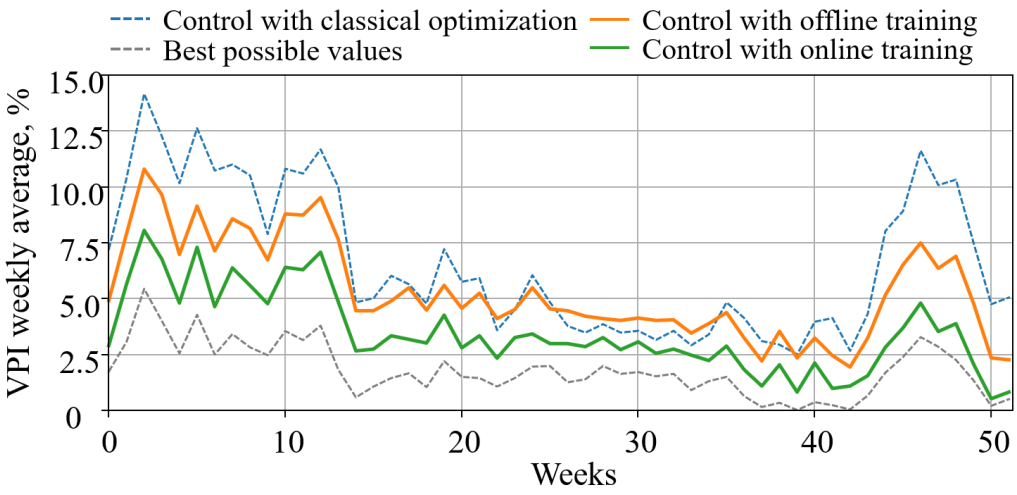


Figure 4.7. Average weekly VPI for optimization-based, offline trained PPO and online trained PPO algorithms for the fourth year of the 10 years of simulation, scenario (1)

4.4.2 Scenario 2: Results for line impedances drifting (aging effects)

The second scenario for impedance uncertainties consists in simulating a continual drifting of the values due to aging effects. In this scenario, impedances follow a daily deviation from their initial values (nominal impedances) to reach a 25% increase after 10 years, resulting in a smooth deviation as illustrated in Figure 4.2. Such a 25% value is chosen to show the performance of the controller even for large deviations. Compared to the previous simulation with abrupt changes (scenario 1), the controller shall here slowly adjust its current policy to account for constantly evolving parameters.

Similar to the previous section, the daily online training of the PPO-based controller for this impedance drifting is simulated over 10 years. The obtained results in terms of monthly VPI are shown in Figure 4.8 and compared with the performances of the model-based controller and the PPO-based control trained offline only. Over the first four years, the controller adapted online shows slightly worse results than the offline one. This undesired behavior can be explained by the daily update of the policy, which is based on the last-day data only. Thus, it leads to small oscillations of performance (convergence to the optimal control of the previous day does not necessarily mean better control for the next day). However, after four years, the impedances' deviation from their original values becomes significant, and the online trained controller is noticeably more robust to the impedance drift than the PPO-based controller with offline training only. The PPO-based controller with online training outperforms the optimization-based control already after a few months and the gap between their performance is growing with time as the model-based controller does not embed any adaptive behavior.

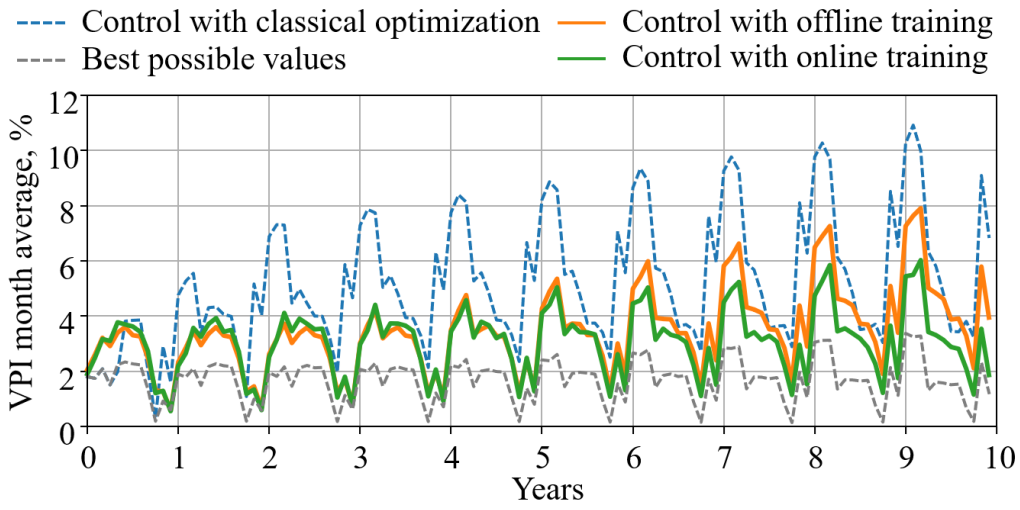


Figure 4.8. Average monthly VPI over 10 years with continual drifting of impedances

For illustration, Figure 4.9 focuses on the results over the last year with average weekly values of VPI. As can be seen, the PPO controller adapted online shows a better VPI throughout the whole year (the yearly VPI is 3.43%), compared to the case where the initial controller issued from the offline training remains unchanged (in that case, the VPI is 5.33%). The two-stage training significantly outperforms the more traditional optimization (whose VPI is 6.99%). Thus, the yearly VPI reached by the PPO with online training is more than 50% smaller than the one obtained by the optimization, which is a quantification of its robustness. However, this is still far from the best possible value of 1.82% which can never be reached in actual deployment due to unavoidable load/solar forecast errors.

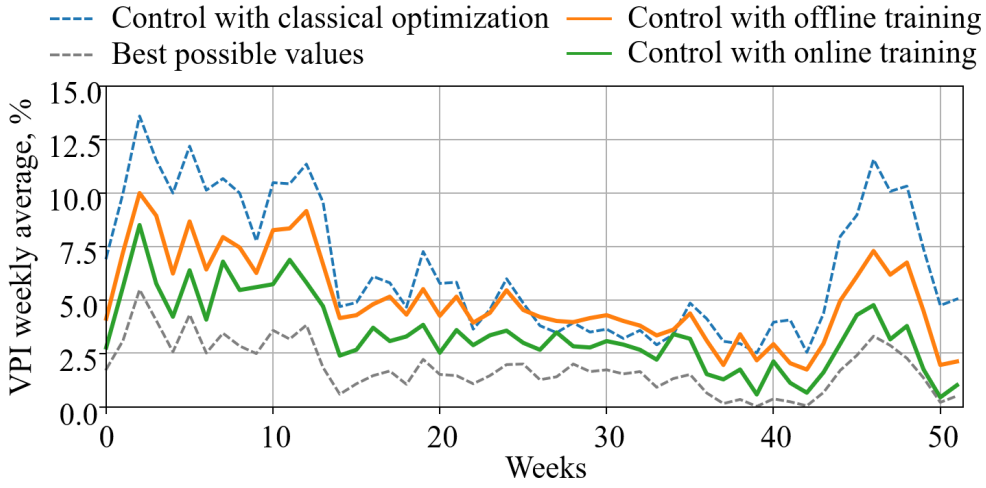


Figure 4.9. Average weekly VPI for optimization-based, offline trained PPO, and online trained PPO algorithms for the last year of the 10 years of simulation

4.5 Scalability test

4.5.1 Proposed framework

To test the scalability of the proposed RL-based controllers, a larger electrical grid with 55 nodes was modeled using line impedances from IEEE 69-bus distribution test system [100]. The grid size was chosen based on the available number of aggregated two years load profiles from the "Smart meters in London" project [69]. The electrical grid of 10 kV (Figure 4.10) contains loads at all 54 nodes (excluding the slack bus) and PV generation at 4 nodes. With the considered load/generation profiles, the voltage at the nodes varies between 0.82 and 1.11 p.u. during the year. The yearly VPI without flexibility control is 20.7% and the VPI for the four representative weeks is 20.9%. Similar to the smaller system, the grid includes four flexibilities with inverter-connected batteries with a rated power of 800 kW/kVar and a capacity of 3000 kWh. Those rated power and capacity are selected in such a way that they are not enough to eliminate all voltage violations so that we can assess the performances of the implemented control strategies. Moreover, the insufficiency of only batteries' capacity to completely smooth out voltage deviations corresponds to the current LV and MV grids.

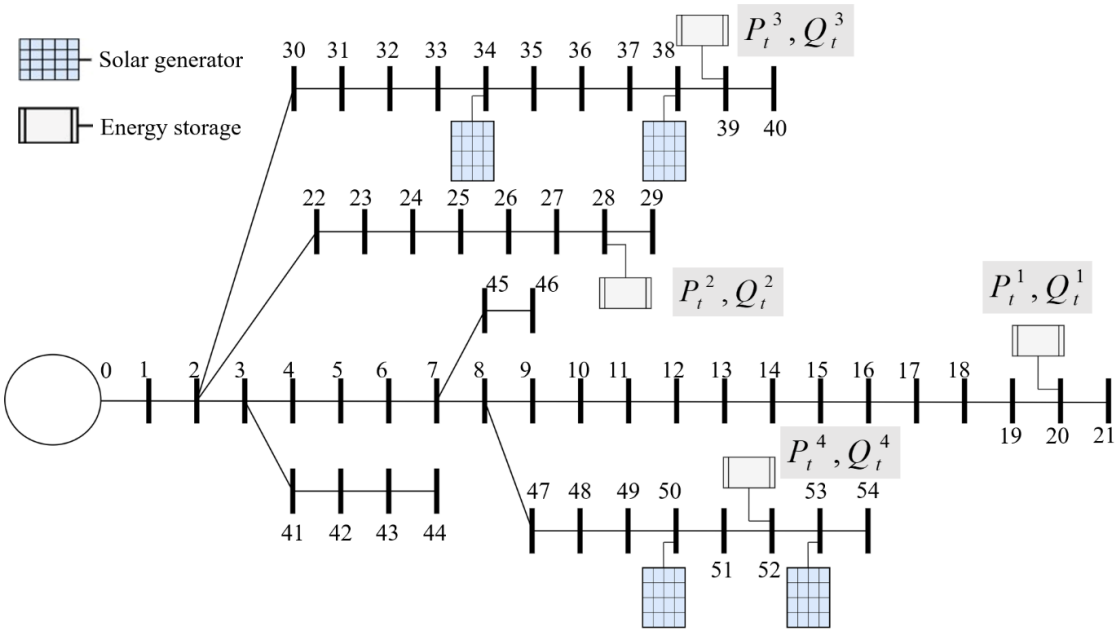


Figure 4.10. Considered MV grid with 55 nodes

4.5.2 Voltage control results for perfectly known impedances

Similar to the work carried out in chapter 3, TD3PG and PPO algorithms were trained for an entire year on the 55-bus grid, but during 15,000 epochs. Over the training, both considered RL algorithms were tested on the four representative weeks of a second test year for the same grid (with fixed impedances) every 50 epochs. Their best results, as well as the results of the optimization-based controller, are presented in Table 4.2. An example of a grid search to find the best set of hyperparameters resulting in the best VPI is given in Appendix 5.

Table 4.2

***EPI** RESULTS FOR DIFFERENT CONTROLLERS*

	Without control	Optimization	TD3PG	PPO
VPI, %	20.9	6.0	7.5	8.0
Number of epochs for the best VPI	-	-	5950	2100

Surprisingly, the number of epochs for the best VPI is comparable to the one obtained for offline training on the smaller 11-bus grid. It then seems that the complexity of the task here is not impacted by the size of the system. This can be also explained by the same number of nodes in the hidden layers of neural networks (32x32). However, the time of grid simulation by the Pandapower tool increases with the complexity of the grid, which impacts the overall training time of the RL algorithm offline. Execution time of the controller once trained remains almost instantaneous (RL-based controller takes a fraction of a second to output 1-step control).

The resulting VPI of the TD3PG-based controller is 7.52%, which is close to the result of the optimization-based controller of 6.02%. This is also better than the best result for the PPO-based controller (7.96%). Better results of TD3PG can here be explained by bigger state space compared to the previous case study with the 11-bus grid. TD3PG handles a total of 67 input features compared to 24 features for the 11-bus grid. The second reason is the insufficient capacity of flexibilities - even with optimal control of flexibilities, the voltages are outside of the limits 6% of the time. Thus, the difference between s_t and s_{t+1} at a late stage of training on the 55-bus grid is more noticeable compared to a previous case study with an 11-bus grid, and TD3PG does not fall into the deadly triad divergence problem (introduced in section 3.5.1). Finally, PPO updates its policy after each epoch and resets its replay buffer. However, the number of transition samples after only one epoch may not be enough for such a large state vector to effectively converge to the global optimum. TD3PG, on the contrary, due to its large replay buffer, can be trained on transition samples from thousands of different epochs, which increases the generalization of the policy.

The examples of voltage profiles before and after TD3PG control for node 54 (which is the last node in a feeder with two PV generations and one flexibility) and for node 21 (which is the last node in a long feeder with only one flexibility) are shown in Figure 4.11 and Figure 4.12, respectively.

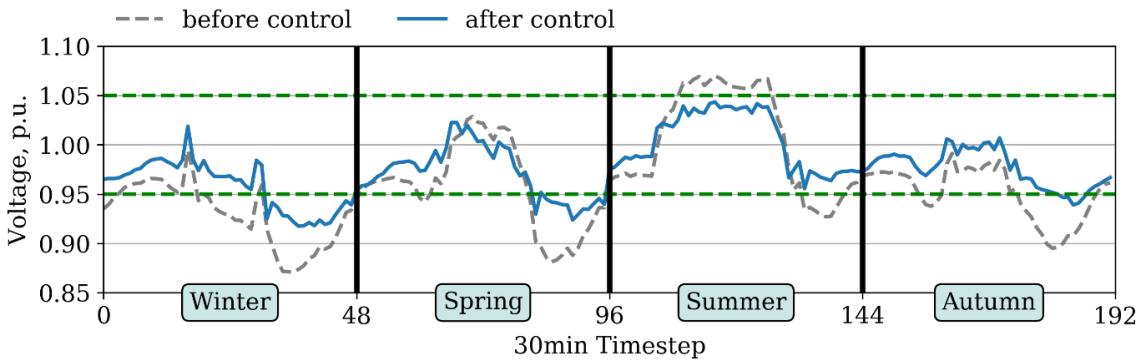


Figure 4.11. Voltage profile at node 54 for four representative days of the test year with TD3PG control

The RL-based controller shows good regulation of overvoltages incurred by high levels of PV generation (Figure 4.11). Undervoltages are globally smoothed, but the controller cannot perfectly mitigate the deepest sags. Thus, the resulting voltage profiles display values that may be outside the specified limits (below 0.95 p.u. in this case). This is especially noticeable for node 54, which doesn't have PV generation units in its feeder. The main reason for such resulting profiles is the lack of additional flexibilities in the grid and the insufficient capacity of existing ones. The resulting VPI of the benchmark optimization-based controller (6%) also confirms this statement. Thus, such VPI is due to the chosen use case here and not because of any wrong control. However, the insufficiency of capacity is a common situation for the flexibility controllers in current electrical grids, thus it is interesting to investigate control under such conditions.

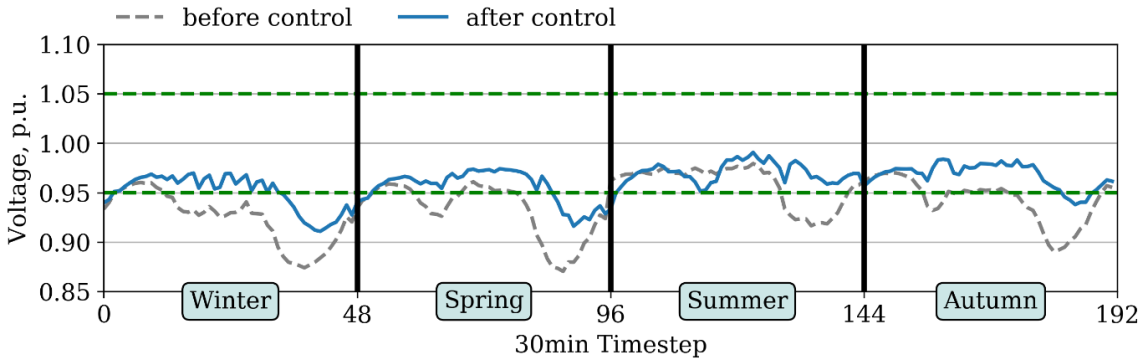


Figure 4.12. Voltage profile at node 54 for four representative days of the test year with TD3PG control

4.5.3 Voltage results under impedance uncertainties

The performance of both RL-based controllers trained offline, TD3PG-based and PPO-based controllers, is tested under impedance uncertainties similarly to section 4.3 and compared to the model-based optimization in Figure 4.13. RL-based controllers outperform optimization-based controllers starting from 5% of impedance standard deviation, and the difference between the results of the controllers increases with standard deviation. The smaller difference between

the VPI results of the RL-based controllers and the optimization-based controllers for small standard deviations (less than 15%) compared to the results for the 11-bus grid is explained by the insufficiency of flexibilities for better control in this considered case. The available flexibilities are already working to their limit, but even with perfectly known parameters (for optimization), they cannot eliminate 29% of voltage violations, still getting 6% of VPI compared to the original 20.9%. It can be noted that unlike to VPI of the optimization-based controller, the VPI of RL-based controllers does not noticeably grow up with an increase of impedance deviation after 5%. Thus, for the considered grid, information about the actual voltages at the nodes is enough for RL algorithms to compute quite accurate control even without knowing the real impedance values and without online learning. Confidence intervals for RL-based controllers are also significantly narrower compared to the confidence interval of optimization, thus the results of the RL algorithm are more predictable.

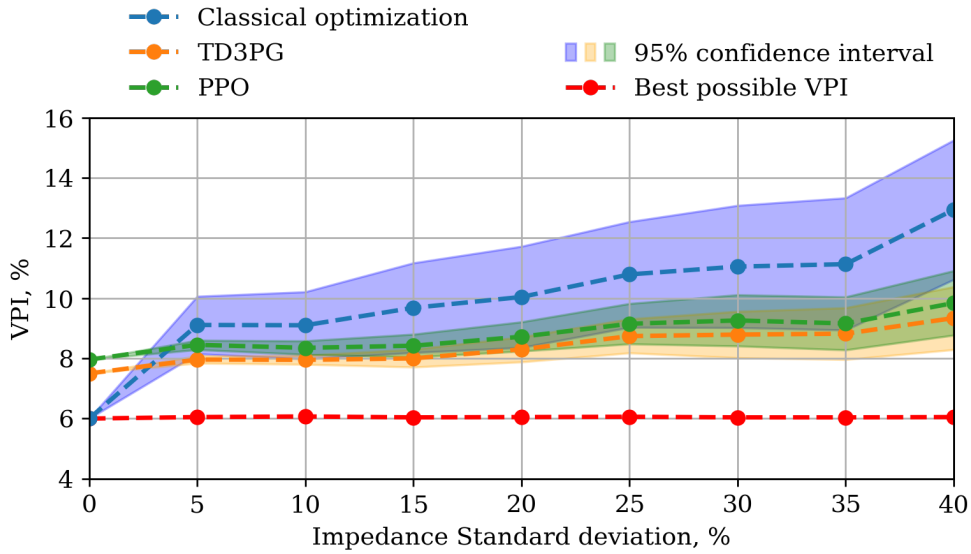


Figure 4.13. VPI of offline trained TD3PG, PPO, and optimization-based algorithms for impedance step variations of 55-bus grid

4.5.4 Online training

Finally, the second stage of the training (online, introduced in section 4.2.2) is tested for the management of a 55-bus electrical grid. For this, the online training of the PPO algorithm after an abrupt deviation (scenario (1) in Figure 4.2) is investigated. PPO-based controller trained offline only is applied to a network that displays impedance deviations of 40% (standard deviation) compared to the values used in the offline model. VPI of the controller trained offline is 9.9%. Then this controller is connected to the grid and adapted online on daily basis for over 10 years. The monthly VPI for this online PPO-based controller is calculated according to equation (3.11). It is compared with a controller based on a PPO trained offline only. The results are presented in Figure 4.14.

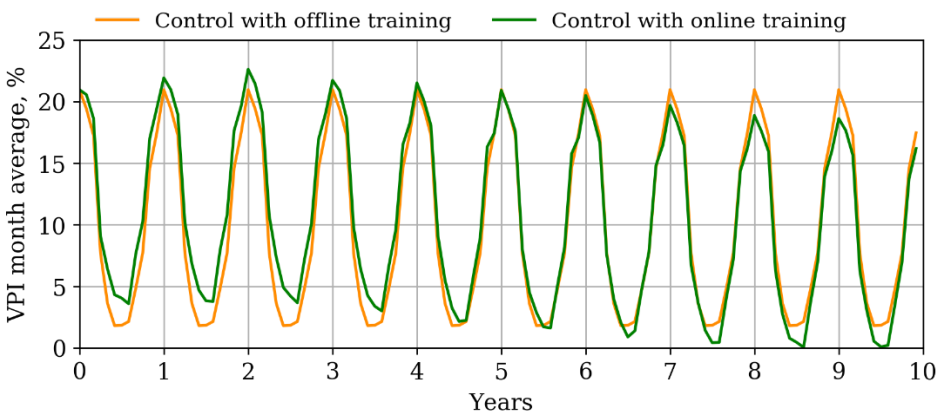


Figure 4.14. Average monthly VPI over 10 years with abrupt impedance deviation

The PPO-based controller with offline training only does not adapt and its performance remains the same over the years (same load/generation profiles are considered). During the first four years, the adaptive controller performs slightly worse, because it needs time to adapt to new impedances. After seven years, the online training allows the controller to reach a yearly VPI of 8.2% and then fluctuate around this value. Fluctuations occur due to the daily update of the controller, which leads to the convergence of the algorithm to the local optimum (the best control for a given day) and may temporarily worsen the control for subsequent days. For illustration, the results of controller operations over the tenth simulated year are presented in Figure 4.15 and are compared with the average weekly VPI values obtained with the controller trained offline.

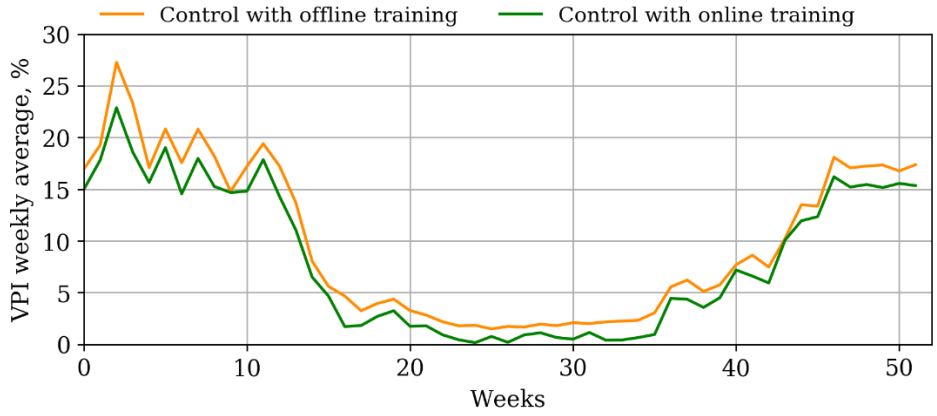


Figure 4.15. Average weekly VPI for optimization-based, offline trained PPO and online trained PPO algorithms for the fourth year of the 10 years of simulation, scenario (1)

The lowest VPI for both controllers is in summer when PV compensates for most of the excess load (as can be seen in Figure 4.11 and Figure 4.12). Similar to the previous results, the online training returns the best performances – i.e. the lowest VPI values. The rather modest difference between offline and online results is explained by a good performance of controller trained offline even for big impedance deviations (VPI is 9.9% for 40% standard deviation compared to VPI of 8% with known impedances).

4.6 Conclusion

An AI-based reinforcement learning method using the Twin Delayed Deep Deterministic Policy Gradient (TD3PG) and Proximal Policy Optimization (PPO) was proposed to control voltage profiles under load and impedance uncertainties in a distribution grid. A more traditional optimization-based control was introduced as a benchmark and a reference model. Especially it allowed the theoretical best performances to be computed in cases there are no uncertainties in the voltage control (i.e., nor forecast, nor impedance uncertainties).

A comparison of the two proposed RL algorithms with the optimization-based approach showed that the AI-based voltage control is particularly more robust to line impedance uncertainty after 5% of deviations from theoretical values. In addition, a two-stage training strategy for PPO was proposed that consisted of offline and online learning phases. The proposed two-stage control outperforms, in terms of accuracy, the optimization-based approach, by decreasing voltage violations by 55% for the considered case (11-bus system). This result is valid for both scenarios that were tested to account for impedance uncertainties - i.e., abrupt impedance deviation (which models unknown impedances) and impedance drifting (which models aging of the lines).

The scalability of the solution was tested on a distribution grid five times larger than the initial one, showing that RL algorithms still effectively control the voltage even for a significant number of buses. Both RL algorithms are more efficient compared to the optimization-based controller. Offline trained TD3PG-based and PPO-based controllers reach 9.34% and 9.84% of average VPI, respectively, compared to 12.94% for optimization control under the same 40% impedance uncertainty. Online training on a 55-bus grid also showed a performance increase from 9.9% of VPI to 8.2%. Thus, the RL-based controller can be considered a robust replacement for the traditional Model Predictive Control (MPC) in real distribution grids with parameter uncertainty.

5 Reinforcement Learning for Power Control in a Distribution Grid

5.1 Introduction

The power exchange between the distribution and transmission grids plays an important role in ensuring safe and reliable operating conditions for both systems. However, these systems are managed independently by two distinct operators - transmission system operators (TSOs) and DSOs, which complicates the overall control. Various control strategies used in distribution grids may strongly affect the power exchange at the TSO-DSO interface, may eventually cause some undesirable effects in transmission grids, and make it difficult to predict the regimes and control them [101]. Moreover, in recent years, system operators have faced new challenges due to the increasing penetration of renewable energy sources (RES) and other DER [102]. As an example, an excess of local generation can lead to a reverse power flow from the distribution grid to the upstream transmission system, which may also cause a voltage increase under low load conditions and complicates at the same time the consumption/production balance control for the TSO. Also, TSOs use controllable power plants, usually based on fossil fuel, for a variety of ancillary services, including voltage and power regulation tasks. However, along with the transition to renewable energy, reducing the use of fossil fuels for such tasks is a necessary action. For these reasons, the TSO may request DSOs to maintain the power consumptions, both active and reactive, at the substation level under a predetermined limit. Violation of these limits may entail penalties for the DSOs, which increase dramatically with the duration and depth of the violation.

Reinforcement learning algorithms may be of interest for the objective of power exchange control, because, unlike the traditional optimization-based approach, when properly trained, they do not need information about the grid or the measurements of each consumer.

In this chapter, the use of RL-based controllers is investigated in two cases. The first use case is dedicated to power exchange regulation at the TSO-DSO interface. Different power limits are tested, including a fixed apparent power limit, an apparent power limit that changes over time, and an active power limit for injection into the transmission grid. The second use case is dedicated to power exchange control together with voltage regulation. The trade-off between power control and voltage control is presented for both TD3PG and PPO algorithms. Their sensitivity to the size of the training dataset is investigated. The test of algorithms'

performances without any information about grid topology, its parameters, or the bus voltages is conducted. Finally, a scalability test is implemented by controlling a grid with 55 buses.

5.2 Proposed framework

To test the control of the power exchange at the TSO-DSO interface, both PPO and TD3PG algorithms previously introduced and the same case system (Figure 5.1) are considered. The power is controlled at the substation level and is characterized by aggregated active power P_t^s and reactive power Q_t^s - equal to the sum of all generations, consumptions, losses in the grid, and flexibilities contributions at each timestep *t*. Consumption is represented by positive values, and injection into the transmission grid is represented by negative values.

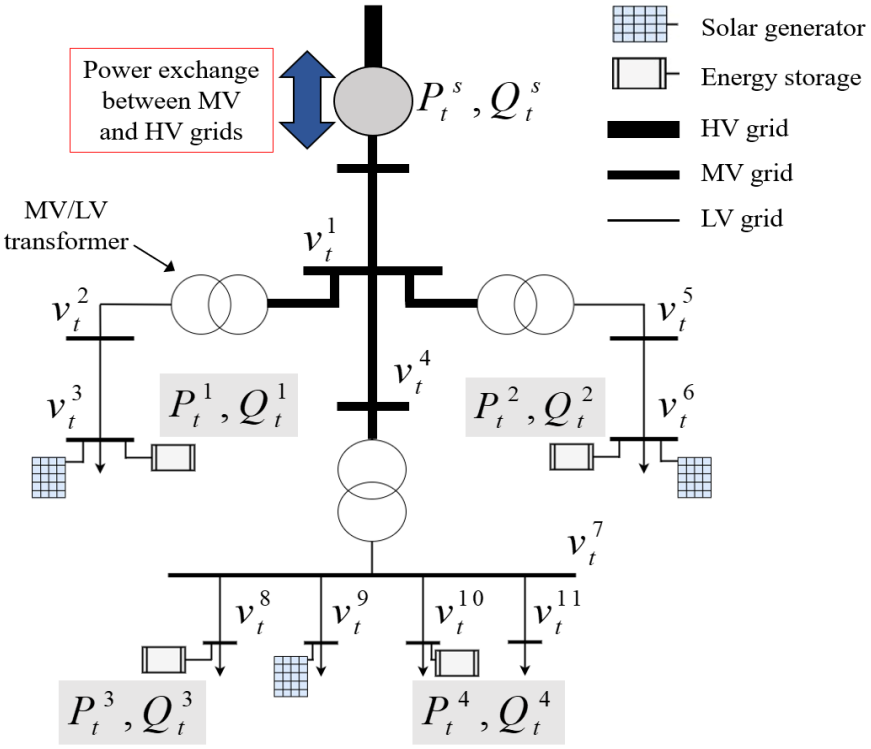


Figure 5.1. Considered MV/LV grid for power exchange tests

The control actions remain the same as before for the voltage regulation– i.e., the active/reactive power references for four flexibilities at the next timestep *t+1* (3.9). The state of the system s_t is similar to the previous one (3.8) (i.e., it contains the voltages at each node $\mathbf{V_t^n}$, the current powers of flexibilities ($\mathbf{P_t^f}, \mathbf{Q_t^f}$), along with the SOC of the batteries $\mathbf{SOC_t^f}$ and the hour *t*). But additionally, two new features are introduced with the substation active and reactive powers (P_t^s, Q_t^s) (5.1). The use of those aggregated measurements at the substation level allows the privacy-by-design principle to be fulfilled (because DSO does not use individual consumption measurements) and reduces the necessary number of meters.

$$s_t = \left[\mathbf{V_t^n}, \mathbf{P_t^f}, \mathbf{Q_t^f}, \mathbf{SOC_t^f}, t, P_t^s, Q_t^s \right] \quad (5.1)$$

The apparent power profile at the substation for four representative weeks of the test year, without flexibilities, is shown in Figure 5.2. For the initial testing, the power limit is fixed at S_{lim}=0.6 MVA. Thus, the grid with the considered system layout and demand/generation profiles has the largest violations of power exchange limit during winter and spring days and does not have any power exchange limit violations in summer due to higher PV production.

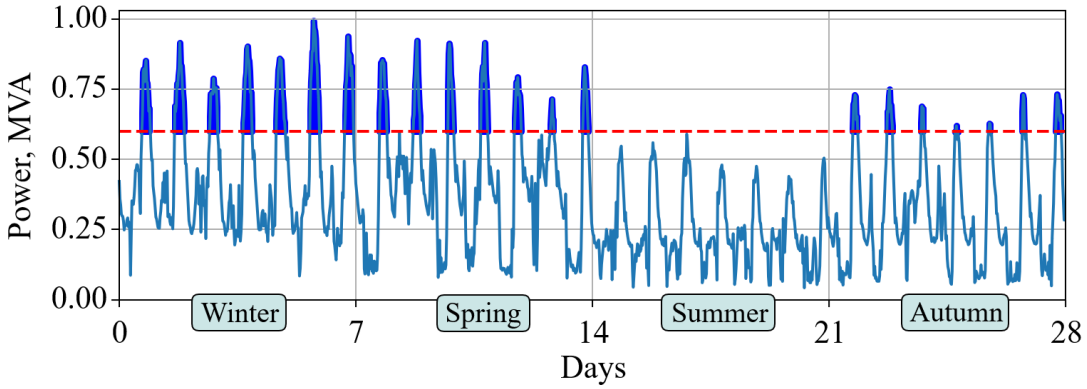


Figure 5.2. Apparent power profile at the substation for four representative weeks of the year

To assess controller performance to regulate power exchange, the energy performance index (EPI) is defined in equation (5.2):

$$EPI = \sum_{t=1}^{T=1344} \max(0, (\sqrt{(P_t^s)^2 + (Q_t^s)^2} - S_{\text{lim}})) \cdot dt \quad (5.2)$$

This proposed metric shows how much energy is injected or consumed (i.e., absolute value in this case) beyond the set limit. In opposition to VPI, EPI takes into account not only the number of timesteps, when the power violates the limit but also how much it exceeds this limit (which is important for the DSO). The EPI over the four test weeks without flexibilities is equal to 13.8 MVAh, which is then the objective to minimize.

A new reward function r_t is then set for PPO and TD3PG algorithms, imposing them to simultaneously control voltages and powers (5.3) (as will be discussed in section 5.4.1, to obtain the best power control, minimal voltage control is interesting). It consists of three parts, where the first part penalizes voltage violations V_{t+1}^n for each bus n of the grid (5.4), the second part penalizes SOC deviations SOC_{t+1}^f of each flexibility f (5.5) and the third part penalizes apparent power exchanges beyond the limit S_{lim} (5.6). The first two parts are introduced in section 3.3, and the third part calculates the value of exceeding the apparent power limit S_{lim} in both directions (export and import). The penalty increases quadratically with the increase of power

surpassing, which imposes the algorithm to prefer a long small surpassing rather than a short but large one.

$$r_t = \alpha \cdot r^1{}_t + \frac{1}{\beta} \cdot r^2{}_t + \tau \cdot r^3{}_t \tag{5.3}$$

Where τ is a penalty coefficient for exceeding the power limit. Coefficients *α, β,* and τ are responsible for voltage, SOC, and power exchange objectives. By varying α and τ values it is possible to change which control task should be preferred.

$$r^1{}_t = -\sum_{n=1}^{N} \frac{(max(0,(V_{t+1}^n - 1.05))^2 + max(0,(0.95 - V_{t+1}^n))^2)}{N} \tag{5.4}$$

$$r^2{}_t = -\sum_{f=1}^{F} \frac{exp(\omega \cdot |\, SOC_{t+1}^f - 0.5\,|)}{F} \tag{5.5}$$

$$r^3{}_t = -\max(0,(\sqrt{(P_t^s)^2 + (Q_t^s)^2} - S_{\text{lim}}))^2 \tag{5.6}$$

5.3 Power control results

5.3.1 Best EPI results

As a result of a sensitivity analysis (which is done by dynamic tuning, i.e., adjusting one parameter at a time to the direction of a lower EPI until the local optima), the values of α and τ corresponding to the best-found EPI values for both RL algorithms, are presented in Table 5.1 (results in detail are presented in section 5.4.1). As can be seen, both algorithms have a non-zero α parameter, which regulates the importance of the voltage control objective, because voltage and power are interdependent. Thus, a small incentive to maintain the voltage level allows the RL algorithm to more easily find an optimal power control (it is discussed further in section 5.4.1), even if the voltage regulation is not itself the main objective. Here, PPO needs three times lower voltage penalization coefficient compared to TD3PG, to maximize the power exchange control objective. The difference between the best parameters of the algorithms' reward function can be explained by the complexity of the reward function and by the different approaches of the algorithms, which can lead to their convergence to different local optima.

Table 5.1		
REWARD PARAMETERS FOR BEST EPI RESULTS		
	TD3PG	PPO
α	3	1
τ	2.5	3

As previously done for the voltage control, a model-based optimization controller was used as a reference for comparison of RL-based controllers' results. This controller is based on the voltage optimization from section 4.2.1, but its main objective is replaced by the minimization of total apparent power flow violations beyond the limit at the substation/slack bus over a predefined time horizon T (5.7):

$$\min\left(\sum_{t=1}^{T}\left(max\left(0,\sqrt{(P_t^s)^2+(Q_t^s)^2}-S_{lim}\right)\right)+C_{loss}\sum_{t=1}^{T}\sum_{l=1}^{L}P_{losst}^{\ l}\right) \qquad \textbf{(5.7)}$$

EPI results and corresponding training time for TD3PG, PPO, and reference model-based optimization control are given in Table 5.2.

Table 5.2				
EPI RESULTS FOR DIFFERENT CONTROLLERS				
	Without control	Optimization	TD3PG	PPO
EPI, MVAh	13.8	0.12	0.13	1.07
Number epochs for the best EPI	-	-	2300	3400

TD3PG displays better results compared to PPO, very close to the benchmark optimization. PPO performs worse for the objective of power control because the reward now has 3 objectives, and it is more difficult for this case to converge to a global optimum without a big enough replay buffer – it is further discussed in section 5.4.2. Due to the higher final accuracy, the power control results in this section are presented for the trained TD3PG algorithm. The apparent power profiles before and after using the TD3PG-based controller for four representative days of the test year are shown in Figure 5.3.

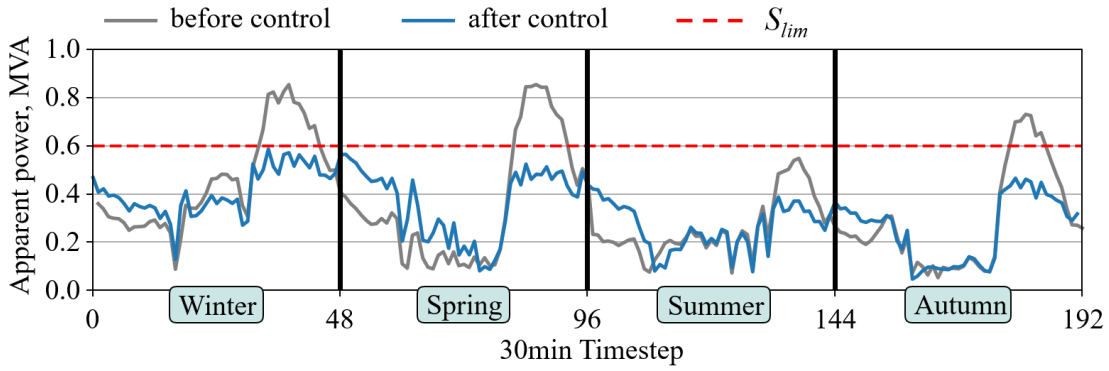


Figure 5.3. Apparent power substation profiles before and after TD3PG control

TD3PG has almost no problems to compensate for excessive exchange, successfully limiting the apparent power below S_{lim} with some extra margins to cope with the absence of load and generation forecast on the next step $t+1$. In other words, the algorithm limits apparent power to values less than the limit of 0.6 MVA. The algorithm chooses this additional margin itself to get the highest possible cumulative reward (according to its expectations). Considering active and reactive power results of flexibilities separately, it can be seen that the algorithm maintains the reactive power exchange close to zero with an amplitude deviation of less than 0.08 MVar (Figure 5.4 (a)). As for the active power control, it mainly compensates for the aggregated consumption, which exceeds 0.5 MW (Figure 5.4 (b)).

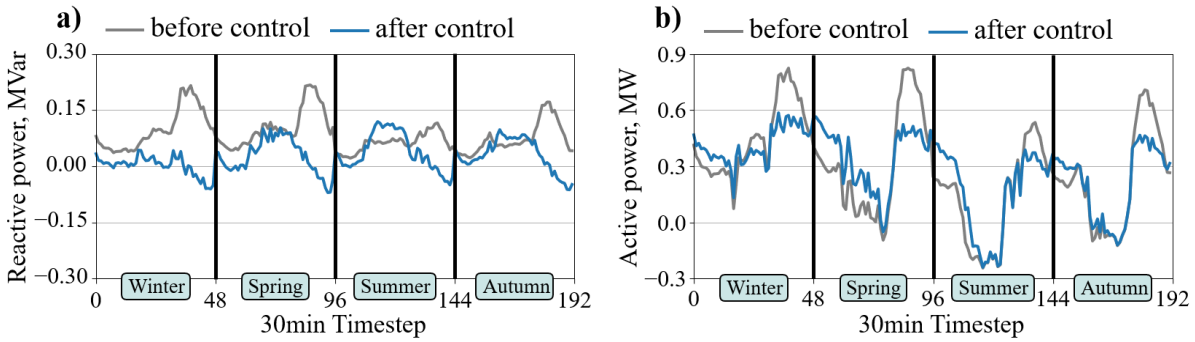


Figure 5.4. Reactive (a) and active (b) power substation profiles before and after TD3PG control for four representative days

By analyzing the SOC profiles for the same days (Figure 5.5), the lowest SOC occurs during the winter, when PV production is at the lowest level and batteries cannot be fully charged before discharging in the evenings to mitigate excessive consumption. By contrast, batteries are fully charged around 35% of the time on summer days, when active power injection into the transmission grid reaches 0.25 MW (Figure 5.4 (b)).

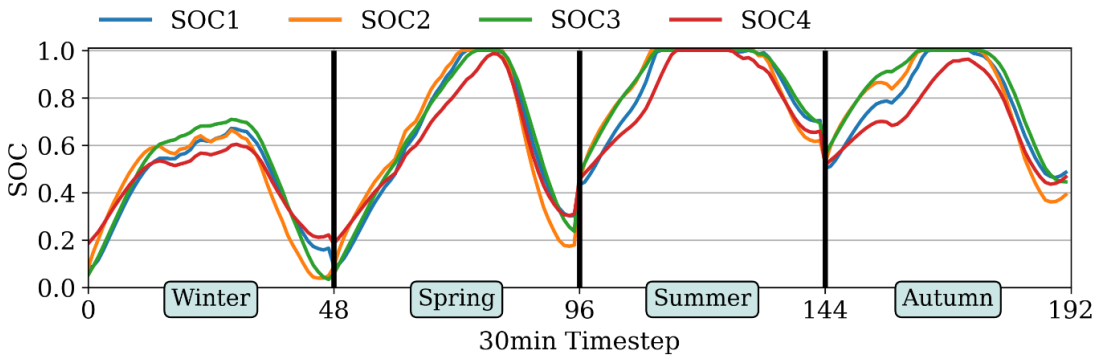


Figure 5.5. SOC profiles for four batteries during four representative days

5.3.2 EPI results for best voltage control

Despite the near-perfect control of the power exchange by the TD3PG-based controller with an EPI of 0.13 MVAh, the corresponding VPI for the considered grid is significant at 14.3% (compared to 2.8% of VPI for voltage control). For testing the hypothesis about the impossibility of completely fulfilling both the voltage control and power control tasks simultaneously for a given grid, five pieces of training are executed with $\tau = 0$, thus forcing the algorithm to maintain only the voltage at the limits, but with information about active and reactive power at the substation (5.1). In that case, the best VPI of TD3PG reaches 2.38%, because thanks to the increased state of environment s_t (compared to the state for pure voltage control (3.8)) TD3PG is less susceptible to the deadly triad problem and has more information about the grid. The resulting apparent power (Figure 5.6), reactive power (Figure 5.7 (a)), and active power (Figure 5.7 (b)) profiles over four representative days of the test year show a worsening of apparent power regulation compared even to power exchanges without any flexibilities control (the blue curve after flexibility control exceeds the limit to a greater extent compared to the grey curve without control).

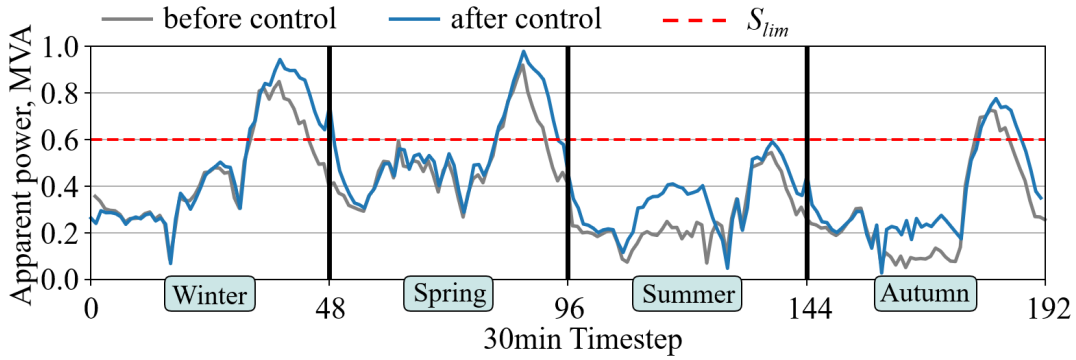


Figure 5.6. Apparent power profile for four representative days before and after voltage control by TD3PG

It can be seen that the largest contribution to the increase of apparent power is made by the reactive power injected by the flexibilities. The injection of inverters' reactive power into the transmission grid (negative values) is increased in the evening to compensate for the undervoltage due to the active power consumption during the same hours. Such undervoltages

are demonstrated, for instance, for nodes 11 and 9 in Figure 3.20 and Figure 3.21, respectively, before control. Thus, the total energy exchange between the distribution grid and the transmission grid beyond the limit reaches 19 MVAh. It is 1.5 times greater than the EPI without flexibility regulation. All results are summarized in Table 5.3.

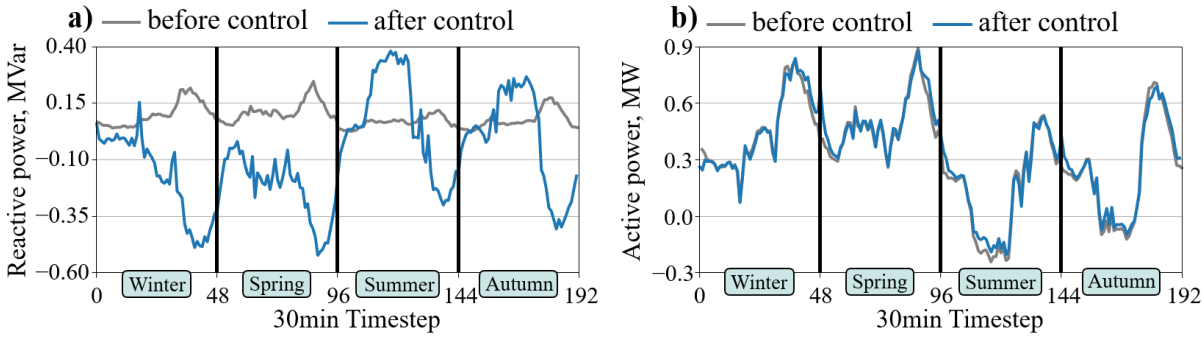


Figure 5.7. Reactive (a) and active(b) power profiles for four representative days before and after voltage control by TD3PG

Thus, the best apparent power control does not correspond to the best voltage control due to parameters of the test grid, as well as due to location and capacities of flexibilities and the load/generation profiles. In such a case, a trade-off between these two objectives is necessary, depending on the specific issues of the DSO and the requirement (plus associated penalties) proposed by the TSO. This trade-off can be adjusted by changing the coefficients of the reward function. In this regard, it is interesting to study the behavior of PPO and TD3PG algorithms for different rewards, as well as their limits on executing two different objectives simultaneously (Pareto front), which is done in the next section.

Table 5.3

EPI RESULTS FOR DIFFERENT CONTROLLERS

	Without control	Best optimization result for the given objective	TD3PG for power control	TD3PG for voltage control
EPI, MVAh	13.8	0.12	0.13	19
VPI, %	18.7	1.29	14.3	2.38

5.4 EPI/VPI trade-off

5.4.1 PPO and TD3PG VPI/EPI

To obtain the VPI/EPI trade-off limits, the training with a total of 36 couples of α and τ values is executed for each algorithm. 36 couples are selected to perform a grid search (an exhaustive search with a given step) for a chosen range of both coefficients. The values of α and τ range from 0 to the values corresponding to the best VPI (Table 3.3) and EPI (Table 5.1). In this way, the grid search is conducted from the best values of EPI and VPI to their trade-off. The best values of α and τ are found by dynamic tuning, i.e., alternately adjusting both

coefficients to the direction of lower EPI or (separately) VPI until local optima. For each pair, four training are done to mitigate the stochasticity of the training process. However, each training does not correspond to one VPI/EPI pair, but a set of EPI and VPI values over the training period. An example of one particular TD3PG training with EPI and VPI computed every 50 epochs is shown in Figure 5.8.

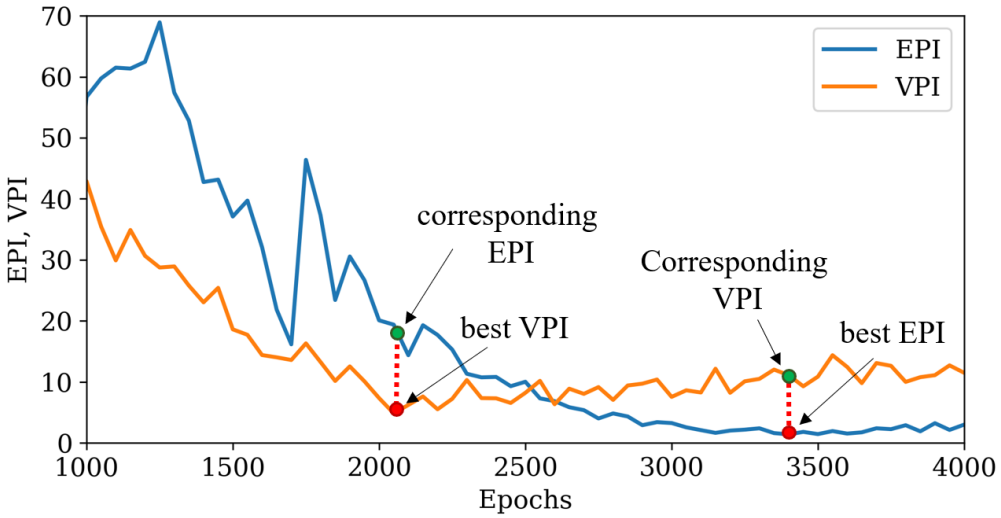


Figure 5.8. EPI/VPI results for four test weeks after every 50 epochs of TD3PG training

As for the sensitivity analysis of the voltage controllers introduced in section 3.4.2.2, the EPI and VPI curves in Figure 5.8 are obtained by stopping the training every 50 epochs, running the algorithm (trained up to that point) over the four test weeks, and getting the corresponding EPI and VPI values, before resuming the training. In this way, the curves represent the dynamic of the algorithm efficiency for the whole test year (because the results for four representative weeks are similar to the results for the whole year - Table 3.1) and not for the current day of the training year. Due to the reward function (5.3), which tries to optimize all three of its components, the ratio between EPI and VPI in Figure 5.8 constantly changes throughout training. VPI reaches 4.9% at 2050 epochs (red point in Figure 5.8) which corresponds to an EPI of 19.4 MVAh (green point). The EPI then decreases to 1.3 MVAh at 3400 epochs, but the corresponding VPI is already 11.3%.

The best obtained VPI and EPI results for all 36 pairs of α and τ of the TD3PG algorithm (except α =0 for VPI and τ=0 for EPI, because it is not relevant) are shown in Figure 5.9 (a) and Figure 5.9 (b) respectively. The VPI results present the expected behavior of the algorithm – VPI decreases with a growth of α and reduction of τ, which increases the importance of voltage regulation. The EPI values show less intuitive results. They decrease with the growth of τ, but the best value corresponds to α = 3. Thus, partial resolution of voltage violations (but not the maximum possible) helps the algorithm to manage the power exchange more easily.

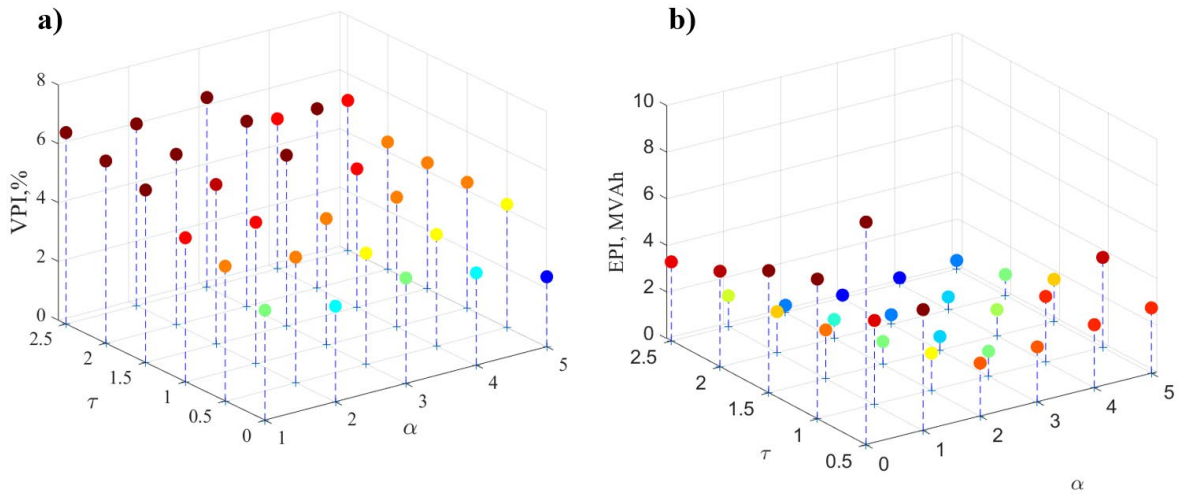


Figure 5.9. Best VPI (a) and EPI(b) results after TD3PG regulation for different values of α and τ

5.4.2 VPI/EPI Pareto front

To select the best pairs that will form the EPI/VPI Pareto front, the results of training with all 36 pairs (one training per pair) for each RL algorithm were uploaded into a specially prepared selection algorithm, the principle of which is detailed in Appendix 3. This is an iterative algorithm, which searches for the best (the lowest) VPI for all EPI/VPI pairs where EPI is lower than the fixed value EPI_{lim}. After finding the pair with the lowest VPI, it reduces EPI_{lim} to the value of EPI from that pair and repeats the process. This way, among the EPI/VPI pairs for PPO and TD3PG, non-dominated points are sorted to finally plot the Pareto front displayed in Figure 5.10 (a). To confirm that the shapes of the obtained curves are typical for the considered algorithms and their chosen parameters, and are not the result of training stochasticity, the second EPI/VPI limit curves are traced (Figure 5.10 (b)). For this, the results of the new (second) training for the same 36 pairs were used to get new non-dominated points. As can be seen, the curves for the second limit (Figure 5.10 (b)) demonstrate the same relations as for the first limit. Therefore, the observations given below can be considered reliable regardless of the stochasticity of the algorithms.

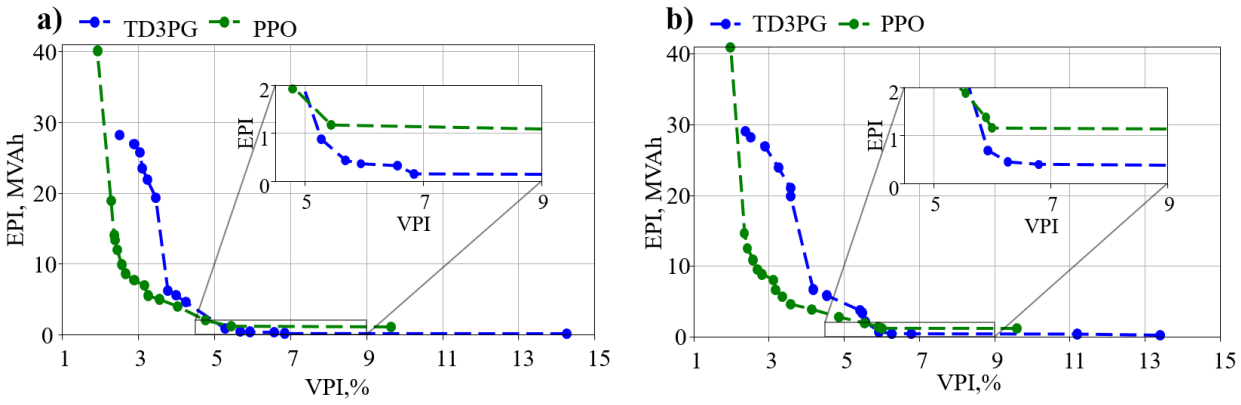


Figure 5.10. EPI/VPI trade-off curves from 30 pairs of α and τ: a) for the first training b) for the second training

We see that each of the algorithms is better suited for one of the objectives, and their trade-off curves intersect at EPI=1.6 MVAh and VPI=5.1%. TD3PG efficiency is noticeably inferior to PPO in maintaining voltage with VPI<4%, mainly due to the deadly triad problem introduced in Section 3.4. However, TD3PG greatly outperforms PPO in power management, reaching 0.16 MVAh at 6.85% of VPI, while PPO has only 1.15 MVAh for the same VPI.

The PPO performs worse for this objective. The reward contains three objectives and two of them, voltage and power control, cannot be fulfilled at the same time. As a result, the PPO has poorer efficiency, because the training is performed only on the data of the last day (48 interactions with 30-min timestep). This may be not enough to correctly converge to the global optimum because all old iterations are immediately forgotten. TD3PG, on the contrary, saves data for the entire training, which allows it to achieve better generalization. To illustrate this statement, all three reward components (r^1_t, r^2_t and r^3_t) are traced for four test weeks of the trained TD3PG controller (Figure 5.11 (a)) and the trained PPO controller (Figure 5.11 (b)). The algorithms with the best EPI are selected for this test (with EPI of 0.16 and 1.15 MVAh respectively). PPO performs well in the second half of the year with a low negative reward for all three components (days 14-28). Due to the low negative reward for the voltage regulation, its total negative reward for this period is smaller than the corresponding reward for TD3PG and reaches r_t = -0.32 per timestep compared to r_t = -0.63 for TD3PG. However, for the first half of the year, PPO has a low negative reward only for r^1_t (voltage component), but sometimes fails to limit the power exchanges. By comparing the reward with the apparent power curves in Figure 5.2, it can be noted that the component r^3_t (power component) increases dramatically when the power exchange is above 0.9 MVA. This is because the RL algorithm does not often encounter such a situation during the training (with such high apparent power exchange). As a result, during winter, batteries are depleted before a peak of power is completely smoothed, and energy is exchanged beyond the set limit. The TD3PG copes with this problem because its big replay buffer allows it to reproduce the situation with power above 0.9 MVA much more often, thereby allowing it to train a policy that successfully manages such power flows. Thus, TD3PG is more efficient in the first half of the year.

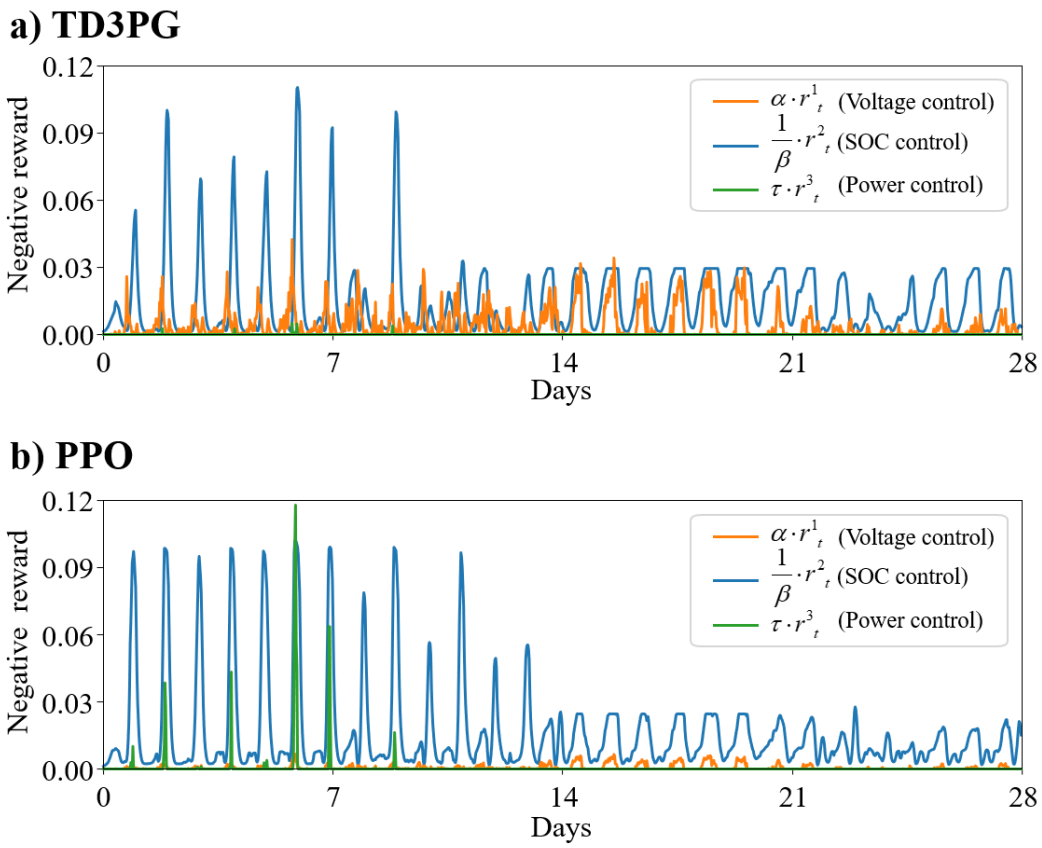


Figure 5.11. Reward components over 4 test weeks for: a) TD3PG b) PPO

5.5 Power control without voltage measurements

The proposed RL algorithm for power control, in addition to aggregated active power P_t^s and reactive power Q_t^s, also uses voltage values at each node as additional data for the controlled distribution grid. However, in low voltage grids, the voltage values of the nodes are often not available. In this regard, it is interesting to investigate the efficiency of an algorithm to perform power regulation at the DSO-TSO interface with only information about power consumption at the substation level (5.8). Such an approach would be not possible for voltage regulation purposes, thus it is concentrated only on power regulation

$$s_t = \left[\mathbf{P_t^f}, \mathbf{Q_t^f}, \mathbf{SOC_t^f}, t, P_t^s, Q_t^s \right] \qquad \textbf{(5.8)}$$

The reward, in this case, consists of only two components because voltage control (5.9) cannot be implemented as there is no feedback on the voltage measurements.

$$r_t = \frac{1}{\beta} \cdot r^2_t + \tau \cdot r^3_t \qquad (5.9)$$

Apparent power flow before and after control by the pre-trained TD3PG controller (as the most efficient for this objective) over four representative days is shown in Figure 5.12 and compared with the power regulation with a controller that embeds voltage measurements (EPI=0.13 MVAh). Regulated power is below the set limit most of the time, except for four points – two at the end of a winter day, one at the beginning, and one at the end of a spring day's evening.

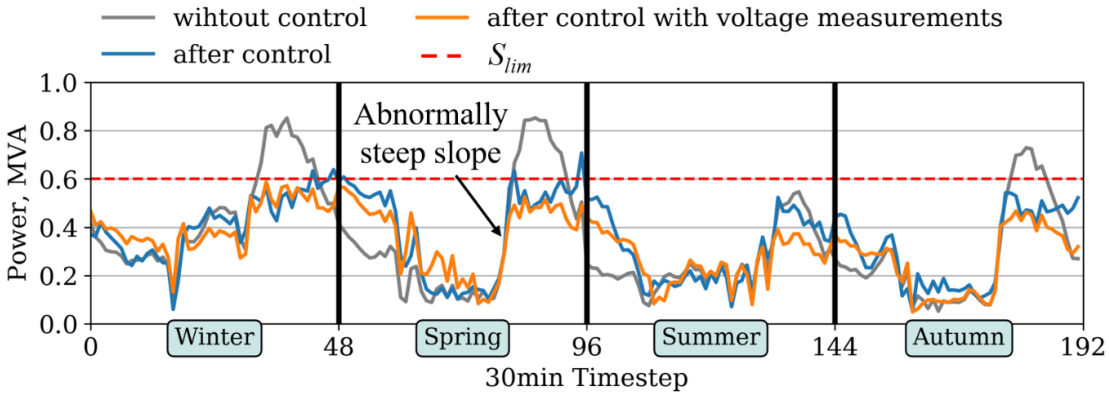


Figure 5.12. Apparent power profile for four representative days before and after power control by TD3PG without voltage information

The lack of any knowledge about individual consumption (including voltage measurements that help to estimate consumption in nodes) makes it difficult to estimate how much an increase in the consumption of each battery increases grid losses. This explains the minor power flow violations on a winter evening for instance. The exceeding of power limits in the spring can be explained mainly by the uncertainty of the load/generation profile for the next time step. The slope of the power increase is abnormally steep at the beginning of the evening, changing from 0.3 MVA to 0.65 MVA in 30 minutes (without flexibility). The RL algorithm does not expect such rapid growth, as a result of which the power flow exceeds the limit for one timestep. However, after one timestep, the RL algorithm receives information (state s_t) about the insufficiency of its flexibilities control to mitigate excess consumption and the TD3PG controller adjusts its actions accordingly. Similarly, at the end of a spring day, after a typical decrease, the load increases again for a short period, which is the opposite of the RL algorithm expectation. This results in an unexpectedly high-power flow for this timestep.

The results of reactive and active power regulation at the substation level for four representative days of the year before and after control based on aggregated power are shown in Figure 5.13 (a) and Figure 5.13 (b), respectively. The reactive power deviation magnitude is noticeably higher than for the control with known voltages' values (Figure 5.4 (a)). Active power management also lacks an estimate of the power distribution (using node voltage) to improve its control.

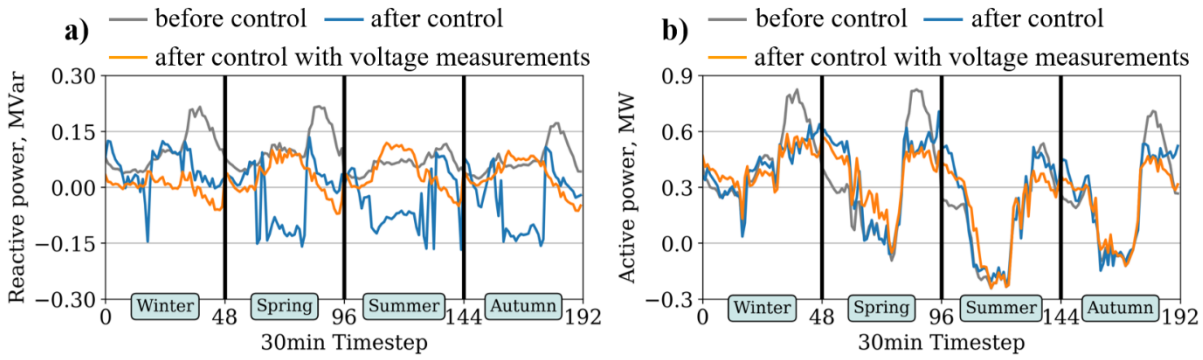


Figure 5.13. Reactive (a) and active(b) power profiles for four representative days before and after power control by TD3PG without voltage information

The EPI over four representative weeks after RL regulation is 1.06 MVAh, i.e., TD3PG reduces the power exchange beyond the limit by a factor of 13. Moreover, the RL algorithm does not have any information about the topology of the grid, its parameters, the number of loads, their voltages or the consumption. The grid represents for the TD3PG controller a black box, which receives the algorithm's flexibility control as input and outputs the aggregated power flow without any additional information (e.g., voltage values in this case). In addition, the aggregated active and reactive power flow depends not only on the flexibilities control but also on the load consumption and PV generation at the next time step. This introduces additional uncertainty into the control problem, which, as already mentioned, the proposed controller mostly successfully copes with. Thus, RL algorithms can be an essential asset for the low voltage and medium voltage grid power control within such a context of lacking information.

5.6 Dynamic power limit

The power limits set for DSO by TSO may depend on forecast consumption, generation availability, and outage of equipment for maintenance. Thus, the power limit is not necessarily fixed as in the previous tests and may change over time. For example, the DSO of the French distribution company Enedis changes its limits typically every 12-24 hours (one per day or different limits for day and night). Thus, the investigation of algorithm performance for such dynamic limits is necessary. The new state and reward function of the TD3PG algorithm are prepared for the test.

To let the algorithm know the required apparent power limit for each time step $S_{\text{lim\_}t}$, its value is added to the state vector (5.10):

$$s_t = \left[\mathbf{V_t^n}, \mathbf{P_t^f}, \mathbf{Q_t^f}, \mathbf{SOC_t^f}, t, P_t^s, Q_t^s, S_{\text{lim\_}t} \right] \qquad \textit{(5.10)}$$

To simplify the test, the dynamic limit $S_{\text{lim\_}t}$ can take one of two values – 0.55 or 0.7 MVA. The value changes every specified period T_{period}. For the training, the period between limit changes T_{period} was fixed to 24 hours. However, three periods, of 12, 24, and 48 hours, are

examined for the tests of the pre-trained (on T_{period} = 24h) RL algorithm. The apparent power profiles before and after the TD3PG control for T_{period} = 12 and T_{period} = 48 hours are shown in Figure 5.14 and for 24 hours in Figure 5.15.

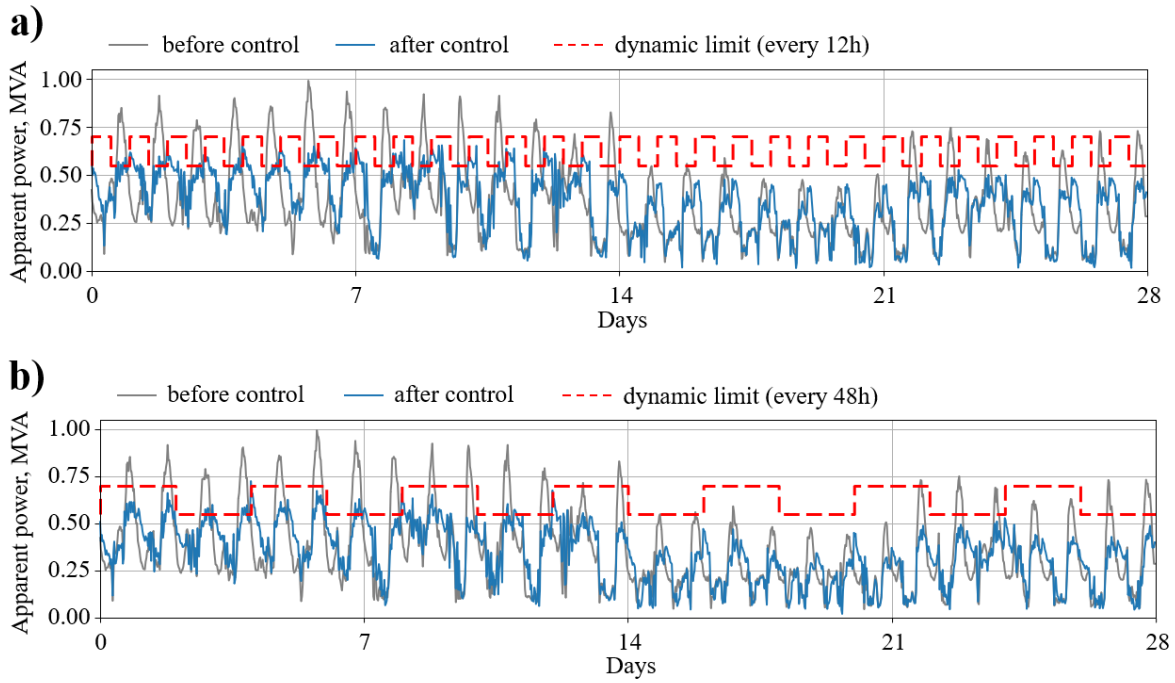


Figure 5.14. Apparent power flow through substation for four representative weeks with a dynamic limit of: a) 12 hours b) 48 hours

The zoom of Figure 5.15 allows better seeing of how the algorithm tries to compensate for the excessive consumption when the limit is low and to recharge the batteries when the limit is high or the consumption is low.

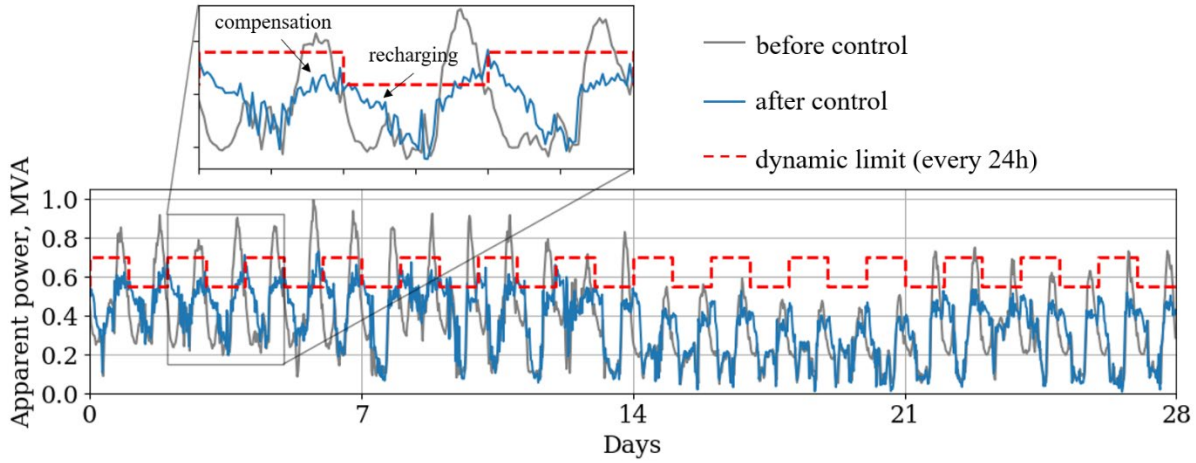


Figure 5.15. Apparent power flow through substation for four representative weeks with a dynamic limit of 24 hours

The EPIs of the four representative weeks for the three cases are presented in Table 5.4. According to the results, the TD3PG-based controller effectively regulates the power at the

substation level, even if it was trained during a different period T_{period}. This allows using the same pre-trained algorithm for different cases which is important for real-life applications.

Table 5.4

EPI RESULTS FOR DIFFERENT LIMITS

Case	Type	EPI, MVAh
Case 1 – new limit every 24h	Without control	12.71
	TD3PG control	0.56
Case 2 – new limit every 48h	Without control	12.15
	TD3PG control	0.41
Case 3 – new limit every 12h	Without control	18.8
	TD3PG control	0.65

5.7 Active power injection limit

One of the most common power limits for the DSO is the constraint on the active power injection to the transmission grid (i.e., reverse flow or back feed power). Moreover, the TD3PG-based controller, according to Figure 5.4 (b), did not limit the active power injection for apparent power control due to the lower active power injection magnitude compared to the power consumption in the simulated system. This explains the interest to test the performance of the RL algorithm to prevent active power injection.

For this situation, given that the power injection into the upstream transmission grid corresponds to negative values, the component of the reward $r^3{}_t$ is modified (5.11):

$$r^3{}_t = -\min(0, P_t^s)^2 \tag{5.11}$$

The metric EPI_P for such an objective is calculated according to (5.12), as the sum of the energy injected into the transmission grid (MWh) over four representative weeks. Without flexibility control, the value is 12.31 MWh.

$$EPI_P = -\sum_{t=1}^{T=1344} \min(0, P_t^s) \cdot dt \tag{5.12}$$

The result of active power regulation by the TD3PG-based controller for four representative days of the year is presented in Figure 5.16. The resulting EPI after the TD3PG control is 0. However, the active power is completely shifted by about 0.25 MW. In other words, the controller increases the total consumption by 0.25 MW for every timestep. The controller does not have access to load control to regulate active power, but only to batteries that cannot continuously charge due to their SOC limits.

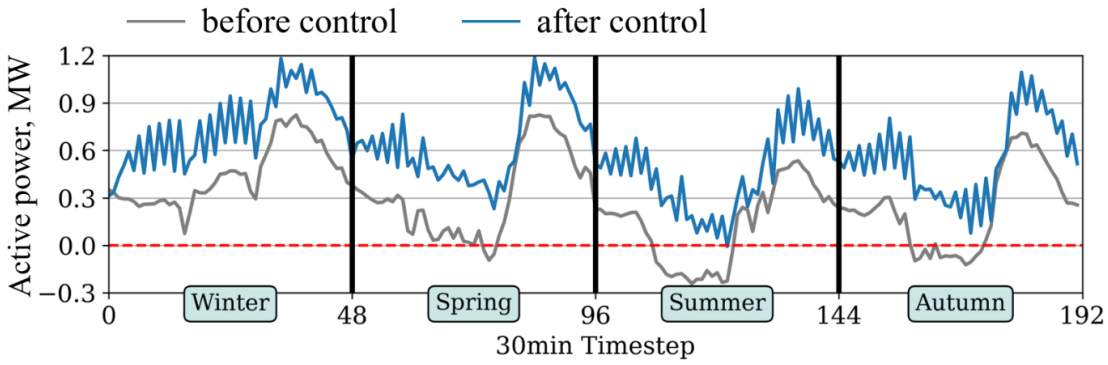


Figure 5.16. Active power flow in the substation for four representative days of the year

To explain the results for active power, the SOC profiles of all four batteries over the same four representative days are shown in Figure 5.17. As it can be seen from the SOC and active power profiles, the RL-based controller uses battery losses during charging and discharging (efficiency of the cycle is 0.81) to create additional consumption throughout the entire operating process. Battery 4 works in an antiphase with batteries 1 and 3, thus there is always at least one battery that is discharging and at least one that is charging at the same time. Thereby, due to the losses of the batteries, in total they always consume power, and resulting active power consumption profile in Figure 5.16 is always positive (i.e., without power injection).

This highlights the difference between the ways of solving a problem by humans and by artificial intelligence. Instead of trying to account for load and generation uncertainty and adapting its flexibility control to each power profile point, the RL algorithm found it easier to take the largest value of active power injection and prepare a policy that constantly provides additional consumption of the same value.

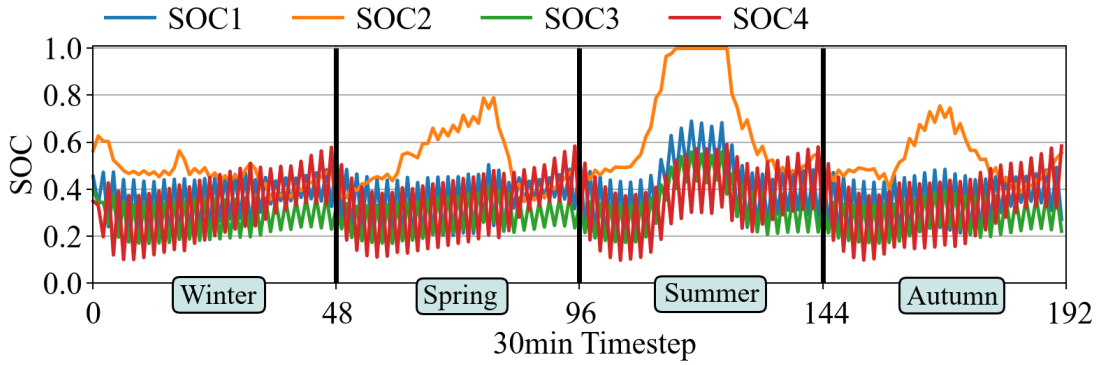


Figure 5.17. SOC profiles of the batteries for four representative days

However, such additional useless consumption is a completely undesirable solution that leads to equipment overloads, additional generation, dangerous storage power ripples (which lead to faster aging of the batteries), and CO_2 emissions. Thus, an additional upper limit of 0.8 MW is added for the reward function to prevent such behavior (5.13). This way, all consumptions exceeding 0.8 MW are also penalized. Thus, the previous policy will no longer be optimal, because it increases power consumption to 1.2 MW during peaks. For this reason,

the TD3PG algorithm must find a way to avoid power injection without increasing the peaks of energy consumption.

$$r^3_t = -\min(0, P_t^s)^2 - \max(0, (P_t^s - 0.8))^2 \qquad (5.13)$$

The resulting 4-day power profile after regulation using the TD3PG algorithm, which was trained with the updated reward function, is presented in Figure 5.18. In this case, EPI_P is equal to 0.11 MWh - the algorithm naturally limits active power injection without exceeding the power consumption peaks. Thus, due to the second limit (0.8 MW), the algorithm no longer uses battery fluctuations and does not create additional useless consumption.

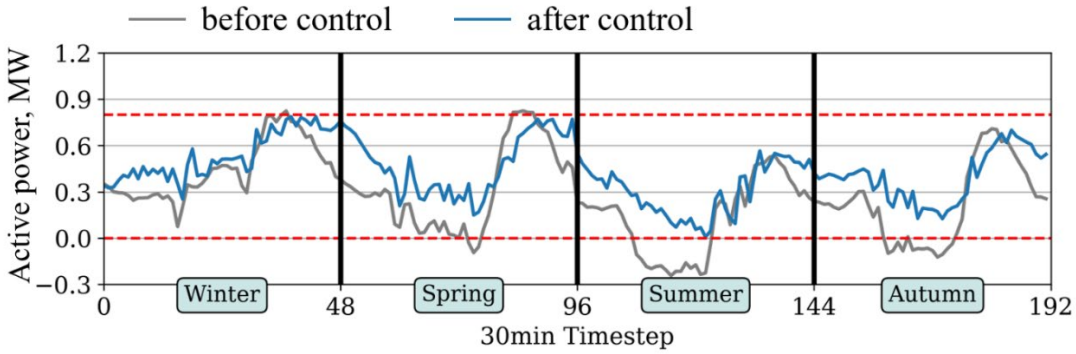


Figure 5.18. Active power profile for four representative days before and after TD3PG control

5.8 Control results for different training set sizes

All results presented in chapters 3-5 were obtained by algorithms trained on 365-day profiles. However, it is interesting to estimate the minimum length of the profiles (the size of the dataset) which allows for obtaining a similar performance. Note that the decrease in the size of a dataset does not directly impact the time of the training but reduces the duration of the measurements required to prepare this dataset. However, algorithms trained on small datasets typically have a poorer generalization and are more biased to these specific days. Thus, it is preferable to use a larger data set, if available. The sensitivity of the EPI and VPI results to the size of the training dataset for both RL algorithms is investigated in this subsection. The VPI results of both RL algorithms (which use voltage measurements and aggregated power measurements as an input) for different profile lengths (in days) are shown in Figure 5.19 (a) and the results of EPI are shown in Figure 5.19 (b). The results show similarities in the required number of epochs for training with different data set sizes. For instance, the best number of epochs to get the lowest EPI for a 365-day profile dataset is 2300, and for a 1-day profile dataset is 2400. However, in the first case, the algorithm used on average the profile of each day 6 times, and in the second case, the algorithm used the same day profile 2400 times over the training.

The metric results of the best algorithms for each objective (PPO for voltage control and TD3PG for power control) vary only slightly for a dataset with a size between 1 and 365 days. Thus, these algorithms have an excellent generalization ability and, trained on only one daily profile of consumption and generation (January 1st), they can successfully implement control close to the optimal for the whole year, even if the daily profiles during the year differ significantly from the selected day (as shown in the PV and load profiles in Figure 3.6). However, the performance decreases drastically when the length of the profiles is reduced to less than a day. It can be concluded that the RL algorithms need data on the evening peak of consumption to be correctly trained.

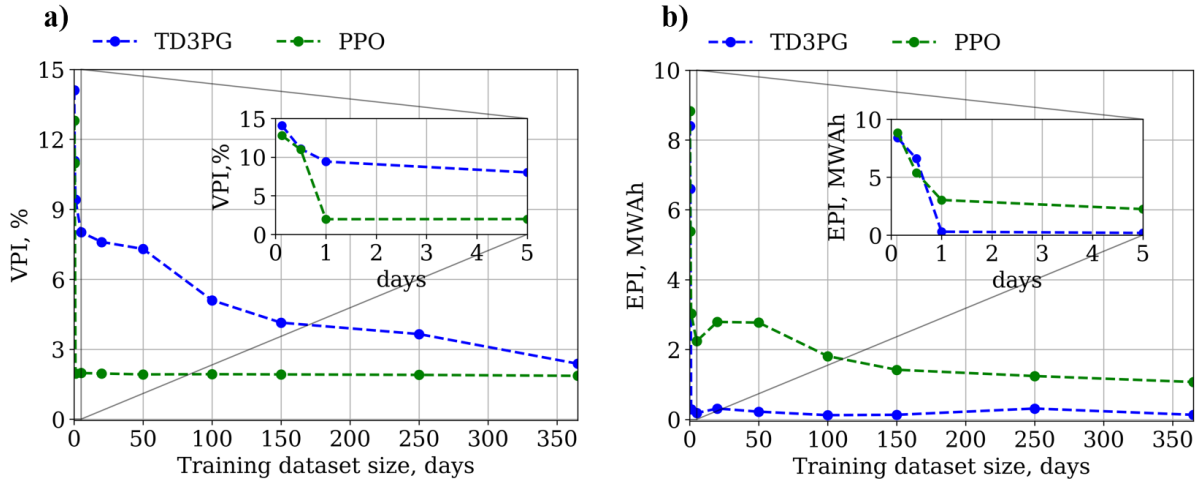


Figure 5.19. TD3PG and PPO metric results from different dataset sizes for: a) VPI b) EPI

The results of the less performant algorithms for each objective (TD3PG for voltage control and PPO for power control) show that they are more sensitive to changes in profile length. The VPI of TD3PG increases from 2.38% to 9.42%, and the EPI of PPO increases from 1.07 to 3.03 MVAh for 1-day profiles compared to 365-day profiles. However, they show results similar to the best algorithms, when they are trained on less than 1-day profiles, intersecting with the best algorithms in the region of 12-15 hours of dataset size (with VPI of 11% and EPI of 8 MVAh).

Similar results can be observed for the voltage controllers that use only voltage measurements as an input without aggregated power input (i.e., controllers presented in Chapter 3). In Figure 5.20 PPO shows a much greater generalization ability – its VPI almost does not change with decreasing of dataset size from 365 days (VPI = 1.85%) to only a single day (VPI=1.96%), whereas the VPI for TD3PG voltage control increases from 2.8% to 8.6%. Thus, the PPO algorithm trained only on a one-day profile (e.g., load/generation on January 1st) can allow training of a voltage controller that is relevant for the whole year with almost the same efficiency as in a case with the one-year profile.

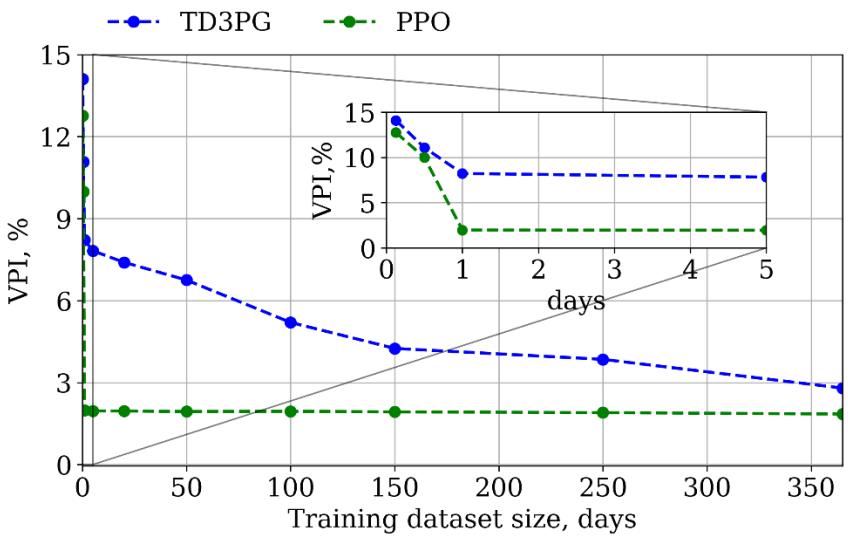


Figure 5.20. TD3PG and PPO VPI results for different dataset sizes

Thus, the additional importance of choosing the right RL algorithm for each objective (regardless of the grid parameters) can be emphasized in the sense that these algorithms can be effectively trained sometimes on one-day measurement data. However, it is often preferable to use a larger dataset, if available, allowing for a better generalization, which is important for profiles that have abnormal consumption days. As future work, it may be interesting to further investigate the performance of algorithms with small datasets for different periods and consumption profiles. For instance, for different seasons, for days with and without high solar insolation, for profiles with commercial consumers, etc.

5.9 Scalability test

The power controller is tested on a larger distribution grid (55 buses) introduced in Section 4.5 (Figure 4.10). Its apparent power flow through the substation over four representative weeks of the test year is shown in Figure 5.21. The power limit is fixed at S_{lim}=9 MVA and then its corresponding EPI without control is 265.39 MVAh. The total number of epochs was set to 15000.

The results for power regulation on the 55-bus distribution grid by TD3PG- and PPO-based controllers as well as by the reference optimization-based controller for four representative weeks of the test year are shown in Table 5.5.

Table 5.5

***EPI* RESULTS FOR THE 55-BUS GRID BY DIFFERENT CONTROLLERS**

	Without control	Optimization	TD3PG	PPO
EPI, MVAh	265.39	13.7	18.09	29.03
Number of epochs for the best EPI	-	-	1950	1400

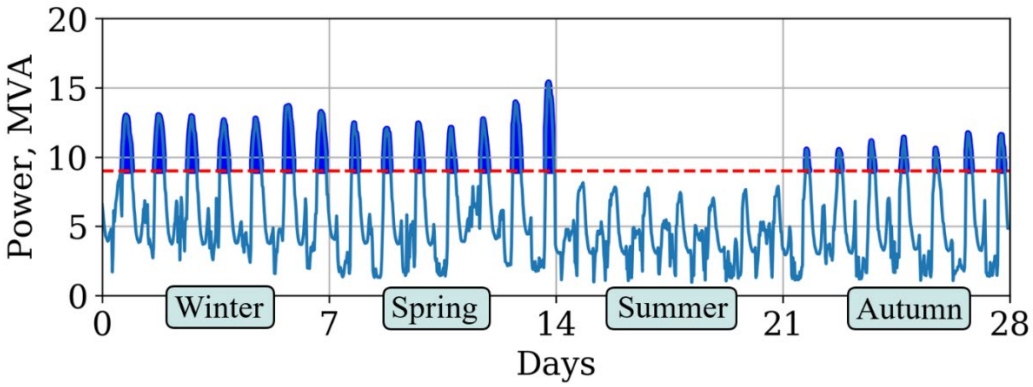


Figure 5.21. Apparent power at the substation for four representative weeks of 55 nodes grid

Similar to the results for the 11-bus grid in section 5.3.1, the TD3PG algorithm regulates power better than PPO and obtains results close to the ones returned by the optimization-based controller. The TD3PG-based controller obtains the EPI of 18.09 MVAh, i.e., it reduces energy exchanges beyond the limit by almost a factor of 15 compared to energy exchanges without flexibility control. Moreover, both RL algorithms need even fewer training epochs than for the 11-bus grid to get their best EPI. This is probably due to the simpler control policy required for near-optimal regulation compared to a small grid. Thus, RL algorithms need less training to construct such a policy. However, it is worth mentioning that simulation of the grid (performed in the simulation tool Pandapower) is about two times longer than for the 11-bus grid. Operation time is the same – RL-based controller takes a fraction of a second to output 1-step control. The results of apparent power regulation by TD3PG-based controller and by the optimization-based controller over four representative days are shown in Figure 5.22.

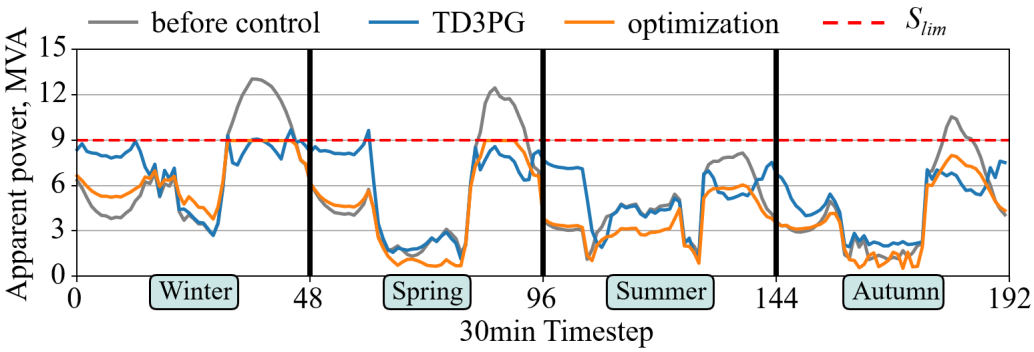


Figure 5.22. Apparent power profile for four representative days before and after TD3PG and optimization control

TD3PG algorithm flattens a little bit too much the peaks, reducing the power beyond the limits to the values of 7-9 MVA. The optimization algorithm (same as the one presented in Chapter 3), for the same profile, exactly matches the limit of 9 MVA, because it uses an ideal consumption forecast and an accurate grid model (not realistic in practice), thus it does not need to consider uncertainties. For the same reason, the regulation of batteries by a TD3PG-based controller leads to higher consumption (compared to optimization) during low load hours. This allows batteries to be fully recharged and used more actively later to safely smooth out peaks. However, the difference between the results of the controllers is only 4.39 MVAh, i.e., only 1.65% of the total energy exchanged beyond the limit without flexibility control (265.39 MVAh).

5.10 Conclusion

The use of reinforcement learning algorithms for the regulation of power exchange between distribution and transmission grids was considered in this chapter. The two RL algorithms, TD3PG and PPO, as well as an optimization-based algorithm were compared. TD3PG gets better results compared to PPO due to its off-policy approach, which allows a large replay buffer to be saved for better generalization. This RL algorithm also shows results close to those of the optimization-based algorithm, reducing the energy exchange beyond the limit from 13.8 MVAh to 0.13 MVAh, while the optimization-based algorithm reaches 0.12 MVAh. However, the RL algorithm, unlike the optimization approach, does not need information about the grid and copes with the uncertainty of loads and generations, whereas optimization uses a perfect forecast for the whole year.

The effectiveness of the RL algorithms has been tested for various cases– apparent power control with a fixed limit, apparent power control with a dynamic limit, and active power control with only injection and both injection and consumption limits. The results show good efficiency in the case of properly set reward functions. Moreover, the ability of the algorithm to perform power regulation based solely on aggregated power values at the substation without any additional information about the grid was tested. Such an algorithm will be especially useful

for low voltage grid regulation, where the DSO has little information due to a lack of measurements or privacy by design principle.

The simultaneous execution of two control tasks (voltage regulation and power regulation) for both RL algorithms was also investigated. It is concluded that for the considered test system and power profiles, the two objectives are somewhat antagonistic. Thus, the Pareto fronts for both algorithms were plotted by changing the parameters of the reward function. PPO shows a better trade-off with the dominance of the voltage regulation objective, while TD3PG shows better results with the dominance of the power regulation objective. Both fronts intersect at EPI=1.6 MVAh and VPI=5.1%. PPO and TD3PG also demonstrate very good generalization ability and low dependence on the size of the training set, showing excellent results with a data size of at least one day.

Finally, the scalability of power control by reinforcement learning was verified on the 55-node distribution grid. TD3PG-based controller reduced the power exchange beyond the limit by 15 times (from 265.39 MVAh to 18.09 MVAh), which is only 1.65% worse than the optimization-based control. Moreover, the time of RL training and its operation time did not increase significantly compared to the 11-bus grid regulation. Thus, the use of RL-based controllers may be relevant even for large electrical grids.

6 General conclusions and perspectives

This thesis focuses on the development of AI-based tools that help DSOs to cope with the increasing complexity in distribution grid management. In the first part, the current challenges faced by the DSO due to the growing integration of DER were highlighted. Especially, the main tasks for the thesis investigations were identified and selected:

i) Detection of PV without a connection agreement in distribution grids;

ii) Voltage regulation in the distribution grid with the control of energy storage systems;

iii) Voltage control in the distribution under grid parameters' uncertainties;

iv) Power management of distribution grid at the primary substation level, TSO-DSO interface.

Then all main types of machine learning algorithms were introduced and briefly described. They can be divided into three main categories – unsupervised learning, supervised learning, and reinforcement learning. Existing solutions have been investigated for the above-listed tasks, and their advantages and drawbacks have been identified.

The second part of the work proposed two different algorithms to detect PV installations in a distribution grid without a connection agreement. Unlike most solutions in the literature that require specific data (such as solar radiation data for a specific location [36], data from other PV stations [34], detailed weather data [33], or high-resolution satellite imagery [35]), the proposed methods use only smart meter and temperature data. The first algorithm (approach A) is based on the classification by a convolutional neural network, and the second algorithm (approach B) combines a forecasting tool based on conventional multi-layer perceptron together with an analytical classification algorithm. The algorithm's performances were compared for different sets of hyperparameters on a distribution grid with 14 nodes. Approach A showed a lower accuracy of 76% compared to an accuracy of 100% for Approach B under the same conditions. But the accuracy of PV detection for Approach B highly depends on the installed PV capacity and the length of the considered period for the training. Thus, it is necessary to find a trade-off between the volume of available data and the expected accuracy of the solution. The highest accuracy for the consideration in the work case could be reached by considering the three sunniest months of the year. In this case, the algorithm can find PV in any bus of the modeled distribution grid if the installed PV capacity is at least 6.2% of the maximum consumption of the same bus. Thus, it can be stated that Approach B is a good basis for PV detection in distribution grids, offering many opportunities to be further expanded.

Future works here can be concentrated on the disaggregation of the net load (at the user meter level) into its local generation component and local consumption contribution. It may be interesting to determine how the proposed identification method works in the presence of PV systems with storage capability, for operational and long-term planning issues. Regarding the data, additional consideration could be given to noise and anomalies, for instance dealing with less clean measurements.

In the third part, the development of a controller based on reinforcement learning algorithms (TD3PG and PPO) to regulate the voltage in a distribution grid was presented. To regulate bus voltages, RL algorithms control some flexibilities (in the form of batteries and inverters, taking into account the SOC of the batteries as well as their both active and reactive power). Compared to most of the solutions in the literature (such as [46]-[51]), the proposed algorithms do not need any consumption/generation data and rely on grid voltage measurements, thus respecting the privacy of the data principle with no access to power profiles at the users' level. Moreover, they do not need an external forecasting tool for the bus power demand/generation. Instead, the controller computes the actions for the next timestep, based on the voltage values and control setpoints from the previous timestep, thus taking implicitly into account the uncertainty of consumers' power profiles. A sensitivity analysis of the algorithms' performance to their main hyperparameters was carried out, which included over 1500 runs. Each set of hyperparameters was tested 3-5 times to mitigate the high stochasticity of the training results for RL algorithms. A sensitivity analysis showed that PPO is less sensitive to the choice of hyperparameters than TD3PG. A distribution grid with 11 buses was considered as a test system. The PPO-trained controller reduced voltage violations from 19.5% to 1.85% of the time, whereas a regulation trained with TD3PG had almost 2.8% of voltage violations over the test period. However, TD3PG needed almost six times fewer epochs (i.e., less iteration and thus computational time) to be trained. The higher accuracy of PPO was also confirmed by an investigation of the algorithms' policies, which showed that PPO uses flexibilities more efficiently and returns expected actions (i.e., storage charge/discharge) for the given voltage values.

The fourth chapter expanded the objective of voltage control, by investigating the performance of the RL-based controller under load and impedance uncertainties. Especially, a traditional optimization-based controller was used as a reference model to compute the theoretical best performances in cases where there are no uncertainties. A comparison of voltage control results showed that the RL algorithms are notably more robust to line impedance uncertainties starting from 5% of deviation from nominal values. Also, a two-stage training was proposed to improve the control results. It consists of offline training on a model of the grid followed by online learning with regular updates of the controller parameters, once connected to the actual systems. Such training was tested for different scenarios (abrupt impedance difference and smooth impedance drifting over ten years) and allowed voltage violations to be decreased by 55% compared to a controller trained offline only. The scalability of the voltage controller was also tested on a distribution grid five times bigger than the initial one, showing that the RL algorithms still effectively control the voltage even for a larger number of buses.

Future works for voltage control solutions can focus on the investigation of other regulating means (capacitors, tap changers, etc.) and investigation of scalability limits, i.e., the maximum number of both buses and regulated flexibilities, which still allows reaching the desired accuracy. Implementation of multiagent control can be also considered for the simultaneous regulation of even larger distribution grids.

The last part of the work can be also considered as an extension of the voltage control task while adding an objective related to the power exchanged between distribution and transmission grids. In this case, TD3PG-based controller displayed better results compared to PPO, reducing the apparent power exchange beyond the fixed limit from 13.8 MVAh to only 0.13 MVAh over the test period. PPO got 1.07 MVAh and the traditional optimization-based algorithm reached 0.12 MVAh. Algorithms were tested for various power exchange objectives. Except for the objective to limit apparent power below the fixed value, algorithms have been tested for the objective to control apparent power with a variable limit and for the objective to avoid active power injection into the transmission grid as well as to control both injection and consumption limits of active power. The algorithms showed good efficiency for all these cases. It was also concluded that for the considered 11-bus system, the voltage and power control objectives could not be completely fulfilled simultaneously. Thus, Pareto fronts for both objectives and two algorithms were built while tuning the parameters of the reward function. The PPO-based controller showed better results with the dominance of the voltage regulation objective over the power limit, while TD3PG showed better results with the dominance of the power regulation objective. Here again, the scalability of the algorithms was tested on a 55-bus distribution grid, and the RL algorithm which does not use measurements of power in the nodes, but only aggregated power at the substation level, showed close to optimal power management results.

Future perspectives on this part may include investigations on the RL algorithm performances with dynamic power limits that may vary in real-time. Such operating conditions may become relevant for grids with a predominance of variable generation based on renewable energy sources. It is also interesting to investigate various reward functions and their impact on algorithms' performances.

To conclude, machine learning techniques can contribute to both, detection/classification and regulation objectives of DSO. RL algorithms do not require complete knowledge of the system but can understand it solely based on the results of interactions with it. This allows them to be used when there is a lack of measurements, which advantageously distinguishes RL-based controllers from traditional optimization-based ones. However, RL algorithms are black boxes, and the results of their training cannot be 100% predictable. Therefore, the best results require the correct choice of hyperparameters and reward and representative training data. In this case, RL-based controllers can successfully replace some of the existing tools and bring additional functionality to distribution grid management.

Résumé français étendu

L'intégration croissante des sources d'énergie renouvelables distribuées (RES), telles que le photovoltaïque (PV) et les éoliennes, ainsi que de nouveaux types de consommateurs, tels que les véhicules électriques (EV), ont un impact important sur le fonctionnement et la planification des réseaux de distribution. L'un des principaux acteurs concernés par ces changements est les Chargés de conduite (CCO). Les CCOs sont les gestionnaires des réseaux de distribution fonctionnant à basse (BT) et moyenne tension (MT) (en dessous de 50 kV en France) et sont responsables de la distribution de l'énergie aux clients finaux. En France, plus de 500 DSO de la principale entreprise de services publics (Enedis) fournissent de tels services dans 29 agences de distribution qui contrôlent 95% des réseaux de distribution du pays.

La nouvelle production intermittente à partir de RES et charges non conventionnelles ont un impact important sur le fonctionnement du réseau de distribution (par exemple, l'augmentation des pertes d'énergie ou des violations de tension) et la rend plus difficile à contrôler. Cependant, de nouveaux moyens de régulation du réseau de distribution, appelés flexibilités, émergent également. Les flexibilités incluent, par exemple, la batterie des EV lorsqu'elle est connectée au réseau, ainsi que les systèmes de stockage classiques et/ou la capacité des clients à contrôler une partie ou la totalité de leur charge (c'est-à-dire la réponse à la demande). Les CCOs peuvent accéder à de tels moyens de régulation tout en rémunérant les fournisseurs de flexibilité pour leur disponibilité de participation. En outre, les flexibilités incluent le contrôle partiel des RES tels que les onduleurs photovoltaïques, ce qui permet une régulation de l'énergie à grande échelle dans le réseau et permet d'utiliser une part encore plus élevée de RES. Enfin, le concept de prosumers peut être défini, comme des consommateurs non passifs qui peuvent modifier leur consommation ou produire eux-mêmes de l'énergie et partager le surplus avec d'autres.

Cette thèse est alors consacrée au développement de tels outils, pour répondre aux enjeux suivants :

• La nature intermittente des RES introduit de l'incertitude et de la variabilité dans les profils de puissance et rend la gestion de la tension et de l'alimentation plus complexe. La situation est encore pire pour les réseaux de distribution avec une part élevée de propriétaires de PV qui peuvent ne pas avoir d'accords de connexion et ne pas être surveillés au niveau centralisé.

• Des pics de consommation supplémentaires dus aux parcs de véhicules électriques et une diminution brutale possible de la production d'énergie distribuée (par exemple, pour le

photovoltaïque en raison des nuages) nécessitent une plus grande réactivité de la part du CCO pour maintenir la tension dans les limites spécifiées. De nouveaux actifs peuvent fournir de nouveaux degrés de liberté qui compléteront avantageusement les leviers traditionnels, tels que les changeurs de prises de charge de transformateurs ou de batteries de condensateurs, qui peuvent ne pas réagir assez rapidement et vieilliraient rapidement avec une utilisation aussi fréquente.

• Les contrôles CCO traditionnels sont basés sur un modèle du réseau, qui utilise les paramètres nominaux du réseau (c'est-à-dire paramètres des lignes) et des mesures en temps réel de la puissance et de la tension (ou leur estimation). Cependant, les mesures peuvent être partiellement indisponibles. Il en va de même pour les paramètres des lignes, en particulier dans les réseaux basse tension où les valeurs d'impédance ne sont pas parfaitement connues. De plus, ces impédances peuvent se dégrader avec le temps en raison du vieillissement qui peut affecter les performances de l'estimation et/ou du contrôle de l'état du réseau.

• L'intégration des RES au niveau de la distribution entraîne une plus grande volatilité des profils de puissance à l'interface avec le réseau de transport. La puissance à cette interface peut alors être soumise à des contraintes/limites imposées par le gestionnaire de réseau de transport (TSO). Ainsi, le CCO doit contrôler simultanément la tension et la puissance (active et réactive) dans son réseau à l'interface avec le TSO.

Traditionnellement, des systèmes experts étaient proposés pour faciliter les tâches de CCO. Un exemple de système expert est un contrôle de supervision et d'acquisition de données (SCADA), reposant sur un calcul de flux de puissance optimal (OPF) et un assistant de diagnostic et de restauration des fautes. Cependant, les systèmes experts présentent des inconvénients tels que :

1. Manque de réponses créatives dont les experts humains sont capables ;

2. Difficulté à automatiser des processus complexes ;

3. Peu ou pas de flexibilité et de capacité d'adaptation aux conditions changeantes ;

4. Difficulté à reconnaître l'absence de réponse.

De plus, tous les nouveaux leviers de flexibilité (par exemple, les batteries et la réponse à la demande), ainsi que les nouvelles variables introduites dans l'équation du bilan énergétique (par exemple, les centrales photovoltaïques, éoliennes et EV) qui affectent les profils de charge et de production, rendent le travail du CCO plus complexe. Ainsi, les capacités des systèmes experts peuvent ne pas être suffisantes pour gérer efficacement des réseaux complexes. Cela nécessite le développement de nouveaux outils qui peuvent surmonter ce défi et aider le CCO dans certaines tâches.

Au cours des dernières années, l'installation de compteurs intelligents (“Linky” par Enedis) a apporté un volume important d'informations sur le réseau BT. Avec l'augmentation de la puissance de calcul ces dernières années, cela conduit à un intérêt accru pour l'utilisation de l'intelligence artificielle (IA), ou plus précisément de l'apprentissage automatique, pour

remplacer certains des outils existants et apporter des fonctionnalités supplémentaires à la gestion des réseaux de distribution.

Cette thèse se concentre sur le développement d'outils basés sur l'IA qui aident les CCO à faire face à la complexité croissante de la gestion du réseau de distribution. Dans la première partie, les défis actuels auxquels est confronté le CCO en raison de l'intégration croissante de la RES ont été mis en évidence. En particulier, les tâches principales pour les investigations de thèse ont été identifiées et sélectionnées :

i) Détection de PV sans accord de raccordement dans les réseaux de distribution ;

ii) Régulation de la tension dans le réseau de distribution avec le contrôle des systèmes de stockage d'énergie ;

iii) Contrôle de la tension dans le réseau de distribution sous les incertitudes des paramètres du réseau ;

iv) Gestion de la puissance dans réseau de distribution au niveau du poste source, interface CCO-TSO.

Ensuite, tous les principaux types d'algorithmes d'apprentissage automatique ont été présentés et brièvement décrits. L'apprentissage automatique est un domaine d'étude de l'intelligence artificielle qui s'appuie sur des approches mathématiques et statistiques pour permettre aux programmes "d'apprendre" à partir de données, c'est-à-dire d'améliorer leurs performances dans la résolution de tâches sans être explicitement programmés pour chacune d'entre elles. Les algorithmes d'apprentissage peuvent être classés en fonction de leur mode d'apprentissage et de leur fonctionnalité en trois catégories principales : apprentissage non supervisé, apprentissage supervisé et apprentissage par renforcement. Les algorithmes de chaque catégorie peuvent être appliqués à différents problèmes. Les solutions existantes ont été étudiées pour les tâches énumérées ci-dessus, et leurs avantages et inconvénients ont été identifiés.

Les sources d'énergie renouvelables distribuées, en particulier PV, ont connu une croissance rapide au cours des deux dernières décennies. Pour cette raison, les informations sur la production PV deviennent cruciales pour les CCOs afin d'effectuer des opérations régulières telles que l'estimation de l'état, la reconfiguration et la gestion de la tension, entre autres applications. Cependant, les installations photovoltaïques appartenant à des ménages n'ont pas nécessairement un accord de connexion avec l'opérateur du système et ne sont donc pas surveillées à un niveau centralisé par défaut. Si elle n'est pas identifiée correctement, cette génération est alors « cachée » du point de vue de l'opérateur du système. Cela peut entraîner des incertitudes supplémentaires dans la mesure et la prévision de la charge nette (en réduisant la charge nette par rapport à celle attendue pendant la journée) et, en particulier, rend plus difficile l'exploitation sûre du réseau de distribution. Avec la croissance constante des capacités photovoltaïques installées, cet enjeu devient de plus en plus important. Pour cette raison, la deuxième partie du travail a proposé deux algorithmes différents pour détecter les installations PV dans un réseau de distribution sans accord de raccordement. Contrairement à la plupart des

solutions de la littérature qui nécessitent des données spécifiques (telles que des données de rayonnement solaire pour un emplacement spécifique, des données d'autres stations PV, des données météorologiques détaillées ou des images satellite à haute résolution), les méthodes proposées utilisent uniquement des compteurs et des données de température.

Le premier algorithme (approche A) est basé sur la classification par un réseau de neurones convolutifs, et le deuxième algorithme (approche B) combine un outil de prévision basé sur le perceptron multicouche classique avec un algorithme de classification analytique. Les performances des algorithmes ont été comparées pour différents ensembles d'hyperparamètres sur un réseau de distribution à 14 nœuds. L'approche A a montré une précision inférieure de 76% par rapport à une précision de 100% pour l'approche B dans les mêmes conditions. Mais la précision de la détection PV pour l'approche B dépend fortement de la capacité PV installée et de la durée de la période considérée pour la formation. Ainsi, il est nécessaire de trouver un compromis entre le volume de données disponibles et la précision attendue de la solution. La précision la plus élevée pour la prise en compte dans le cas de travail pourrait être atteinte en considérant les trois mois les plus ensoleillés de l'année. Dans ce cas, l'algorithme peut trouver du PV dans n'importe quel nœud du réseau de distribution modélisé si la capacité PV installée est d'au moins 6,2% de la consommation maximale du même nœud. Ainsi, on peut affirmer que l'approche B est une bonne base pour la détection PV dans les réseaux de distribution, offrant de nombreuses possibilités d'extension.

Les travaux futurs ici peuvent être concentrés sur la désagrégation de la charge nette (au niveau du compteur utilisateur) en sa composante de production locale et sa contribution à la consommation locale. Il peut être intéressant de déterminer comment la méthode d'identification proposée fonctionne en présence de systèmes PV avec capacité de stockage, pour des questions opérationnelles et de planification à long terme. En ce qui concerne les données, une attention supplémentaire pourrait être accordée au bruit et aux anomalies, par exemple en traitant des mesures moins propres.

L'une des principales tâches du CCO est de maintenir les niveaux de tension dans des limites spécifiées, généralement entre 0,95 et 1,05 normalisé par unité (p.u.) dans des conditions normales. Les CCO peuvent utiliser différents leviers pour réguler la tension. Par exemple, la régulation conventionnelle consiste en le contrôle d'un changeur de prises en charge (OLTC) ou de batteries de condensateurs. Ces dernières années, de nouvelles capacités de contrôle sont apparues en raison de la pénétration croissante des RES et de la gestion de la demande, appelées “flexibilités”. Ils peuvent être utilisés pour une large gamme d'opérations sur le réseau en dehors de la régulation de la tension, telles que la minimisation des pertes. Le contrôle typique de ces ressources flexibles est basé sur des approches d'optimisation/basées sur des modèles. Ces approches peuvent consister à regarder vers l'avenir et/ou à des processus de prise de décision en temps quasi réel. Cependant, ils nécessitent des valeurs de consommation des compteurs ainsi que des données de réseau (c'est-à-dire la topologie et l'impédance des lignes) pour estimer l'état futur des systèmes. Ces données sont généralement intégrées dans des stratégies de contrôle prédictif de modèle (MPC) qui reposent sur des prévisions de séries temporelles pour la puissance du nœud (c'est-à-dire la charge/génération). Cependant, les algorithmes basés sur

l'optimisation peuvent nécessiter un temps important pour calculer les contrôles, en particulier lorsque l'on considère l'atténuation des incertitudes comme une caractéristique essentielle, par exemple les optimisations stochastiques, ou lorsqu'un plus grand nombre de ressources doivent être contrôlées. De plus, les algorithmes présentés dans la littérature reposent principalement sur l'estimation de la puissance du nœud ou la prédiction à partir d'un outil externe.

Ainsi, dans la troisième partie, le développement d'un contrôleur basé sur des algorithmes d'apprentissage par renforcement (RL), TD3PG et PPO, pour réguler la tension dans un réseau de distribution a été présenté pour traiter les problèmes des contrôleurs classiques. Pour régler les tensions des nœuds, des algorithmes RL contrôlent certaines flexibilités sous forme de batteries et d'onduleurs, prenant en compte l'état de charge (SOC) des batteries ainsi que leur puissance à la fois active et réactive. Par rapport à la plupart des solutions de la littérature, les algorithmes proposés ne nécessitent aucune donnée de consommation/production et s'appuient sur des mesures de tension de réseau, respectant ainsi le principe de confidentialité des données sans accès aux profils de puissance au niveau des utilisateurs. De plus, ils n'ont pas besoin d'un outil de prévision externe pour la demande/la production d'énergie du bus. Au lieu de cela, le contrôleur calcule les actions pour le pas de temps suivant, sur la base des valeurs de tension et des valeurs de contrôle du pas de temps précédent, prenant ainsi implicitement en compte l'incertitude des profils de puissance des consommateurs. Une analyse de sensibilité des performances des algorithmes à leurs principaux hyperparamètres a été réalisée, qui comprenait plus de 1500 exécutions. Chaque ensemble d'hyperparamètres a été testé 3 à 5 fois pour atténuer la stochasticité élevée des résultats d'entraînement pour les algorithmes RL. Une analyse de sensibilité a montré que la PPO est moins sensible au choix des hyperparamètres que la TD3PG. Un réseau de distribution avec 11 nœuds a été considéré comme un système de test. Le contrôleur basé sur PPO a réduit les violations de tension de 19,5% à 1,85% du temps, alors qu'un contrôle par TD3PG a enregistré près de 2,8% de violations de tension sur la période de test. Cependant, TD3PG avait besoin de près de six fois moins d'époques (c'est-à-dire moins d'itérations et donc de temps de calcul) pour être entrainé. La plus grande précision du PPO a également été confirmée par une étude sur les politiques des algorithmes, qui a montré que le PPO utilise les flexibilités plus efficacement et renvoie les actions attendues (c'est-à-dire la charge/décharge de stockage) pour les valeurs de tension données.

Le quatrième chapitre a élargi l'objectif du contrôle de tension, en étudiant les performances du contrôleur basé sur RL sous des incertitudes de charge et d'impédance. En particulier, un contrôleur traditionnel basé sur l'optimisation a été utilisé comme modèle de référence pour calculer les meilleures performances théoriques dans les cas où il n'y a pas d'incertitudes. Une comparaison des résultats du contrôle de tension a montré que les algorithmes RL sont nettement plus robustes aux incertitudes d'impédance de ligne à partir de 5% d'écart par rapport aux valeurs nominales. Aussi, un entraînement en deux étapes a été proposée pour améliorer les résultats du contrôle. Il consiste en un entraînement hors ligne sur un modèle du réseau suivie d'un entraînement en ligne avec des mises à jour régulières des paramètres du contrôleur, une fois connecté aux systèmes réels. Un tel entraînement a été testée pour différents scénarios (différence d'impédance brusque et déviation d'impédance lente sur

dix ans) et a permis de réduire les violations de tension de 55% par rapport à un contrôleur entraîné hors ligne uniquement. L'évolutivité du contrôleur de tension a également été testée sur un réseau de distribution cinq fois plus grand que le réseau initial, montrant que les algorithmes RL contrôlent toujours efficacement la tension même pour un plus grand nombre de bus.

Les travaux futurs pour les solutions de contrôle de tension peuvent se concentrer sur l'étude d'autres moyens de régulation (condensateurs, changeurs de prises, etc.) et l'étude des limites d'évolutivité, c'est-à-dire le nombre maximum de bus et de flexibilités régulées, ce qui permet tout de même d'atteindre la précision souhaitée. La mise en œuvre d'un contrôle multi-agents peut également être envisagée pour la régulation simultanée de réseaux de distribution encore plus importants.

La dernière partie des travaux peut également être considérée comme une extension de la tâche de contrôle de la tension tout en ajoutant un objectif lié à la puissance échangée entre les réseaux de distribution et de transport. L'échange de puissance entre les réseaux de distribution et de transport joue un rôle important dans la garantie de conditions de fonctionnement sûres et fiables pour les deux systèmes. Cependant, ces systèmes sont gérés indépendamment par deux opérateurs distincts - les gestionnaires de réseau de transport (GRT) et les CCO, ce qui complique le contrôle global. Diverses stratégies de contrôle utilisées dans les réseaux de distribution peuvent affecter fortement l'échange de puissance à l'interface GRT-CCO, peuvent éventuellement provoquer des effets indésirables dans les réseaux de transport et rendre difficile la prédiction des régimes et leur contrôle. De plus, ces dernières années, les gestionnaires de réseau ont été confrontés à de nouveaux défis en raison de la pénétration croissante des sources d'énergie renouvelables. Par exemple, un excès de production locale peut entraîner un flux d'énergie inverse du réseau de distribution vers le réseau de transport amont, ce qui peut également provoquer une augmentation de tension dans des conditions de faible charge et compliquer en même temps le contrôle de l'équilibre consommation/production pour le GRT. Pour ces raisons, le GRT peut demander aux CCO de maintenir les consommations d'énergie, actives et réactives, au niveau de la sous-station sous une limite prédéterminée. La violation de ces limites peut entraîner des pénalités pour les CCO, qui augmentent considérablement avec la durée et la profondeur de la violation.

Le contrôleur basé sur TD3PG a affiché de meilleurs résultats que le PPO pour l'objectif de contrôle de la puissance, réduisant l'échange de puissance apparent au-delà de la limite fixe de 13,8 MVAh à seulement 0,13 MVAh au cours de la période de test. Le PPO a obtenu 1,07 MVAh et l'algorithme traditionnel basé sur l'optimisation a atteint 0,12 MVAh. Des algorithmes ont été testés pour différents objectifs d'échange de puissance. À l'exception de l'objectif de limiter la puissance apparente en dessous de la valeur fixe, des algorithmes ont été testés pour l'objectif de contrôler la puissance apparente avec une limite variable et pour l'objectif d'éviter l'injection de puissance active dans le réseau de transport ainsi que pour contrôler à la fois les limites d'injection et de consommation de puissance active. Les algorithmes ont montré une bonne efficacité pour tous ces cas. Il a également été conclu que pour le système à 11 bus considéré, les objectifs de contrôle de la tension et de la puissance ne pouvaient pas être

complètement atteints simultanément. Ainsi, des fronts de Pareto pour les deux objectifs et deux algorithmes ont été construits tout en ajustant les paramètres de la fonction de récompense. Le contrôleur basé sur PPO a montré de meilleurs résultats avec la domination de l'objectif de régulation de tension sur la limite de puissance, tandis que TD3PG a montré de meilleurs résultats avec la domination de l'objectif de régulation de puissance. Là encore, l'évolutivité des algorithmes a été testée sur un réseau de distribution à 55 bus, et l'algorithme RL qui n'utilise pas de mesures de puissance dans les nœuds, mais uniquement de la puissance agrégée au niveau de la sous-station, a montré des résultats de gestion de l'énergie proches de l'optimum.

Les perspectives futures sur cette partie pourraient inclure des investigations sur les performances de l'algorithme RL avec des limites de puissance dynamiques pouvant varier en temps réel. De telles conditions d'exploitation peuvent devenir pertinentes pour les réseaux avec une prédominance de la production variable basée sur des sources d'énergie renouvelables. Il est également intéressant d'étudier diverses fonctions de récompense et leur impact sur les performances des algorithmes.

En conclusion, les techniques d'apprentissage automatique peuvent contribuer à la fois aux objectifs de détection/classification et de régulation de l’CCO. Les algorithmes RL ne nécessitent pas une connaissance complète du système mais peuvent le comprendre uniquement en fonction des résultats des interactions avec celui-ci. Cela leur permet d'être utilisés en cas de manque de mesures, ce qui distingue avantageusement les contrôleurs basés sur RL des contrôleurs traditionnels basés sur l'optimisation. Cependant, les algorithmes RL sont des boîtes noires et les résultats de leur entraînement ne peuvent pas être prévisibles à 100%. Par conséquent, les meilleurs résultats nécessitent le choix correct des hyperparamètres et des données de récompense et de données d'entraînement représentatives. Dans ce cas, les contrôleurs basés sur RL peuvent remplacer avec succès certains des outils existants et apporter des fonctionnalités supplémentaires à la gestion du réseau de distribution.

References

[1] D. Waldemar, “The role of distribution system operator in the context of energy security – the case of Poland,” *Modern Electric Power Systems* 2010, Wroclaw, Poland.

[2] “Report on Regulatory Frameworks for European Energy Networks 2021,” *Council of European Energy Regulators*, 31.01.2022, Ref: C21-IRB-61-03.

[3] X. Fang, S. Misra, G. Xue, D. Yang, “Smart grid — the new and improved power grid: a survey,” *IEEE Commun Surv Tutor*, 14(4):944–980, 2012.

[4] P.P. Malya, L. Fiorini, M. Rouhani et al., ”Electric vehicles as distribution grid batteries: a reality check,” *Energy Inform* 4, 29, 2021. https://doi-org.gaelnomade-1.grenet.fr/10.1186/s42162-021-00159-3

[5] G. Fernández, N. Galan, D. Marquina, D. Martínez., A. Sanchez., P. López, H. Bludszuweit, J. Rueda, “Photovoltaic Generation Impact Analysis in Low Voltage Distribution Grids,” *Energies* 2020, 13, 4347. https://doi.org/10.3390/en13174347

[6] Z. Rehman, M. Anzar, R. Sohail, A. Wamiq, N. Usman, S. Khurram, “Prosumer based energy management and sharing in smart grid,” *Renewable and Sustainable Energy Reviews*, Volume 82, Part 1, 2018, pp. 1675-1684, ISSN 1364-0321, https://doi.org/10.1016/j.rser.2017.07.018.

[7] B.F. Wollenberg, "Expert Systems in Energy Management Systems," *IFAC Proceedings Volumes*, Volume 22, Issue 9, 1989, Pages 19-24, ISSN 1474-6670, https://doi.org/10.1016/S1474-6670(17)53239-5.

[8] T. Jain, “K-Means Clustering,” April 2021, URL: https://medium.datadriveninvestor.com/k-means-clustering-ac3ff1d3463d

[9] H. Abdi, L.J. Williams, “Principal component analysis,” *Wiley Interdiscip. Rev. Comput. Stat.* 2 (4) (2010) 433–459

[10] M. Minsky, S. Papert, “Perceptrons: An Introduction to Computational Geometry,” *MIT Press.* 1969, ISBN 978-0-262-63022-1.

[11] Richard Sutton, "Learning to predict by the methods of temporal differences". *Machine Learning*. 3 (1): 9–44. 1988, doi:10.1007/BF00115009

[12] “Machine Learning Market Size, Share & Trends Analysis Report By Component, By Enterprise Size, By End Use (Healthcare, BFSI, Law, Retail, Advertising & Media), And Segment Forecasts,” 2019 – 2025. Report ID: GVR-4-68038-095-8. URL: https://www.grandviewresearch.com/industry-analysis/machine-learning-market

[13] M. Balat, “Hydropower Systems and Hydropower Potential in the European Union Countries,” *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, Volume 28, 2006

[14] “Solar energy production capacity in France between 2010 and 2020. Statista Research Department,” Jul 5, 2021, URL: https://www.statista.com/statistics/1073552/capacity-production-energy-solar-la-france/

[15] “Renewable Energy Statistics 2022, The International Renewable Energy Agency,” *IRENA*, Abu Dhabi, 2022. URL: https://www.irena.org/Publications

[16] “Bilan Electrique 2022,” *RTE*, URL : https://bilan-electrique-2021.rte-france.com/

[17] D. Motyka, M. Kajanová and P. Braciník, "The Impact of Embedded Generation on Distribution Grid Operation," *7th International Conference on Renewable Energy Research and Applications (ICRERA)*, 2018, pp. 360-364, doi: 10.1109/ICRERA.2018.8566741.

[18] T. Hossen, S. J. Plathotam, R. K. Angamuthu et al., “Short-term load forecasting using deep neural networks (DNN),” *North American Power Symposium (NAPS)*, 2017 [Online]. Available: https://engineering.und.edu/electrical/faculty/prakashranganathan/ p232.pdf

[19] E. G. Kardakos, M. C. Alexiadis, S. I. Vagropoulos, C. K. Simoglou, P. N. Biskas and A. G. Bakirtzis, "Application of time series and artificial neural network models in short-term forecasting of PV power generation," *2013 48th International Universities' Power Engineering Conference (UPEC)*, 2013, pp. 1-6, doi: 10.1109/UPEC.2013.6714975.

[20] Donadio, Lorenzo, Jiannong Fang, and Fernando Porté-Agel. 2021. “Numerical Weather Prediction and Artificial Neural Network Coupling for Wind Energy Forecast,” *Energies* 14, no. 2: 338. https://doi.org/10.3390/en14020338

[21] S. Hosein and P. Hosein, “Load forecasting using deep neural networks,” *IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, Washington, DC, USA, 23-26 April 2017.

[22] X. Dong, L. Qian, and L. Huang, ‘‘A CNN based bagging learning approach to short-term load forecasting in smart grid,’’ *Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov*. Aug. 2017, pp. 1–6.

[23] X. S. Dong, L. J. Qian, and L. Huang, “Short-term load forecasting in smart grid: a combined CNN and K-means clustering approach,” *IEEE International Conference on Big Data and Smart Computing (BigComp)*, Jeju, Korea, Feb. 13–16, 2017.

[24] A. Gensler, J. Henze, B. Sick, N. Raabe, “Deep Learning for solar power forecasting—An approach using AutoEncoder and LSTM Neural Networks,” *Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Budapest, Hungary, 9–12 October 2016; pp. 2858–2865.

[25] V. Suresh, P Janik, J. Rezmer, Z. Leonowicz, “Forecasting Solar PV Output Using Convolutional Neural Networks with a Sliding Window Algorithm,” *Energies* 2020, 13, 723.

[26] K. Wang, X. Qi, H. Liu, “A comparison of day-ahead photovoltaic power forecasting models based on deep learning neural network,” *Appl. Energy* 2019, 251, 113315.

[27] R.A. Rajagukguk, R.A.A. Ramadhan, H.-J. Lee, "A Review on Deep Learning Models for Forecasting Time Series Data of Solar Irradiance and Photovoltaic Power," *Energies* 2020, 13, 6623. https://doi.org/10.3390/en13246623

[28] S. Hanifi, X. Liu, Z. Lin, S. Lotfian, "A Critical Review of Wind Power Forecasting Methods—Past, Present and Future," Energies 2020, 13, 3764. https://doi.org/10.3390/en13153764

[29] B. Bilal, M. Ndongo, K. H. Adjallah, A. Sava, S.M.F. Kebe, P.A. Ndiaye, V. Sambou, "Wind turbine power output prediction model design based on artificial neural networks and climatic spatiotemporal data," Proceedings of the IEEE International Conference on Industrial Technology 2018, Lyon, France, 20–22 February 2018; pp. 1085–1092.

[30] Zhao, Y.; Ye, L.; Li, Z.; Song, X.; Lang, Y.; Su, J. "A novel bidirectional mechanism based on time series model for wind power forecasting," Appl. Energy 2016, 177, 793–803.

[31] D. L. Donaldson, D. Jayaweera. "Effective solar prosumer identification using net smart meter data," *Int J Electric Power Energy Syst* 2020;118:105823. https:// doi.org/10.1016/j.ijepes.2020.105823. ISSN 0142-0615.

[32] X. Zhang, S. Grijalva, "A data-driven approach for detection and estimation of residential PV installations," *IEEE Trans Smart Grid*, 7(5):2477–85, 2016.

[33] S. M. Ali and S. D. Silvey, "A general class of coefficients of divergence of one distribution from another," *J. Royal Stat. Soc. Ser. B*, vol. 28, no. 1, pp. 131–142, 1966.

[34] F. Wang, K. Li, X. Wang, L. Jiang, J. Ren, Z. Mi, et al., "A distributed PV system capacity estimation approach based on support vector machine with customer net load curve features," *Energies*, 11(7):1750, 2018.

[35] J. M. Malof, K. Bradbury, L. M. Collins, R. G. Newell "Automatic detection of solar photovoltaic arrays in high resolution aerial imagery," *Appl Energy* 2016;183:229–40.

[36] F. Sossan, L. Nespoli, V. Medici, M. Paolone, "Unsupervised disaggregation of photovoltaic production from composite power flow measurements of heterogeneous prosumers," *IEEE Trans Industr Inf* 2018;14(9):3904–13.

[37] 6th CEER benchmarking report on the quality of electricity and g. supply, "Electricity – voltage quality," *Council of European Energy Regulators (CEER), Tech. Rep.*, 2016 (cit. on p. 9).

[38] P. N. Vovos, A. E. Kiprakis, A. R. Wallace, and G. P. Harrison, "Centralized and distributed voltage control: Impact on distributed generation penetration," *IEEE Transactions on power systems*, vol. 22, no. 1, pp. 476–483, 2007 (cit. on p. 8).

[39] M. Z. C. Wanik, M. M. Bukshaisha, and S. R. Chaudhry, "Pv generation in distribution network and its impact on power transformer on-load tap changer operation," *2017 IEEE Manchester PowerTech, IEEE*, 2017, pp. 1–6 (cit. on p. 12).

[40] F.U. Nazir, B.C. Pal, R.A. Jabr, “A two-stage chance constrained Volt/VAR control scheme for active distribution networks with nodal power uncertainties,” *IEEE Trans. Power Syst.* 34 (1), pp. 314–325, Jan. 2019.

[41] V. Sarfi and H. Livani, ‘‘Optimal Volt/VAR control in distribution systems with prosumer DERs,’’ *Electr. Power Syst. Res.*, vol. 188, Nov. 2020, Art. no. 106520, doi: 10.1016/j.epsr.2020.106520.

[42] N.Chettibi, A.Massi Pavan, A.Mellita, A.J.Forsyth, R.Todd, “Real-time prediction of grid voltage and frequency using artificial neural networks: An experimental validation,” *Sustainable Energy, Grids and Networks*, September 2021, doi: 10.1016/j.segan.2021.100502.

[43] N. Kumar, B. Singh, B. K. Panigrahi, “PNKLMF-Based Neural network control and learning-based HC MPPT technique for multiobjective grid integrated solar PV based distributed generating system,” *IEEE Trans. Ind. Inf.*, 15, pp. 3732–3742, June 2019.

[44] S. Li, Y. Sun, M. Ramezani, and Y. Xiao, “Artificial neural networks for Volt/VAR control of DER inverters at the grid edge,” *IEEE Trans. on Smart Grid*, vol. 10, no. 5, pp. 5564–5573, Sep. 2019.

[45] Y. Xu, W. Zhang, W. Liu, and F. Ferrese, ‘‘Multiagent-based reinforcement learning for optimal reactive power dispatch,’’ *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 6, pp. 1742–1751, Nov. 2012.

[46] J. Duan, D. Shi, R. Diao, H. Li, Z. Wang, B. Zhang, D. Bian, Z. Yi, “Deep-reinforcement-learning-based autonomous voltage control for power grid operations,” *IEEE Trans. Power Syst*. 35, pp. 814–817, Jan. 2020.

[47] S. Wang, J. Duan, D. Shi, C. Xu, H. Li, R. Diao, and Z. Wang, “A data-driven multi-agent autonomous voltage control framework using deep reinforcement learning,” *IEEE Trans. Power Syst*., Nov. 2020.

[48] D. Cao, J. Zhao, W. Hu, F. Ding, N. Yu, Q. Huang, Z. Chen. “Model-free voltage control of active distribution system with PVs using surrogate model-based deep reinforcement learning,” *Applied Energy*. v. 306, Part A, January 2022.

[49] D. Cao, J. Zhao, W. Hu, N. Yu, F. Ding, Q. Huang, and Z. Chen, “Deep reinforcement learning enabled physical-model-free two-timescale voltage control method for active distribution systems,” *IEEE Trans. on Smart Grid*, 2021

[50] J-F. Toubeau, B. B. Zad, M. Hupez, Z. De Grève, F. Vallée, “Deep reinforcement learning-based voltage control to deal with model uncertainties in distribution networks,” *Energies*, vol. 13, no. 15, 2020.

[51] X. Sun and J. Qiu, "A Customized Voltage Control Strategy for Electric Vehicles in Distribution Networks With Reinforcement Learning Method," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 10, pp. 6852-6863, Oct. 2021, doi: 10.1109/TII.2021.3050039.

[52] “TSO-DSO interaction: An Overview of current interaction between transmission and distribution system operators and an assessment of their cooperation in Smart Grids,” *ISGAN Discussion Paper Annex 6 Power T&D Systems*, Task 5. 09.2014

[53] H. Gerard, E. Rivero, D. Six, “Basic schemes for TSO-DSO coordination and ancillary services provision,” *Smart Net*, 02.12.2016

[54] “IRENA (2020), Innovation landscape brief: Co-operation between transmission and distribution system operators,” *International Renewable Energy Agency*, Abu Dhabi.

[55] D. Zhang, “Optimal Design and Planning of Energy Microgrids,” Ph.D Thesis, Department of Chemical Engineering, University College London, 2013.

[56] Y. Riffonneau, S. Bacha, F. Barruel, and S. Ploix, “Optimal power flow management for grid connected PV systems with batteries,” *IEEE Trans. Sustain. Energy*, vol. 2, no. 3, pp. 309–320, 2011.

[57] M. O. Badawy and Y. Sozer, “Power Flow Management of a Grid Tied PV-Battery System for Electric Vehicles Charging,” *IEEE Trans. Ind. Appl*., vol. 53, no. 2, pp. 1347–1357, 2017.

[58] M. Hijjo, F. Felgner, and G. Frey, “PV-Battery-Diesel microgrid layout design based on stochastic optimization,” *2017 6th International Conference on Clean Electrical Power: Renewable Energy Resources Impact, ICCEP 2017*, 2017, pp. 30–35.

[59] A. O. Erick and K. A. Folly, "Reinforcement Learning Approaches to Power Management in Grid-tied Microgrids: A Review," *Clemson University Power Systems Conference (PSC)*, 2020, pp. 1-6, doi: 10.1109/PSC50246.2020.9131138.

[60] T.A. Nakabi, P. Toivanen, “Deep reinforcement learning for energy management in a microgrid with flexible demand,” *Sustainable Energy, Grids and Networks*, 2021, 25, doi:10.1016/j.segan.2020.100413

[61] J. G. Vlachogiannis and N. D. Hatziargyriou, “Reinforcement learning for reactive power control,” *IEEE transactions on power systems*, vol. 19, no. 3, pp. 1317–1325, 2004.

[62] P. Kofinas, A.I. Dounis, G.A. Vouros, “Fuzzy Q-learning for multi-agent decentralized energy management in microgrids,” *Appl. Energy 219* (2018) 53–67, http://dx.doi.org/10.1016/j.apenergy.2018.03.017.

[63] R. El Helou, D. Kalathil and L. Xie, "Fully Decentralized Reinforcement Learning-Based Control of Photovoltaics in Distribution Grids for Joint Provision of Real and Reactive Power," *IEEE Open Access Journal of Power and Energy*, vol. 8, pp. 175-185, 2021, doi: 10.1109/OAJPE.2021.3077218.

[64] C. Li, C. Jin, and R. Sharma, ‘‘Coordination of PV smart inverters using deep reinforcement learning for grid voltage regulation,’’ *Proc. 18th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2019, pp. 1930–1937.

[65] S. Ali, B. J. Choi, “State-of-the-Art Artificial Intelligence Techniques for Distributed Smart Grids: A Review,” *Electronics*, 9, 2020.

[66] PyTorch. PyTorch Documentation — PyTorch 1.5.0 Documentation. URL: https://pytorch.org/docs/stable/index.html (visited on 06/08/2020).

[67] Taud, H., Mas, J. (2018). "Multilayer Perceptron (MLP)," *Geomatic Approaches for Modeling Land Change Scenarios. Lecture Notes in Geoinformation and Cartography. Springer*, https://doi-org.gaelnomade-1.grenet.fr/10.1007/978-3-319-60801-3_27

[68] S. Abirami, P. Chitra, "Chapter Fourteen - Energy-efficient edge based real-time healthcare support system," *Advances in Computers, Elsevier*, Volume 117, Issue 1, 2020, pp. 339-368,

[69] "Smartmeter energy consumption data in london households," [Online]. Available: http://data.London.gov.uk/dataset/smartmeter-energy-usedata-in-London-households.

[70] F. Bu, Y. Yuan, Z. Wang, K. Dehghanpour, and A. Kimber, "A Time-series Distribution Test System based on Real Utility Data," *2019 North American Power Symposium (NAPS), Wichita, KS, USA*, 2019, pp. 1-6. [PDF]

[71] M. Sengupta, Y. Xie, A. Lopez, A. Habte, G. Maclaurin, and J. Shelby, "The national solar radiation data base (NSRDB)," *Renewable Sustain. Energy Rev*., vol. 89, pp. 51–60, 2018.

[72] Dark Sky API [Online]. Available: https://darksky.net/dev

[73] K. Strunz, N. Hatziargyriou, and C. Andrieu, "Benchmark systems for network integration of renewable and distributed energy resources," *Cigre Task Force C*, vol. 6, 2009

[74] Pandapower [Online]. Available: //pandapower.readthedocs.io/en/v2.9.0/

[75] Y. Zheng, Q. Liu, E. Chen, Y. Ge, J. L. Zhao, "Exploiting Multi-Channels Deep Convolutional Neural Networks for Multivariate Time Series Classification," *Web-age information management*, pp 298–310, 2014.

[76] A. Petrusev, R. Rigo-Mariani, V. Debusschere, P. Reignier, N. Hadjsaid, "Model-free Detection of Distributed Solar Generation in Distribution Grids Based on Minimal Exogenous Information," ELECTRIMACS 2022, 2022.

[77] N. Ketkar, "Stochastic Gradient Descent. In: Deep Learning with Python," *Apress, Berkeley, CA*. 2017 https://doi-org.gaelnomade-1.grenet.fr/10.1007/978-1-4842-2766-4_8

[78] D. P. Kingma and J. L. Ba. "Adam: A method for stochastic optimization," *International Conference on Learning Representations (ICLR)*, 2015.

[79] O. Parson et al., "Dataport and NILMTK: A building data set designed for non-intrusive load monitoring," 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), 2015, pp. 210-214, doi: 10.1109/GlobalSIP.2015.7418187.

[80] J. Morin, F. Colas, J. Dieulot, S. Grenard, X. Guillaud, "Embedding OLTC nonlinearities in predictive Volt Var control for active distribution networks," *Electr. Power Syst. Res.* 143, pp. 225–234, 2017.

[81] M. H. K. Tushar, C. Assi, “Volt-VAR control through joint optimization of capacitor bank switching, renewable energy, and home appliances,” *IEEE Trans. Smart Grid* 9 (5), pp. 4077–4086, Sept. 2018.

[82] Y. P. Agalgaonkar, B. C. Pal and R. A. Jabr, "Stochastic Distribution System Operation Considering Voltage Regulation Risks in the Presence of PV Generation," *IEEE Trans. on Sustainable Energy*, vol. 6, no. 4, pp. 1315-1324, Oct. 2015.

[83] S.V. Kamarthi, S. Pittner, “Accelerating neural network training using weight extrapolations,” *Neural Networks* 12 (9) (1999) 1285–1297.

[84] S. Fujimoto, H. van Hoof, and D. Meger, “Addressing function approximation error in actor-critic methods,” arXiv preprint arXiv:1802.09477, 2018.

[85] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.

[86] R. S. Sutton and A. G. Barto, “Reinforcement learning,” “On-policy Control with Approximation,” in *Reinforcement Learning: an Introduction*. Second edition. Cambridge, MA: The MIT Press, 2018.

[87] J. Achiam, "Spinning Up in Deep Reinforcement Learning (2018)," URL: https://spinningup. openai. com/en/latest 11 2018.

[88] P. Palanisamy “Hands-On Intelligent Agents with OpenAI Gym: Your guide to developing AI agents using deep reinforcement learning,” Packt Publishing Ltd, 2018.

[89] R. S. Sutton and A. G. Barto, “Reinforcement learning,” “Temporal-Difference Learning,” in *Reinforcement Learning: an Introduction. Second edition*. Cambridge, MA: The MIT Press, 2018.

[90] H. H. Tan and K. H. Lim, “Review of second-order optimization techniques in artificial neural networks backpropagation,” 2019. IOP Conf. Ser.: Mater. Sci. Eng. 495 012003

[91] R. Liu and J. Zou, “The effects of memory replay in reinforcement learning,” *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, October, 2018.

[92] N.S. Keskar, J. Nocedal, D. Mudigere, M. Smelyanskiy, and P.T.P. Tang, “On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima,” *ICLR 2017*.

[93] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014

[94] L. P. Kaelbling, M. L. Littman, A. W. Moore, “Reinforcement Learning: A Survey,” *Journal of Artif. Intelligence Research* 4, pp. 237-285, May 1996.

[95] Y. Yuan, “Recent advances in trust region algorithms,” *Math. Program*. 151, 249–281 (2015). https://doi-org.gaelnomade-1.grenet.fr/10.1007/s10107-015-0893-2

[96] H. Van Hasselt, Y. Doron, F. Strub, M. Hessel, N. Sonnerat, and J. Modayilm “Deep reinforcement learning and the deadly triad,”. arXiv preprint arXiv:1812.02648, 2018

[97] Basic Standard Types. Lines. Pandapower documentation. URL : https://pandapower.readthedocs.io/en/v2.9.0/std_types/basic.html

[98] M. A. Putratama, R. Rigo-Mariani, V. Debusschere, Y. Besanger, “Parameter Tuning for LV Centralized and Distributed Voltage Control with high PV production,” *Powertech conference 2021, Madrid, Spain (Online)*, July 2021. DOI: 10.1109/PowerTech46648.2021.9494802

[99] A. Hazra, “Using the confidence interval confidently,” *J. Thorac. Dis*., 9(10), pp. 4125–4130, Oct. 2017, doi: 10.21037/jtd.2017.09.14

[100] I. Pisica, C. Bulac, M. Eremia, “Optimal distributed generation location and sizing using genetic algorithms,” *Intelligent System Applications to Power Systems, 2009. ISAP'09. 15th International Conference on, IEEE*, 2009.

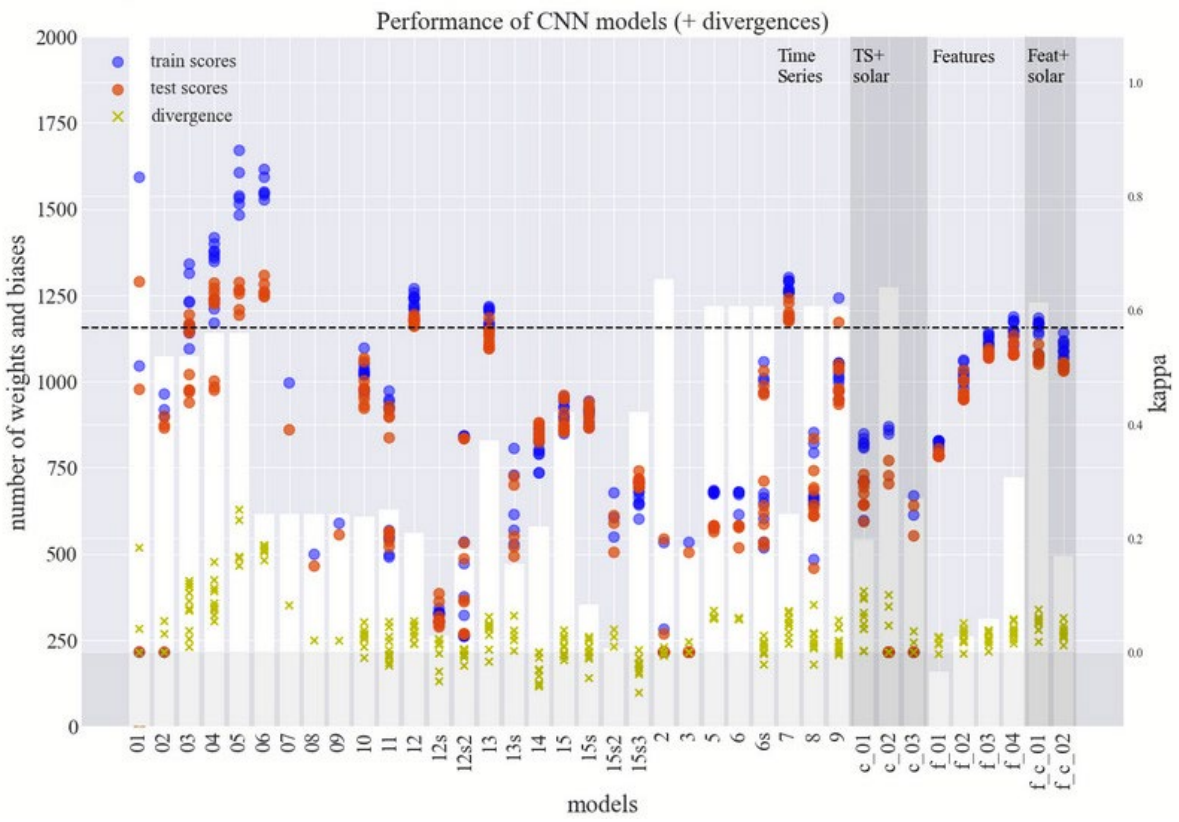
[101] F. Marten et al., "Analysis of a reactive power exchange between distribution and transmission grids," 2013 *IEEE International Workshop on Intelligent Energy Systems (IWIES)*, 2013, pp. 52-57, doi: 10.1109/IWIES.2013.6698561.

[102] M. Tomaszewski, S. Stanković, I. Leisse and L. Söder, "Minimization of Reactive Power Exchange at the DSO/TSO interface: Öland case," *2019 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*, 2019, pp. 1-5, doi: 10.1109/ISGTEurope.2019.8905712.

Appendices

Appendix 1. Performance of CNN with different architectures

The methodology of the model search is to first explore several architectures and have a spread look into the parameter space. From that the best models for each type are selected and further examined. This includes fixed parameter choices for comparability.



This graph shows the kappa scores (right y-axis) for all models (x-axis) in the exploration phase. The model abbreviations *f* and *c* indicate features and a combination of time series or features with weather data, *s* - time series. Red points represent test scores, blue ones - training scores. The divergence (train minus test score) is marked by green crosses at the bottom of the graphic. The white bars in the background signify the number of parameters (left y-axis) of the models. The dashed line indicates the kappa baseline score. The leftmost model has 8000 parameters. The selection highlights only those samples that have been trained with 1000 epochs.

Cohen's kappa score is calculated with the following formula:

$$K = \frac{p_0 - p_e}{1 - p_e}$$

Where p_0 is the overall accuracy of the model and p_e is the measure of the agreement between the model predictions and the actual class values as if happening by chance.

Observations:

f,c Adding weather data to models with features does not seem to improve scores. More input is not always better.

01 has 8832 parameters and overfits heavily. Too complex models tend to overfit.

02/03 03 has ReLU, and a performs slightly better, thus ReLU can improve performance.

03/04 04 has batch normalization with better performance. Batch normalization can improve performance.

04/05 05 has no dropout, but a higher divergence. Dropout helps generalizing.

05/06 06 has half the amount of parameters and the same performance. The same result can be achieved with 1140 and 616 parameters.

06/07 06 has no dropout, worse scores. Dropout does not always help.

6/62 6s has sigmoid and lower divergence. Sigmoid can reduce overfitting.

10/11 11 has many small convolutions and a smaller divergence.

01-16/2-9 2-9 are very complex and have no regularization, performance is worse. Regularization can help, model does not need to be overly complex.

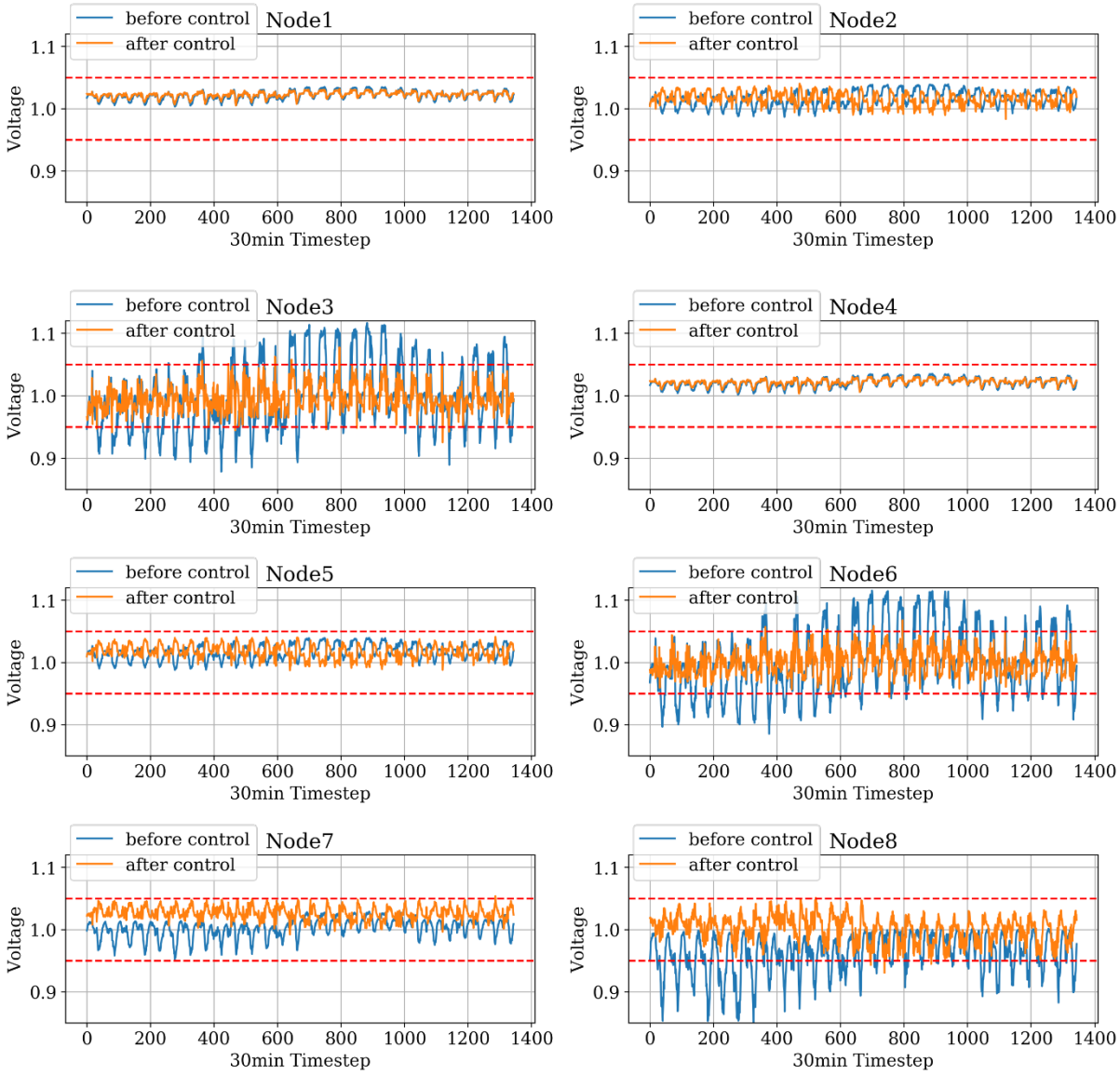
01-07 Those models overfit heavily as training scores surpass those for testing by far.

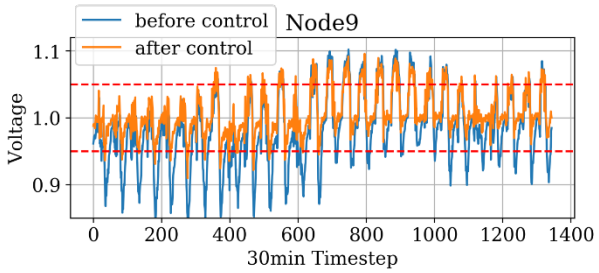
model	conv1	conv2	conv3	linear	ReLU	Sigmoid	Dropout	BatchNorm	Max-Pooling	# params
	1:4, 4:3	-	-	60:20, 20:1	1	Y	-	-	-	1.296
2	1:4, 4:3	-	-	60:20, 20:1	Y	Y	-	-	-	1.317
3	-	-	-	24:20, 20:1	-	-	-	-	-	542
5	1:5, 5:4, 4:2	-	-	36:11, 11:2	-	-	-	-	-	541
6	1:10, 10:3	-	-	60:11, 11:2	-	-	-	-	-	828
6s	1:10, 10:3	-	-	60:11, 11:2	-	Y	-	-	-	828
7	1:40, 40:3	-	-	60:11, 11:2	-	Y	-	-	-	1218
8	1:40, 40:3	-	-	60:11, 11:2	Y	Y	-	-	-	1218
9	1:10, 10:10, 10:10, 10:3	-	-	48:11, 11:2	Y	Y	-	-	-	1316
01	1:50, 50:50, 50:3	-	-	54:11, 11:2	Y	Y	-	-	-	8832
02	1:10, 10:10, 10:3	-	-	54:11, 11:2	-	Y	Y	-	-	1072
03	1:10, 10:10, 10:3	-	-	54:11, 11:2	Y	Y	Y	-	-	1072
04	1:10, 10:10, 10:3	-	-	54:11, 11:2	Y	Y	Y	Y	-	1140
05	1:10, 10:10, 10:3	-	-	54:11, 11:2	Y	Y	-	Y	-	1140
06	1:10, 10:3	-	-	60:7, 7:2	Y	Y	-	Y	-	616
07	1:10, 10:3	-	-	60:7, 7:2	Y	Y	Y	Y	-	616
08	1:10, 10:3	-	-	60:7, 7:2	-	Y	Y	Y	-	616
09	1:10, 10:3	-	-	60:7, 7:2	Y	Y	Y (0.5)	Y	-	616
10	1:20, 20:2	-	-	40:8, 8:2	Y	Y	Y (0.5)	Y	-	608
11	1:6, 6:6, 6:6, 6:2	-	-	32:8, 8:2	Y	Y	Y (0.5)	Y	-	628
12	1:10, 10:8, 8:5, 5:2	-	-	32:2	Y	Y	Y	Y	-	561
12s	1:7, 7:5, 5:3, 3:1	-	-	16:2	Y	Y	Y	Y	-	262
12s2	1:10, 10:8, 8:5, 5:2	-	-	32:2	Y	Y	Y	Y	-	511
13	1:20, 20:10, 10:2	-	-	6:2	Y	Y	Y	Y	Y	830
13s	1:10, 10:10, 10:2	-	-	6:2	Y	Y	Y	Y	Y	470
14	1:10(1), 10:1(1)	1:10(1), 10:1(3)	1:10(1), 10:1(5)	33:10, 10:2	Y	Y	Y	Y	Y	581
15	1:10(1), 10:1(1)	1:10(1), 10:1(3)	1:10(1), 10:1(5)	33:10, 10:2	Y	Y	Y	Y	-	581
15s	1:10(1), 10:1(1)	1:10(1), 10:1(3)	1:10(1), 10:1(5)	33:2	Y	Y	Y	Y	-	581
15s2	1:4(1), 4:1(1)	1:4(1), 4:1(3)	1:4(1), 4:1(5)	33:2	Y	Y	Y	Y	-	581
15s3	1:10(1), 10:1(1)	1:10(1), 10:1(3)	1:10(1), 10:1(5)	33:2	Y	Y	Y (0.5)	Y	-	581
16	1:20, 20:2	-	-	48:2	PReLU	Y	Y	Y	-	346

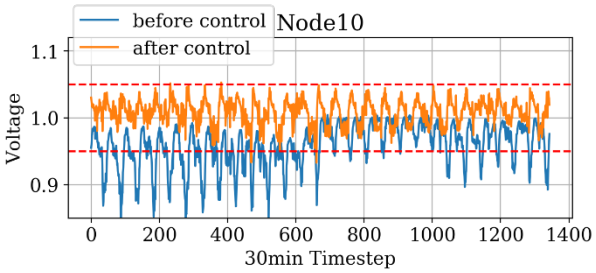
model	branch 1	branch 2	final	ReLU	Sigmoid	Dropout	BatchNorm	Max-Pooling	# params
c_01	1:20, 20:2	1:20, 20:2	24:2	Y	Y	Y	Y	Y	542
c_02	1:20, 20:10, 10:2	1:20, 20:10, 10:2	18:10, 10:2	Y	Y	Y	Y	Y	1274
c_03	1:10, 10:2	1:10, 10:2	24:15, 15:2	Y	Y	Y	Y	Y	659
c_04	1:10, 10:8, 8:5, 5:2	1:10, 10:2	80:2	Y	Y	Y	Y	-	759
c_05	1:30, 30:2	1:30, 30:2	96:2	Y	Y	Y	Y	-	930
c_06	1:10, 10:8, 8:5, 5:3	1:10, 10:8, 8:5, 5:3	68:2	Y	Y	Y	Y	-	1108
c_f_01	1:20, 20:2	4:10, 10:4	16:2	Y	Y	Y	Y	Y	402
c_06	1:10, 10:8, 8:5, 5:3	1:10, 10:8, 8:5, 5:3	68:2	Y	Y	Y	Y	-	1108
f_01	-	-	7:11, 11:5, 5:2	-	Y	-	-	-	160
f_02	1:10, 10:3	-	9:7, 7:2	Y	Y	Y	Y	-	259
f_03	-	-	7.15, 15:8, 8:2	Y	Y	Y	Y	-	312
f_04	-	-	7:30, 30:12, 12:2	Y	Y	Y	Y	-	722
f_04a	-	-	7:30, 30:12, 12:2	PReLU	Y	Y	Y	-	724
f_05	-	-	7:30, 30:12, 12:2	PReLU	Y	Y	Y	-	722
f_06	-	-	7:45, 45:20, 20:2	PReLU	Y	Y	Y	-	1452
f_c_01	7:30, 30:12	1:10, 10:2	24:15, 15:2	Y	Y	Y	Y	Y	1229
f_c_02	7:10, 10:5	1:10, 10:2	17:10, 10:2	Y	Y	Y	Y	Y	493
f_c_02s	(lin) 7:5, 5:5	1:5, 5:2	17.2	Y	Y	Y	Y	Y	192

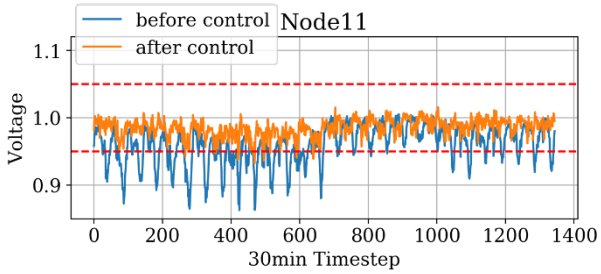
Appendix 2. Voltage profiles four representative weeks of the test year with constant impedances and PPO control

PPO-based controller mainly maintains the voltages of all nodes in the given limits except for the node 9 which contains PV, but no flexibility. The RL control mostly handles the problem of undervoltage thanks to flexibilities from neighboring nodes. However, overvoltage's cannot be as efficiently mitigated as undervoltage. This is explained by the lack of flexibility in this node. Moreover, the controller cannot lower the voltage of node 9 using flexibilities in nodes 8 and 10, because it would lead to significant voltage drop in the node 11 below 0.95 p.u.




before control
after control
Node9
Voltage
1.1
1.0
0.9
0
200
400
600
800
1000
1200
1400
30min Timestep


before control
after control
Node10
Voltage
1.1
1.0
0.9
0
200
400
600
800
1000
1200
1400
30min Timestep


before control
after control
Node11
Voltage
1.1
1.0
0.9
0
200
400
600
800
1000
1200
1400
30min Timestep

Appendix 3. The algorithm's principle for selecting the best EPI/VPI pairs

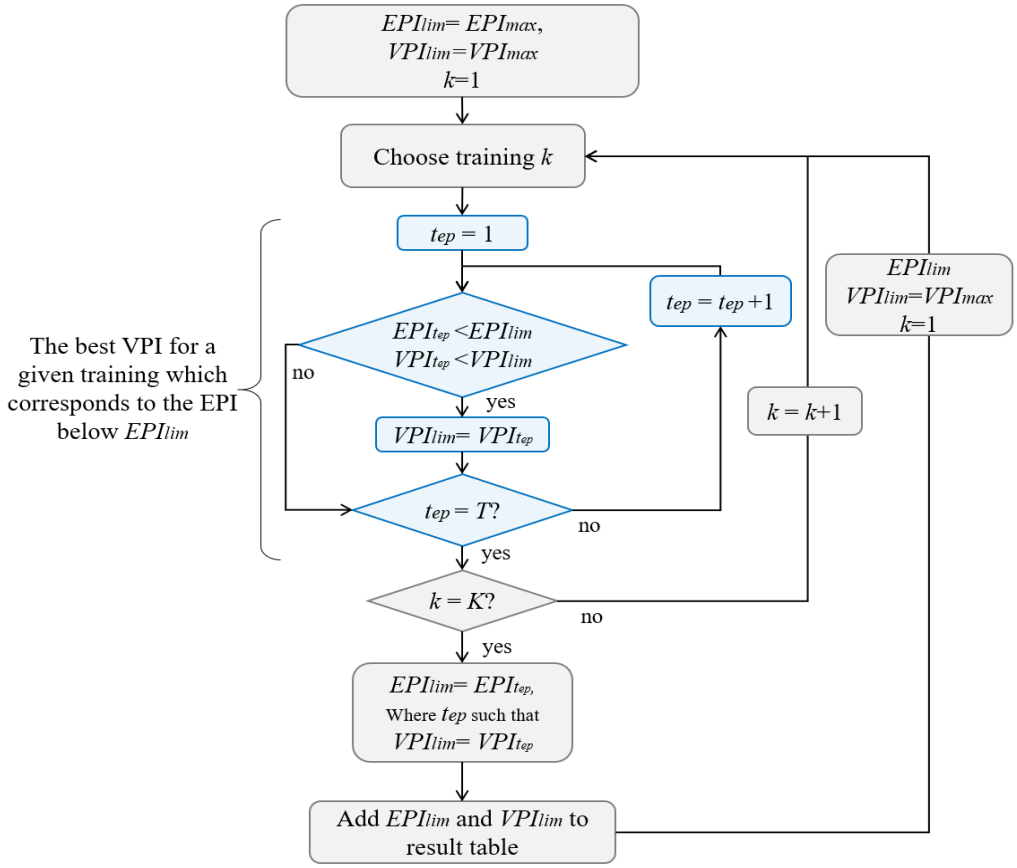


Figure 0.1. The algorithm's principle for selecting the best EPI/VPI pairs

First, both limits EPI_{lim} and VPI_{lim} are set to the values that are known to be greater than any EPI and VPI from training results. Next, the first training result k=1 from the set of training results of size K is selected and for each of its steps t_{ep} (the size of step is equal 50 epochs in this work), the corresponding computed EPI_{tep} and VPI_{tep} are checked to be less than EPI_{lim} and VPI_{lim}. If both inequalities are true, $VPI_{lim} = VPI_{tep}$ and loop continues until the last step t=T of the last training k=K. The EPI_{lim} is then updated to the EPI value that corresponds to the best found VPI_{tep}, and their values are written to the result table. The process then starts over, except EPI_{lim} retains its last value. This continues until the algorithm can find a new pair of EPI_{lim} and VPI_{lim}.

Appendix 4. Learned action-value functions

To check how good the pretrained algorithms approximate the reward, their learned value functions (for the episode with the length of 1 timestep) is compared with the constructed reward function (3.10). A graphical representation of the reward function with the found values of the parameters α, β and ω (responsible for trade-off between voltage regulation and SOC control) for various SOC values and voltage deviations is shown in Figure 0.2 (a). Due to limitations of the 3D representation, only 2 variables can vary when calculating the reward. Thus, the surface is plotted under assumption that all SOC of batteries (the first variable) and similarly all voltages in the nodes (the second variable) have the same values (except node 1, which is equal to slack bus voltage). The constructed reward is a convex function which, due to the exponential part for SOC (3.10), significantly varies at extremes and has almost a plateau if the state variables remain in the range of [0.2, 0.8] and [0.95, 1.05] for SOC and voltage, respectively.

The Q-function of the Critic network (designed to evaluate actions, and thus showing how RL algorithm learned the reward function) of pre-trained TD3PG algorithm is presented in Figure 0.2 (b). Q-function of the RL algorithm depends on the state variables $\mathbf{V}_t^{\mathrm{n}}, \mathbf{SOC}_t^{\mathrm{f}}, t, \mathbf{P}_t^{\mathrm{f}}, \mathbf{Q}_t^{\mathrm{f}}$ and action variables $\mathbf{P}_{t+1}^{\mathrm{f}}, \mathbf{Q}_{t+1}^{\mathrm{f}}$. Thus, it is plotted under the same assumptions as reward function but with additional condition that timestep t is fixed at noon and current flexibility powers $\mathbf{P}_t^{\mathrm{f}}, \mathbf{Q}_t^{\mathrm{f}}$ and action at the next step $a_t = \mathbf{P}_{t+1}^{\mathrm{f}}, \mathbf{Q}_{t+1}^{\mathrm{f}}$ are equal to zeros.

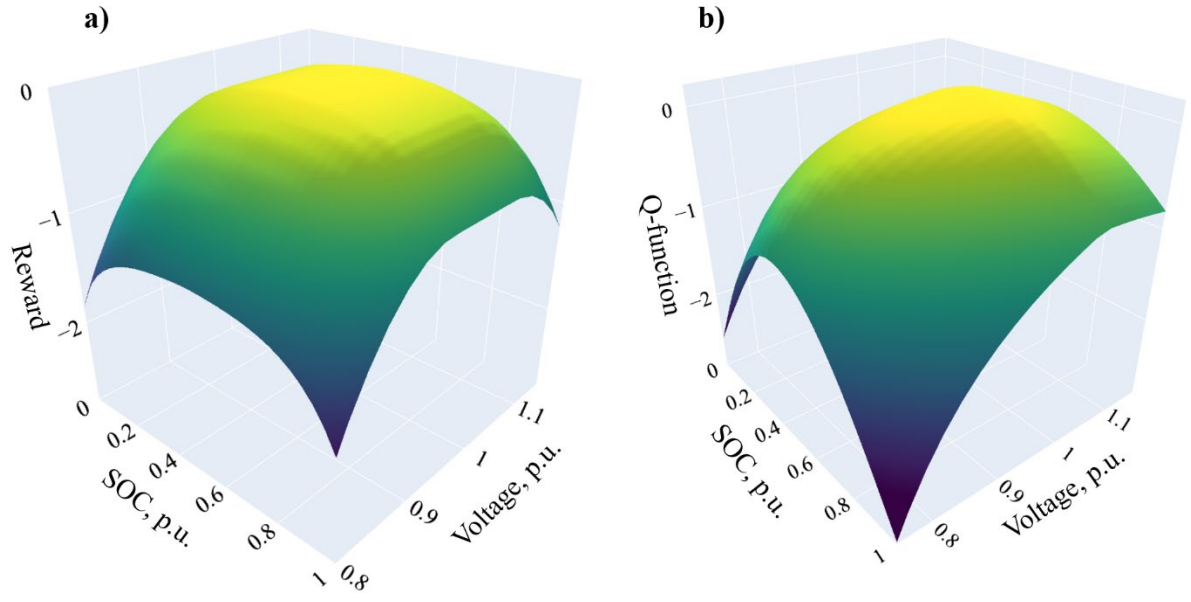


Figure 0.2. Reward function (a) and Q-function obtained after the training (b)

Thus, plotted Q-function represents the algorithm's expectation of a reward for the next timestep without any flexibility regulation. The resulting shape of Q-function for the best trained TD3PG algorithm represents a convex function which, however, has slight differences from the reward function - the most noticeable difference is observed at the extreme points. Q-function has a less slope for high voltage (>1.05 p.u.), reaching a value of -0.81 for 1.15 p.u.,

while the reward function reaches -1.24. Similarly, it reaches -2.45 for voltages of 0.75 p.u. whereas the real reward is -1.82. This can be explained by several factors.

First, noticeable overvoltages can be only in the nodes with PV and only in summer, (as demonstrated in Figure 3.19 and Figure 3.21). Thus, the algorithm rarely encounters such voltages and has poorer generalization in this region. Secondly, due to the limitations of the 3D representation, it is assumed that 10 nodes have the same undervoltage or overvoltage value, which is physically impossible, and this also affects the generalization of the function. Finally, the trained algorithm in the later stages of the training does not have significant overvoltage or undervoltage at any timestep *t* due to near-optimal control it performed in the previous timestep *t-1*. Thus, its Q-function is mostly updated for values close to range of [0.95, 1.05] p.u. for voltage and [0.2, 0.8] for SOC. To demonstrate this, the evolution of the average reward (which comprises SOC and voltage penalization) during the training is shown in Figure 0.3 and can be compared with reward values in Figure 0.2 (a).

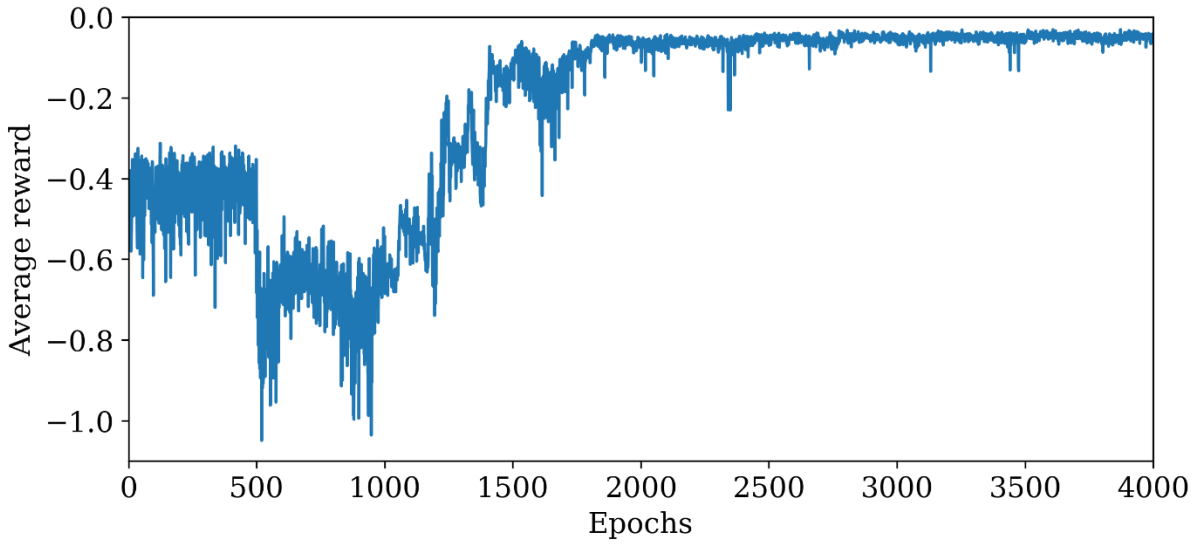


Figure 0.3. Average reward during TD3PG training

The average reward fluctuates around -0.5 for the first 500 epochs which correspond to the stage of generating random actions. It then drops to around -0.7 when TD3PG starts using its own policy, and then slowly increase to around -0.08 after 1700 epochs. Thereby, for more than half of the training, the algorithm operates with an average reward of -0.08, thus staying mostly near the optimal voltage and SOC values, as can be seen in Figure 0.2 (a) (the best value of reward function is 0 for SOC of 0.5 and voltage inside the range of [0.95;1.05]). However, the algorithm does not need to know the exact reward values for these extremes, because its objective is to maintain the voltage close to the nominal voltage, thus operating in a much narrower range, except for the first timestep.

It is also worth noting that Q-function is convex at [0.95, 1.05] p.u. voltage range for the fixed SOC values (but reward function is constant in this case). This is because the algorithm also takes into account the uncertainty of load and generation. This uncertainty exists due to the fact that algorithm, based on voltages in the timestep *t,* chooses the action which is applied for

the timestep $t+1$. For this reason, Q-function fairly estimates the possible reward r_t for a voltage $\mathbf{V_t^n}$ =1 p.u. higher than the possible reward for a voltage of 0.95 or 1.05 p.u. Because the nodes with a voltage of 1p.u. at timestep t are more probably stay within the limits [0.95, 1.05] even with a new load and generation values at the next timestep $t+1$.

Finally, it is more important for the algorithm to learn not the exact absolute values of each point of the reward function, but the relationship between these points to understand which of any two states is better.

Appendix 5. Grid search for hyperparameters of PPO algorithm

The results of a grid search for the best set of hyperparameters resulting in the lowest VPI for PPO algorithm from section 4.5 are presented below. First, initial values of hyperparameters are found using dynamic tuning, where the parameters are adjusted one at a time towards a direction of a lower VPI until there is no improvement. In the second step, a grid search is performed, which investigates the performance of the algorithm with the values of hyperparameters that simultaneously vary around the values found by dynamic tuning.

Parameters found by dynamic tuning with a VPI of 9.16% are presented in Table 1.

Table 1. Parameters for the PPO Algorithm

Parameter	Value	Parameter	Value
Hidden layers structure	32x32	α	3
ω	7.5	β	4000
Activation function ("Actor" network)	Tanh	Activation function ("Critic" network)	ReLU
Nb. of gradient descent steps for policy/value function	20/20	Logarithm of standard deviation	-0.5
Clipping parameter ε	0.1	KL-divergence	0.02
Learning rate for policy optimizer	$1e^{-4}$	Learning rate for value function	$2e^{-4}$

Table 2 presents the results of sensitivity analysis of VPI results from values of Policy learning rate, KL-divergence, Clipping parameter ε and Initial standard deviation. Other parameters are fixed according to Table 1. For convenience, each parameter's change in the second, third and fourth columns of Table 2 is highlighted in red. For each set of hyperparameters two separate runs are done to mitigate the stochasticity of the training. Best VPI values are highlighted in green.

Table 2. Policy learning rate, KL-divergence, Clipping parameter ε and Initial standard deviation impact study

№	Learning rate for policy optimizer	KL-divergence	Clipping parameter ε	Initial value of logarithm of standard deviation	best VPI	Number epochs for the best VPI
1.	1e-4	0.02	0.1	-0.5	8.98	800
2.	1e-4	0.02	0.1	-0.5	8.95	800
3.	3e-4	0.02	0.1	-0.5	8.84	1750
4.	3e-4	0.02	0.1	-0.5	9.44	1650
5.	1e-3	0.02	0.1	-0.5	9.82	1650
6.	1e-3	0.02	0.1	-0.5	9.89	600
7.	1e-4	0.05	0.1	-0.5	9.19	600

8.	1e-4	0.05	0.1	-0.5	9.03	600
9.	3e-4	0.05	0.1	-0.5	9.38	300
10.	3e-4	0.05	0.1	-0.5	9.17	1750
11.	1e-3	0.05	0.1	-0.5	9.33	2900
12.	1e-3	0.05	0.1	-0.5	9.51	1650
13.	1e-4	0.1	0.1	-0.5	8.75	550
14.	1e-4	0.1	0.1	-0.5	8.74	550
15.	3e-4	0.1	0.1	-0.5	9.03	250
16.	3e-4	0.1	0.1	-0.5	9.27	850
17.	1e-3	0.1	0.1	-0.5	10.29	1300
18.	1e-3	0.1	0.1	-0.5	9.61	4750
19.	1e-4	0.02	0.3	-0.5	9.08	800
20.	1e-4	0.02	0.3	-0.5	9.24	1500
21.	3e-4	0.02	0.3	-0.5	8.96	800
22.	3e-4	0.02	0.3	-0.5	8.82	800
23.	1e-3	0.02	0.3	-0.5	9.69	800
24.	1e-3	0.02	0.3	-0.5	9.68	600
25.	1e-4	0.05	0.3	-0.5	8.55	800
26.	1e-4	0.05	0.3	-0.5	9.1	750
27.	3e-4	0.05	0.3	-0.5	9.45	300
28.	3e-4	0.05	0.3	-0.5	8.9	550
29.	1e-3	0.05	0.3	-0.5	9.61	750
30.	1e-3	0.05	0.3	-0.5	10.1	3300
31.	1e-4	0.1	0.3	-0.5	8.83	600
32.	1e-4	0.1	0.3	-0.5	8.66	600
33.	3e-4	0.1	0.3	-0.5	9.35	1800
34.	3e-4	0.1	0.3	-0.5	9.26	1800
35.	1e-3	0.1	0.3	-0.5	10.08	3950
36.	1e-3	0.1	0.3	-0.5	9.61	2650
37.	1e-4	0.02	0.5	-0.5	9.44	1500
38.	1e-4	0.02	0.5	-0.5	9.66	750
39.	3e-4	0.02	0.5	-0.5	9.36	1350
40.	3e-4	0.02	0.5	-0.5	9.38	1350
41.	1e-3	0.02	0.5	-0.5	9.57	800
42.	1e-3	0.02	0.5	-0.5	9.85	800
43.	1e-4	0.05	0.5	-0.5	9.49	800
44.	1e-4	0.05	0.5	-0.5	9.33	750
45.	3e-4	0.05	0.5	-0.5	9.6	1350
46.	3e-4	0.05	0.5	-0.5	9.18	800
47.	1e-3	0.05	0.5	-0.5	9.07	750
48.	1e-3	0.05	0.5	-0.5	9.81	2600
49.	1e-4	0.1	0.5	-0.5	9.21	600
50.	1e-4	0.1	0.5	-0.5	9.38	2500
51.	3e-4	0.1	0.5	-0.5	9.94	1200

52.	3e-4	0.1	0.5	-0.5	9.63	550
53.	1e-3	0.1	0.5	-0.5	9.17	550
54.	1e-3	0.1	0.5	-0.5	9.77	450
55.	1e-4	0.02	0.1	-0.2	8.99	950
56.	1e-4	0.02	0.1	-0.2	8.99	950
57.	3e-4	0.02	0.1	-0.2	8.47	600
58.	3e-4	0.02	0.1	-0.2	9.02	500
59.	1e-3	0.02	0.1	-0.2	9.89	4000
60.	1e-3	0.02	0.1	-0.2	9.86	3050
61.	1e-4	0.05	0.1	-0.2	9.26	750
62.	1e-4	0.05	0.1	-0.2	9.17	750
63.	3e-4	0.05	0.1	-0.2	9.41	300
64.	3e-4	0.05	0.1	-0.2	8.88	350
65.	1e-3	0.05	0.1	-0.2	9.5	250
66.	1e-3	0.05	0.1	-0.2	10.12	3450
67.	1e-4	0.1	0.1	-0.2	9.07	800
68.	1e-4	0.1	0.1	-0.2	8.94	750
69.	3e-4	0.1	0.1	-0.2	9.27	450
70.	3e-4	0.1	0.1	-0.2	9.39	450
71.	1e-3	0.1	0.1	-0.2	9.66	250
72.	1e-3	0.1	0.1	-0.2	9.84	4050
73.	1e-4	0.02	0.3	-0.2	9.44	750
74.	1e-4	0.02	0.3	-0.2	9.42	1500
75.	3e-4	0.02	0.3	-0.2	9.32	800
76.	3e-4	0.02	0.3	-0.2	9.01	800
77.	1e-3	0.02	0.3	-0.2	9.65	750
78.	1e-3	0.02	0.3	-0.2	9.24	800
79.	1e-4	0.05	0.3	-0.2	8.94	800
80.	1e-4	0.05	0.3	-0.2	9.36	750
81.	3e-4	0.05	0.3	-0.2	8.52	600
82.	3e-4	0.05	0.3	-0.2	8.99	600
83.	1e-3	0.05	0.3	-0.2	9.63	750
84.	1e-3	0.05	0.3	-0.2	9.7	600
85.	1e-4	0.1	0.3	-0.2	8.86	1700
86.	1e-4	0.1	0.3	-0.2	9.06	750
87.	3e-4	0.1	0.3	-0.2	9.4	650
88.	3e-4	0.1	0.3	-0.2	9.54	1050
89.	1e-3	0.1	0.3	-0.2	9.41	1650
90.	1e-3	0.1	0.3	-0.2	9.85	1600
91.	1e-4	0.02	0.5	-0.2	9.61	4400
92.	1e-4	0.02	0.5	-0.2	9.75	4000
93.	3e-4	0.02	0.5	-0.2	9.11	800
94.	3e-4	0.02	0.5	-0.2	9.52	800
95.	1e-3	0.02	0.5	-0.2	9.3	750

96.	1e-3	0.02	0.5	-0.2	9.73	1300
97.	1e-4	0.05	0.5	-0.2	9.32	2350
98.	1e-4	0.05	0.5	-0.2	9.39	1350
99.	3e-4	0.05	0.5	-0.2	9.18	1000
100.	3e-4	0.05	0.5	-0.2	9.16	800
101.	1e-3	0.05	0.5	-0.2	9.63	800
102.	1e-3	0.05	0.5	-0.2	9.77	650
103.	1e-4	0.1	0.5	-0.2	9.25	800
104.	1e-4	0.1	0.5	-0.2	9.15	1100
105.	3e-4	0.1	0.5	-0.2	9.22	750
106.	3e-4	0.1	0.5	-0.2	9.31	2500
107.	1e-3	0.1	0.5	-0.2	9.3	600
108.	1e-3	0.1	0.5	-0.2	9.99	1050
109.	1e-5	0.02	0.1	-0.2	10.43	3550
110.	1e-5	0.02	0.1	-0.2	10.4	3550
111.	2e-5	0.02	0.1	-0.2	9.55	2400
112.	2e-5	0.02	0.1	-0.2	9.51	2400
113.	5e-5	0.02	0.1	-0.2	9.34	3950
114.	5e-5	0.02	0.1	-0.2	9.32	3950
115.	1e-5	0.05	0.1	-0.2	10.43	3550
116.	1e-5	0.05	0.1	-0.2	10.41	3550
117.	2e-5	0.05	0.1	-0.2	9.58	2400
118.	2e-5	0.05	0.1	-0.2	9.5	2400
119.	5e-5	0.05	0.1	-0.2	9.46	3950
120.	5e-5	0.05	0.1	-0.2	9.51	3950
121.	1e-5	0.1	0.1	-0.2	10.41	3550
122.	1e-5	0.1	0.1	-0.2	10.42	3550
123.	2e-5	0.1	0.1	-0.2	9.65	2400
124.	2e-5	0.1	0.1	-0.2	9.53	2400
125.	5e-5	0.1	0.1	-0.2	9.32	3950
126.	5e-5	0.1	0.1	-0.2	9.2	3950
127.	1e-5	0.02	0.3	-0.2	10.22	4600
128.	1e-5	0.02	0.3	-0.2	10.18	4600
129.	2e-5	0.02	0.3	-0.2	10.08	2050
130.	2e-5	0.02	0.3	-0.2	9.96	4400
131.	5e-5	0.02	0.3	-0.2	9.68	4700
132.	5e-5	0.02	0.3	-0.2	9.61	4200
133.	1e-5	0.05	0.3	-0.2	10.22	4600
134.	1e-5	0.05	0.3	-0.2	10.19	4600
135.	2e-5	0.05	0.3	-0.2	9.4	4850
136.	2e-5	0.05	0.3	-0.2	9.63	4550
137.	5e-5	0.05	0.3	-0.2	9.46	3600
138.	5e-5	0.05	0.3	-0.2	9.58	3950
139.	1e-5	0.1	0.3	-0.2	10.19	4600

140.	1e-5	0.1	0.3	-0.2	10.2	4600
141.	2e-5	0.1	0.3	-0.2	9.57	4850
142.	2e-5	0.1	0.3	-0.2	9.43	4150
143.	5e-5	0.1	0.3	-0.2	9.74	3050
144.	5e-5	0.1	0.3	-0.2	9.47	3200
145.	1e-5	0.02	0.5	-0.2	10.09	4600
146.	1e-5	0.02	0.5	-0.2	10.11	4600
147.	2e-5	0.02	0.5	-0.2	9.81	4050
148.	2e-5	0.02	0.5	-0.2	9.99	4050
149.	5e-5	0.02	0.5	-0.2	9.53	4750
150.	5e-5	0.02	0.5	-0.2	9.8	4750
151.	1e-5	0.05	0.5	-0.2	10.15	4600
152.	1e-5	0.05	0.5	-0.2	10.14	4600
153.	2e-5	0.05	0.5	-0.2	9.39	4750
154.	2e-5	0.05	0.5	-0.2	9.46	4750
155.	5e-5	0.05	0.5	-0.2	9.6	4300
156.	5e-5	0.05	0.5	-0.2	9.6	3300
157.	1e-5	0.1	0.5	-0.2	10.15	4600
158.	1e-5	0.1	0.5	-0.2	10.15	4600
159.	2e-5	0.1	0.5	-0.2	9.68	4850
160.	2e-5	0.1	0.5	-0.2	9.67	4400
161.	5e-5	0.1	0.5	-0.2	9.45	3050
162.	5e-5	0.1	0.5	-0.2	9.57	1700

The best found VPI value of 8.47% corresponds to the following parameters:

- Policy learning rate of 3e-4,
- Initial value of logarithm of standard deviation of -0.2,
- KL-divergence of 0.02,
- Clipping parameter ε of 0.1.

The found values of the first two parameters (Policy learning rate and Initial standard deviation) are fixed, and the study of impact of Value function learning rate, KL-divergence and Clipping parameter ε is performed further. The results are presented in Table 3.

Table 3. Value function learning rate, KL-divergence and Clipping parameter ε impact study

№	Learning rate for value function	KL-divergence	Clipping parameter ***ε***	best VPI	Number epochs for the best VPI
163.	1e-4	0.002	0.05	9.51	1000
164.	1e-4	0.002	0.05	9.38	750
165.	2e-4	0.002	0.05	9.54	1900

166.	2e-4	0.002	0.05	9.5	4750
167.	5e-4	0.002	0.05	9.48	1700
168.	5e-4	0.002	0.05	9.5	4000
169.	1e-4	0.005	0.05	8.95	3850
170.	1e-4	0.005	0.05	9.04	1300
171.	2e-4	0.005	0.05	9.34	750
172.	2e-4	0.005	0.05	8.68	750
173.	5e-4	0.005	0.05	8.74	950
174.	5e-4	0.005	0.05	9.1	600
175.	1e-4	0.01	0.05	8.52	600
176.	1e-4	0.01	0.05	8.87	400
177.	2e-4	0.01	0.05	8.47	400
178.	2e-4	0.01	0.05	8.8	550
179.	5e-4	0.01	0.05	8.7	600
180.	5e-4	0.01	0.05	8.97	400
181.	1e-4	0.002	0.1	9.8	1350
182.	1e-4	0.002	0.1	9.23	1550
183.	2e-4	0.002	0.1	9.71	1700
184.	2e-4	0.002	0.1	9.56	4750
185.	5e-4	0.002	0.1	9.06	2250
186.	5e-4	0.002	0.1	9.87	4750
187.	1e-4	0.005	0.1	8.7	1350
188.	1e-4	0.005	0.1	8.88	950
189.	2e-4	0.005	0.1	9.12	1350
190.	2e-4	0.005	0.1	9.41	1300
191.	5e-4	0.005	0.1	9.28	950
192.	5e-4	0.005	0.1	9.23	950
193.	1e-4	0.01	0.1	8.66	750
194.	1e-4	0.01	0.1	8.53	600
195.	2e-4	0.01	0.1	9.22	600
196.	2e-4	0.01	0.1	8.61	600
197.	5e-4	0.01	0.1	8.6	3650
198.	5e-4	0.01	0.1	9.14	4300
199.	1e-4	0.002	0.3	9.36	1900
200.	1e-4	0.002	0.3	9.61	1900
201.	2e-4	0.002	0.3	9.52	2600
202.	2e-4	0.002	0.3	9.33	1900
203.	5e-4	0.002	0.3	9.58	2250
204.	5e-4	0.002	0.3	9.69	4400
205.	1e-4	0.005	0.3	9.48	1350
206.	1e-4	0.005	0.3	9.64	4400
207.	2e-4	0.005	0.3	9.9	1500
208.	2e-4	0.005	0.3	9.57	2450
209.	5e-4	0.005	0.3	9.91	900

210.	5e-4	0.005	0.3	9.39	1800
211.	1e-4	0.01	0.3	9.13	1350
212.	1e-4	0.01	0.3	9.02	1350
213.	2e-4	0.01	0.3	9.16	1350
214.	2e-4	0.01	0.3	9.17	1500
215.	5e-4	0.01	0.3	9.41	1900
216.	5e-4	0.01	0.3	9.5	1500

This study didn't show noticeable improvement, the best VPI is still 8.47% with the following values of hyperparameters:

- Value function learning rate is 2e-4,
- KL-divergence is 0.01,
- Clipping parameter is 0.05.

Finally, the impact of coefficients of reward function α, β and ω is investigated. The results of trainings are summarized in table 4.

Table 4. α, β and ω impact study

№	α	β	ω	best VPI	Number epochs for the best VPI
217.	2.5	3000	6	8.37	570
218.	2.5	3000	6	8.36	570
219.	3	3000	6	8.86	580
220.	3	3000	6	8.59	770
221.	3.5	3000	6	8.44	750
222.	3.5	3000	6	8.41	570
223.	2.5	4000	6	8.4	750
224.	2.5	4000	6	8.26	570
225.	3	4000	6	8.21	750
226.	3	4000	6	8	930
227.	3.5	4000	6	8.11	770
228.	3.5	4000	6	8.66	800
229.	2.5	5000	6	8.25	930
230.	2.5	5000	6	8.06	750
231.	3	5000	6	8.4	750
232.	3	5000	6	8.25	950
233.	3.5	5000	6	8.39	940
234.	3.5	5000	6	8.2	950
235.	2.5	3000	6.5	8.61	370
236.	2.5	3000	6.5	8.5	580
237.	3	3000	6.5	8.12	600
238.	3	3000	6.5	8.75	590
239.	3.5	3000	6.5	8.55	570

240.	3.5	3000	6.5	8.54	600
241.	2.5	4000	6.5	8.33	750
242.	2.5	4000	6.5	8.49	390
243.	3	4000	6.5	8.49	570
244.	3	4000	6.5	8.29	750
245.	3.5	4000	6.5	8.3	950
246.	3.5	4000	6.5	8.4	940
247.	2.5	5000	6.5	8.03	750
248.	2.5	5000	6.5	8.17	750
249.	3	5000	6.5	8.48	570
250.	3	5000	6.5	8.32	940
251.	3.5	5000	6.5	8.3	930
252.	3.5	5000	6.5	8.46	750
253.	2.5	3000	7	8.89	410
254.	2.5	3000	7	8.6	590
255.	3	3000	7	8.44	380
256.	3	3000	7	8.62	620
257.	3.5	3000	7	8.63	390
258.	3.5	3000	7	8.63	590
259.	2.5	4000	7	8.51	600
260.	2.5	4000	7	8.58	600
261.	3	4000	7	8.43	600
262.	3	4000	7	8.71	370
263.	3.5	4000	7	8.37	750
264.	3.5	4000	7	8.45	390
265.	2.5	5000	7	8.33	750
266.	2.5	5000	7	8.4	720
267.	3	5000	7	7.96	560
268.	3	5000	7	8.15	750
269.	3.5	5000	7	8.49	750
270.	3.5	5000	7	8.1	750
271.	2.5	3000	7.5	9.55	450
272.	2.5	3000	7.5	9.25	550
273.	3	3000	7.5	9.08	550
274.	3	3000	7.5	8.94	600
275.	3.5	3000	7.5	8.67	600
276.	3.5	3000	7.5	8.99	400
277.	2.5	4000	7.5	8.54	600
278.	2.5	4000	7.5	8.74	600
279.	3	4000	7.5	8.77	400
280.	3	4000	7.5	8.69	600
281.	3.5	4000	7.5	8.81	600
282.	3.5	4000	7.5	8.64	600
283.	2.5	5000	7.5	8.83	400

284.	2.5	5000	7.5	8.97	550
285.	3	5000	7.5	8.54	600
286.	3	5000	7.5	8.27	750
287.	3.5	5000	7.5	8.27	600
288.	3.5	5000	7.5	8.65	600
289.	2.5	3000	8	9.71	550
290.	2.5	3000	8	9.84	750
291.	3	3000	8	9.41	400
292.	3	3000	8	9.57	400
293.	3.5	3000	8	9.35	600
294.	3.5	3000	8	9.47	550
295.	2.5	4000	8	9.34	550
296.	2.5	4000	8	9.36	550
297.	3	4000	8	9.54	600
298.	3	4000	8	9.33	550
299.	3.5	4000	8	8.75	600
300.	3.5	4000	8	8.77	350
301.	2.5	5000	8	8.83	600
302.	2.5	5000	8	9.01	600
303.	3	5000	8	9.06	350
304.	3	5000	8	8.95	800
305.	3.5	5000	8	8.85	750
306.	3.5	5000	8	8.61	600

The best VPI obtained is 7.96%, which corresponds to reward coefficients α=3, β=5000 and ω=7. Thus, after 306 executed trainings, the value of VPI is improved from 9.16% to 7.96% by adjusting hyperparameters according to Table 5.

Table 5. Best parameters for the PPO Algorithm

Parameter	Value	Parameter	Value
Hidden layers structure	32x32	α	3
ω	7	β	5000
Activation function ("Actor" network)	Tanh	Activation function ("Critic" network)	ReLU
Nb. of gradient descent steps for policy/value function	20/20	Logarithm of standard deviation	-0.2
Clipping parameter ε	0.05	KL-divergence	0.01
Learning rate for policy optimizer	$3e^{-4}$	Learning rate for value function	$2e^{-4}$