Pontifícia Universidade Católica
do Rio de Janeiro

Mauricio Raphael Waisblum Barg

Deep Reinforcement Learning for Voltage Control in Power Systems

Dissertação de Mestrado

Dissertation presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor: Prof. Marcus Vinicius Soledade Poggi de Aragão

Rio de Janeiro
April 2021

Pontifícia Universidade Católica
do Rio de Janeiro

Mauricio Raphael Waisblum Barg

Deep Reinforcement Learning for Voltage Control in Power Systems

Dissertation presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática. Approved by the Examination Committee:

Prof. Marcus Vinicius Soledade Poggi de Aragão
Advisor
Departamento de Informática – PUC-Rio

Prof. Marco Serpa Molinaro
Departamento de Informática – PUC-Rio

Prof. Alexandre Street de Aguiar
Departamento de Engenharia Elétrica – PUC-Rio

Prof. Thiago Trezza Borges
Departamento de Engenharia Elétrica – UFF

Rio de Janeiro, April 19th, 2021

Mauricio Raphael Waisblum Barg

Graduated in Electrical Engineering by Universidade Federal Fluminense

Bibliographic data

Raphael Waisblum Barg,Mauricio

Deep Reinforcement Learning for Voltage Control in Power Systems / Mauricio Raphael Waisblum Barg; advisor: Marcus Vinicius Soledade Poggi de Aragão. – Rio de janeiro: PUC-Rio, Departamento de Informática, 2021.

v., 62 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui bibliografia

1. Informática – Teses. 2. Sistemas de Potência;. 3. Controle de Tensão;. 4. *Reinforcement Learning*.. I. Soledade Poggi de Aragão, Marcus Vinicius. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

To my family and friends, for their support
and encouragement.

Acknowledgments

To my parents, Maisa and Jayme, my brothers Bruno and Renato, my girlfriend Iris and to others in my close family for their unconditional support and encouragement.

To my many friends and co-workers at Radix that work with me on a daily basis, epecially to Bárbara Siqueira, Elian Pinheiro, Flávio Loução, Hugo Reiser and Maurício Magalhães without whom this work wouldn't be possible, for all the teaching, talks, jokes and good times during these two years.

To my advisor, Marcus Poggi for all his great help and attention during the development and writing of this work.

To the people from ISA CTEEP that worked with me, believed in this project and made it possible.

To the staff at PUC-Rio Informatics Department for their direct and indirect contribution.

This work was developed as part of RD project 00068-0044/2019 from ANEEL titled "Ferramenta Inteligente de Apoio à Decisão em Tempo Real para Centros de Operações de Transmissão - COT", for ISA CTEEP S.A. The project was executed by Radix Engenharia e Software, Rio de Janeiro, Brasil.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Abstract

Raphael Waisblum Barg,Mauricio; Soledade Poggi de Aragão, Marcus Vinicius (Advisor). **Deep Reinforcement Learning for Voltage Control in Power Systems**. Rio de Janeiro, 2021. 62p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Electrical Power Systems are "cyber-physical" systems responsible for the generation and transportation of energy from its generating source to the final customers. During this process many different activities must be conducted in order to keep quality of service and the system's safety and stability. One of these activities regards control of various equipment in order to keep the voltage level on each system bus between specified limits. This control, which is usually conducted by system's operators in real time and by automatic control equipment involves many different constraints and considerations that are hardly ever taken into account during the decision process. In order to mitigate this problem a smart agent capable of deciding which action is best in order to keep the voltages in adequate levels taking into account system's conditions is proposed. The proposed methodology consists on the Deep Reinforcement Learning technique along with three novel variations: *windowed*, *ensemble* and *windowed ensemble Q-Learning*, which consist on the division of the problem in training windows, the usage of multiple learning agents for the same process and on the combination of both these techniques. The variations are tested on academically consecrated test circuits and are capable of attaining expressive results when compared to the traditional Deep Reinforcement Learning approach which is used in other academic studies and also with the systems' intrinsic control, keeping voltage under control along the day.

Keywords

Power Systems; Voltage Control; Reinforcement Learning.

Resumo

Raphael Waisblum Barg,Mauricio; Soledade Poggi de Aragão, Marcus Vinicius. ***Deep Reinforcement Learning* para Controle de Tensão em Sistemas de Potência**. Rio de Janeiro, 2021. 62p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Os sistemas de potência são sistemas "*cyber-físicos*" responsáveis pela geração e transporte da energia elétrica desde sua fonte geradora até os consumidores finais. Durante este percurso, existem diversos processos que devem ser seguidos para se manter a qualidade do serviço e a segurança e estabilidade do sistema. Um destes processos envolve o controle de diversos equipamentos de maneira que a tensão dos barramentos do sistema se mantenha dentro de faixas pré-estabelecidas. Este controle, normalmente realizado pelos operadores do sistema em tempo real e por equipamentos automáticos de controle, envolve um número muito grande de considerações que dificilmente serão avaliadas no momento da decisão. Para contornar este problema, propõe-se a utilização de uma ferramenta inteligente que seja capaz de escolher as melhores ações a serem tomadas para que a tensão do sistema se mantenha nos níveis adequados levando em consideração as variadas condições do sistema. A metodologia utilizada pela ferramenta consiste na técnica de *Deep Reinforcement Learning* juntamente com três novas variações: *windowed*, *ensemble* e *windowed ensemble Q-Learning*, que consistem na divisão do processo otimizado em janelas de treinamento, utilização de múltiplos agentes inteligentes para um mesmo processo e a combinação destas duas metodologias. As variações são testadas em circuitos consagrados na literatura e são capazes de obter resultados expressivos quando comparados com a abordagem de *Deep Reinforcement Learning* tradicional utilizada em outros estudos e com o controle intrínseco do próprio sistema, mantendo a tensão sob controle ao longo do dia.

Palavras-chave

Sistemas de Potência; Controle de Tensão; *Reinforcement Learning.*

Table of contents

1 Introduction **15**

2 The Voltage Control Problem **17**
2.1 Voltage Control Equipment 18
2.2 Current Scenario and Problems 19
2.3 Formal Definition 20

3 State of the Art **22**

4 Reinforcement Learning for Voltage Control **26**
4.1 Reinforcement Learning 26
4.2 Deep Reinforcement Learning 30
4.3 Voltage Control as a DRL Problem 35

5 Computational Experiments and Results **37**
5.1 Training 38
5.2 Results 46

6 Conclusion **58**
6.1 Future Work 58

7 References **59**

List of figures

Figure 1.1 A traditional EPS structure (1) 15

Figure 2.1 Short line model (6): (a) one line diagram; (b) phasor diagram. 18
Figure 2.2 The optimization process flow. 21

Figure 4.1 Reinforcement learning process (29) 27
Figure 4.2 Initial Q-Table 28
Figure 4.3 Final Q-Table 29
Figure 4.4 Deep Q-Learning Structure 30
Figure 4.5 Proposed Windowed Q-Learning 34
Figure 4.6 Proposed Ensemble Q-Learning 34
Figure 4.7 Proposed Windowed Ensemble Q-Learning 35

Figure 5.1 IEEE Circuits: (a) 13-bus; (b) 123-bus; (c) 37-bus; 37
Figure 5.2 Neural Network Structure 39
Figure 5.3 IEEE 13-bus network loss (above) and accumulated reward (below) (Reinforcement Learning) 40
Figure 5.4 IEEE 13-bus network loss (above) and accumulated reward (below) (Windowed Reinforcement Learning) 40
Figure 5.5 IEEE 13-bus network loss (above) and accumulated reward (below) (Ensemble Reinforcement Learning) 41
Figure 5.6 IEEE 13-bus network loss (above) and accumulated reward (below) (Windowed Ensemble Reinforcement Learning) 41
Figure 5.7 IEEE 37-bus network loss (above) and accumulated reward (below) (Reinforcement Learning) 42
Figure 5.8 IEEE 37-bus network loss (above) and accumulated reward (below) (Windowed Reinforcement Learning) 42
Figure 5.9 IEEE 37-bus network loss (above) and accumulated reward (below) (Ensemble Reinforcement Learning) 43
Figure 5.10 IEEE 37-bus network loss (above) and accumulated reward (below) (Windowed Ensemble Reinforcement Learning) 43
Figure 5.11 IEEE 123-bus network loss (above) and accumulated reward (below) (Reinforcement Learning) 44
Figure 5.12 IEEE 123-bus network loss (above) and accumulated reward (below) (Windowed Reinforcement Learning) 44
Figure 5.13 IEEE 123-bus network loss (above) and accumulated reward (below) (Ensemble Reinforcement Learning) 45
Figure 5.14 IEEE 123-bus network loss (above) and accumulated reward (below) (Windowed Ensemble Reinforcement Learning) 45
Figure 5.15 IEEE 13-bus single-day results: Average system voltage during the day 47
Figure 5.16 IEEE 13-bus single-day results: Voltage distribution during the day 48

Figure 5.17 IEEE 13-bus single-day voltage profile: (a) No Control; (b) System Control; (c) Reinforcement Learning; (d) Windowed; (e) Ensemble; (f) Windowed Ensemble; 49
Figure 5.18 IEEE 37-bus single-day results: Average system voltage during the day 50
Figure 5.19 IEEE 37-bus single-day results: Voltage distribution during the day 51
Figure 5.20 IEEE 37-bus single-day voltage profile: (a) No Control; (b) System Control; (c) Reinforcement Learning; (d) Windowed; (e) Ensemble; (f) Windowed Ensemble; 52
Figure 5.21 IEEE 123-bus single-day results: Average system voltage during the day 53
Figure 5.22 IEEE 123-bus single-day results: Voltage distribution during the day 54
Figure 5.23 IEEE 123-bus single-day voltage profile: (a) No Control; (b) System Control; (c) Reinforcement Learning; (d) Windowed; (e) Ensemble; (f) Windowed Ensemble; 55

List of tables

Table 5.1	Test Circuits' State and Action Space Sizes	38
Table 5.2	Reward Values	39
Table 5.3	Agents' Training Time in Seconds	46
Table 5.4	13-bus System Results	56
Table 5.5	37-bus System Results	56
Table 5.6	123-bus System Results	56

List of algorithms

Algorithm 1	Q-Learning	29
Algorithm 2	Double Deep Q-Learning with Experience Replay	33
Algorithm 3	Rewards	36

List of symbols

EPS – Electrical Power Systems

QoS – Quality of Service

SO – System Operator

STATCOM – Static Compensator

TCR – Thyristor-Controller Reactor

TSC – Thyristor-Switched Capacitor

SVC – Static VAR Compensators

OLTC – On-Load Tap Changer

RL – Reinforcement Learning

DRL – Deep Reinforcement Learning

I do not fear computers. I fear the lack of them.

Isaac Asimov, *The Computer Society: The Age of Miracle Chips.*

1
Introduction

Electric Power Systems (EPS) are complex systems with both physical and digital resources that must work together in order to deliver electricity safely and efficiently to all kinds of customers be them industrial, commercial or residential (19). Over the last decades the demand for energy is constantly increasing as people wish for better quality of life (24). With this growth, maintaining the quality of service (QoS) and system safety becomes harder and new tools and resources must be included into the process of operating and controlling the system.

Mainly, the operation of EPSs is executed by several trained professionals called system operators (SO) who must constantly observe the system's condition and conduct different tasks such as dispatch of generators, frequency control, fault mitigation, among others (33). One of these tasks known as voltage control regards keeping voltage on all system buses between certain limits which are usually defined by regulatory associations and take into account the systems' correct functioning. As the demand fluctuates through the day, the voltages on system's buses also change and operators must maneuver and switch several equipment in order to keep it under control.

As most other tasks, voltage control must be executed by the SOs in real time who usually must react in a short time frame as the system's resources may be at stake. Voltage can also be controlled by automatic control equipment that react to voltage fluctuations according to some embedded logic using control mechanisms that monitor the voltage at certain system's points and issues control commands that keep it in between the predefined limits (6).

Because of the fast reaction time needed, operators usually do not have

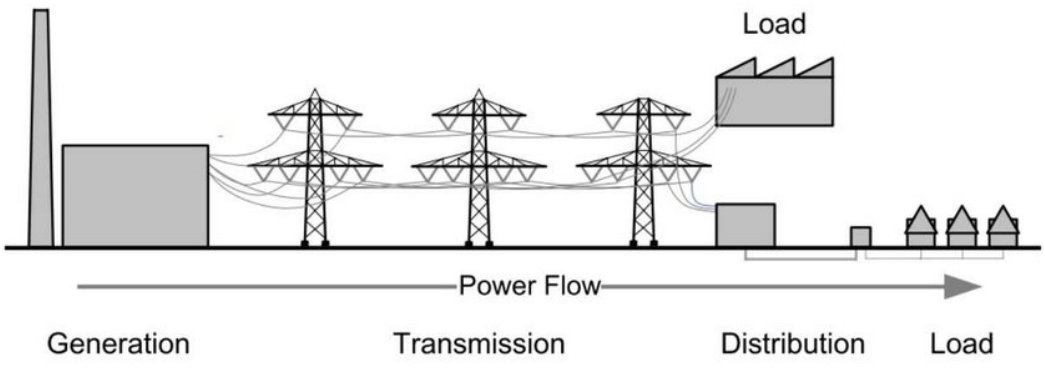

Figure 1.1: A traditional EPS structure (1)

sufficient time to conduct detailed analysis of their actions. Therefore they normally resort solely to their experience and take actions that are known to have previously worked. These actions however may not be optimal, since the system behaves differently depending on many factors such as time of the year, day of the week, climate, etc. and the full extent of their immediate and future impact on the system are mostly unknown. Furthermore, automatic control equipment commonly have fixed control logic that does not account for the system's ever changing dynamic.

Thus, this work proposes a framework to address this problem. The proposed solution utilizes reinforcement learning in order to constantly monitor the system's conditions and take voltage control actions that are as optimal as possible while taking into account the equipment restrictions and different system's parameters. Besides a classic reinforcement learning approach, three new methodologies that account for the systems' complexity are proposed. Although voltage control is a process conducted both on transmission and distribution systems, in this study the methodologies are tested on three IEEE **distribution** circuits with 13, 37 and 123 buses.

This document is structured as follows. In Chapter 2 the voltage control problem is explained in more details, showing its considerations and constraints. In Chapter 3 a extensive revision of studies is conducted showing what has already been done regarding voltage control using both reinforcement learning and other techniques. In Chapter 4 the proposed methodologies are described in detail. In Chapter 5 the case studies and their results are presented. Finally, in Chapter 6 the conclusion is presented and further improvements that can be made are described.

2
The Voltage Control Problem

As briefly explained in Chapter 1, voltage control consists on maneuvering equipment, either automatically or manually in order to counteract voltage fluctuations due to the constant change in demand that happens during the day on power systems.

During the operation, as loads are introduced and removed from the systems, the system's equivalent impedance (Z) changes. This in turn causes changes on the demanded current (I) and the system has to compensate by adjusting its voltages V. When load is increased, Z is reduced and I increases. If the system doesn't have the resources to maintain V, it decreases. Respectively, when load is decreased V tends to increase (33). This can be seen simply by observing Ohm's Law (Equation 2-1).

$$V = Z * I \tag{2-1}$$

In order to supply the loads, power must flow through the system from its generation point to the loads. However, loads are most of the time not purely resistive. That is, not all the power that flows through the system is actually consumed by the load. That means Z is composed by both a real (R) and an imaginary (X) part (Equation 2-2). The imaginary part (called reactance) causes a different kind of power, namely reactive power, to also flow through the system. Reactive power (Q), differently from active power (P), is not actually consumed by loads, nevertheless it plays an important role in the system's behavior by sustaining its electric and magnetic fields.

$$Z = R \pm jX \tag{2-2}$$

The flow of active and reactive powers is directly related to the voltages angle and magnitude respectively (6, 10). This can be seen by observing a power system's simplified short line model and its respective phasor diagram (Figure 2.1). $\overline{v_1}$ and $\overline{v_2}$ are the phase voltages and $\overline{\imath_1}$ and $\overline{\imath_2}$ are the currents at the line extremities. Because this is a simplified short line model, $\imath = \imath_1 = \imath_2$. The angle between $\bar{\imath}$ and $\overline{v}$ is denoted by φ and the components of the current are $I_i = I\cos\varphi$ and $I_r = I\sin\varphi$. Considering $\overline{v_1}$ constant and $\overline{v_2}$ as the phase origin, the potential difference $\Delta\overline{v} = \overline{Z_{\bar{\imath}}}$ has two components, shown in Equation 2-3.

$$\Delta u = RI_i + XI_r, \quad \delta u = XI_i - RI_r \tag{2-3}$$

Considering the single-phase complex power $\overline{S_2} = V_2(I_i + jI_r) = P_2 + jQ_2$, Δu and δu become as shown in Equation 2-4.

$$\Delta u = \frac{RP_2 + XQ_2}{V_2}, \quad \delta u = \frac{XP_2 - RQ_2}{V_2} \tag{2-4}$$

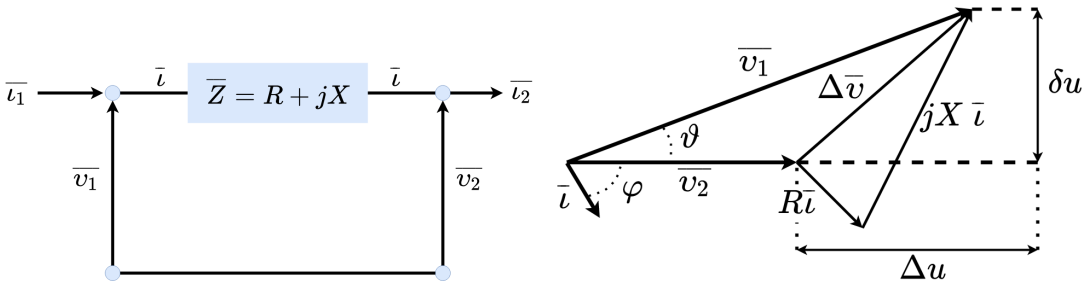

Figure 2.1: Short line model (6): (a) one line diagram; (b) phasor diagram.

Because of a general power systems' characteristic where $R << X$, 2-4 can be simplified, as show in Equation 2-5.

$$\Delta u \approx \frac{XQ_2}{V_2}, \quad \delta u \approx \frac{XP_2}{V_2} \tag{2-5}$$

Therefore, it can be seen that the magnitude variation Δu is mostly dependant on Q_2. Consequently, most of the equipment used to control voltage actually modify reactive power in some way. Usually low load voltage means that the system lacks reactive power, so a device capable of injecting it into the system is necessary. Conversely, when the voltage is too high, the excess reactive power must be consumed. Even though this simplification is not true for all power system sectors (i.e. distribution), it serves as a justification as to why most voltage control equipment actually modify reactive power, independently on where they're used. There are several control equipment capable of executing this task.

2.1 Voltage Control Equipment

Many different equipment capable of controlling voltage exist, with different characteristics. In this work, two main ones will be used: shunt capacitors and tap-changing transformers. There are many other equipment, capable of performing both continuous and discrete control such as: shunt reactors, synchronous condensers, synchronous generators, etc. Also, there are many static devices that are able to change reactive power delivery electronically such as STATCOMs, TCRs, TSCs, SVCs, among others (6). Furthermore, most of these equipment can be controlled either by EPSs' operators or automatically.

2.1.1 Shunt Capacitors

Shunt capacitors are local reactive power compensating devices that are normally connected to a bus bar. They are low cost equipment and are vastly present on both distribution and transmission systems. Capacitors are many times installed as "banks" with different stages, each with a different capacity that can be turned on and off individually to compensate for different amounts of reactive power. Since they operate on stages, capacitor banks are discrete control devices. They can be operated manually by system operators and automatically, using a control scheme that monitors the voltage at the bus where it's installed. Constantly switching capacitor banks on and off degrades its physical integrity and is therefore avoided.

2.1.2 Tap-Changing Transformers

Tap-changing transformers control voltage differently from capacitors and other reactive power devices. Since transformers work by having a different number of turns on each side's coil in order to raise or lower voltage, having a mechanism that can change the number of coils on one or both sides of the transformer can change the ratio which the voltage is transformed. This device is called a on-load tap changer (OLTC) and can operate even while the transformer is energized. Usually, besides the default transforming relation, there is a band of ratios that can be chosen for the transformation (usually ± 16). OLTCs can be operated manually or automatically in which case they're better known as voltage regulators. Since transformers are an integral component on any power system, OLCTs are usually widely available for voltage control across varying voltage levels and are constantly used to maintain voltage within its specified limits.

2.2 Current Scenario and Problems

As society progresses, the demand for energy grows steadily. In consequence, power systems grow more and more complex. With complexity, studies and system analyses become increasingly more difficult as it's harder to simulate perfectly many different components. Furthermore, long known system behaviors can change drastically as its topology is modified. Unfortunately, controlling voltage also become a more arduous task, since the effect a control action might have can be very unpredictable, both on the short and long term.

Besides commonly requiring a fast response time, operators are usually concerned with many different tasks besides voltage control. Thus, there's almost no time available to evaluate the consequences of a control action on the system, leading operators to rely solely on experience and taking actions that are known to have previously worked. However, due to the systems' ever changing dynamic, these assumptions can quickly become false. Furthermore, even though short-term effects can be reasonably guessed based on the operators' experience, future effects are mostly unknown due to the limited analysis capacity.

Equipment that are controlled automatically usually have a fixed control logic that doesn't adapt itself depending on the systems' conditions and topology. This can have unwanted results on the system since a control action can have very different results depending on the moment of the day and which equipment are on or off.

So, the way that the voltage control process is conducted nowadays is bound to change, especially considering the constant modernization of equipment, increase in computing power and digitization of power systems.

2.3 Formal Definition

As indirectly explained on previous sections:

> Voltage Control is the act of **operating** and **configuring** different **control equipment** on the **right moments** in order to keep **bus' voltages** within **specified limits**, which are determined by physical and commercial aspects.

More formally, the problem can be written as shown in Equation 2-6.

$$\begin{aligned} \min \sum_{t=0}^{T} \sum_{b \in B} distance(V_{bt}, \overline{V}) \\ \text{subject to} \\ V_{-} \leq V_{bt} \leq V_{+} \ (a) \\ V_{bt} = f(P, Q, [e_{0t}, e_{1t}, ..., e_{nt}]) \ (b) \\ [e_{il}] \leq [e_{it}] \leq [e_{iu}], \ i = 0...n \ (c) \\ [-1] \leq [e_{it+1} - e_{it}] \leq [+1], \ i = 0...n \ (d) \end{aligned} \tag{2-6}$$

Where t represents the current time step and T the maximum possible time step and n is the total number of equipment. In the real world this time would be continuous as voltage is controlled all the time. Although in order to simulate it, the operation period must be discretized. A certain

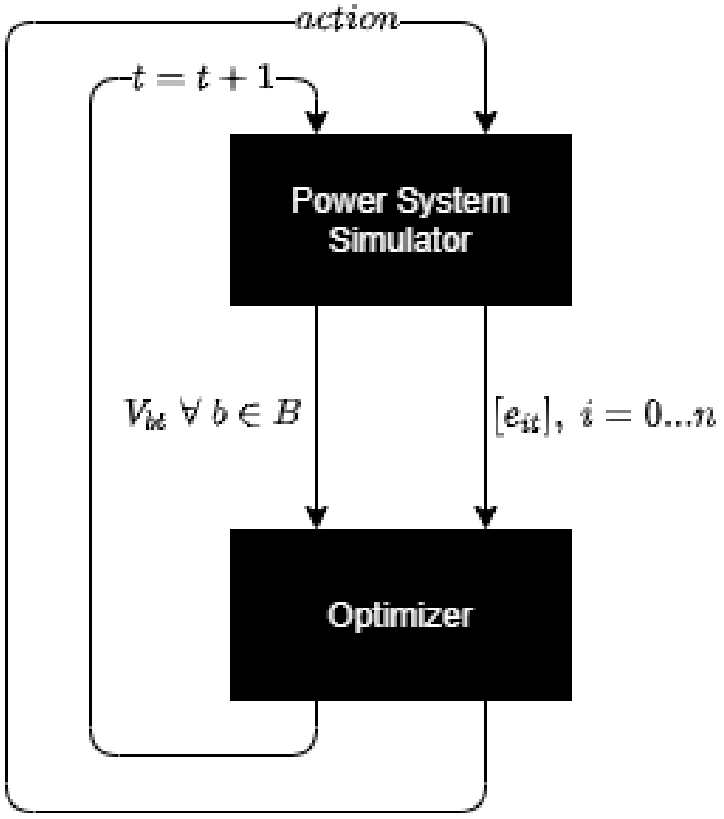

Figure 2.2: The optimization process flow.

system bus is represented by b and B is the set of all system buses. V_{bt} is the voltage at bus b on time t and $\overline{V}$ is the voltage target. For the constraints, in constraint (a), called the *voltage limit constraint*, V_- and V_+ represent the lower and upper voltage limits across all buses and in constraint (b), V_{bt} is represented as a function of the system's active (P) and reactive (Q) powers and all equipment set-points at time t, which is how the voltage is essentially controlled (by changing equipment set-points). In constraint (c), also known as the *equipment set-point constraint*, since control equipment have different set-points restrictions, e_{il} and e_{iu} represent the minimum and maximum set-points that an equipment e_i can have at a certain time (t). Finally, constraint (d) called the *maximum set-point change*, represents how much the set-point of an equipment can change from one time step to another. This is to account for delays that an equipment may have in order to change its set-point.

It is important to note that the data needed for the optimization (i.e. the voltage on each bus) is obtained online and in real time. That means that V_{bt} can only be obtained at time step t and not before that. Therefore, the optimization process is conducted at each time step, as shown in Figure 2.2.

3
State of the Art

Over the years several different techniques have been proposed in order to address the voltage control problem. As it is formally known, *volt-var* control has seen applications of a plethora of different methods, such as classical optimization (26, 4, 27), meta-heuristics (22, 23), classic control (3, 5), neural networks (11), fuzzy logic (18, 17), etc. Also, it has been successfully applied both to distribution and transmission systems.

There are many different approaches to voltage control since it can be seen as both a planning and a real-time operation problem. When treated as a planning problem, it involves choosing the best places to install voltage control equipment (such as capacitors and voltage regulators) (9, 16, 8), defining optimal network topologies (15), etc. When dealt with as a real time problem, it involves switching several equipment on and off in order to satisfy restrictions regarding statutory voltage limits and system safety.

Regarding online control, there are also many different approaches and methodologies used to address it. In (14) a technique is proposed to control tap-changing transformers' taps and capacitor banks in distribution systems. The optimization is done based on the day-ahead load forecast therefore the transformers and capacitors' next day's dispatch schedule is obtained. Consequently, although the control actions are to be executed in real time, the problem itself is solved in a previous setting. Regarding the load profile, the optimization is done based on load levels. This is done in order to address the effect that the probabilistic nature of load-forecasting has on choosing transformer taps. Finally, while controlling the voltage, the equipment switching is kept at a minimum. The results show that the proposed technique is able to balance the systems' voltage profile.

On (13) a classic control approach is used in order to switch capacitor banks on a distribution network. The proposed methodology depends on the existence of remote terminal units (RTUs) on the network. Each RTU is connected to a bus with a capacitor and monitor its voltage as well as upstream and downstream active and reactive power flows. With data from the RTUs, the voltage on its adjacent buses can be estimated. The estimated data is shared between the RTUs which then perform the required calculations to determine the change on the capacitor's reactive power injection needed to bring the bus voltage close to 1 p.u. The change in reactive power needed is used to calculate the number of capacitor steps that needed to be turned on or off. The results

show that the proposed solution is capable of improving the system's voltage profile when compared to keeping the capacitors on fixed steps.

(34) proposes a batch reinforcement learning approach to find the optimal settings of taps on tap changing transformers. The technique uses only voltage measurements and the system's topology information and uses a linear power flow in order to estimate voltage magnitudes and prevent the training process from interfering with the system's operation. A smart operation agent observes the system's state at every time step (consisting of the bus voltages and the tap settings) an chooses an action from all possible tap settings. Then, by observing the effects the action has on the system and repeating this process many times, the agent can learn which action is best for each system state. When compared to an exhaustive search approach (which is not feasible in real life), the agent is able to improve the system's voltage profile along the operation day.

In the work presented by (7) an autonomous operation agent is created using deep reinforcement learning. In order to control the system's voltage, the generators' setpoints are adjusted. Each generator has a range of setpoints that the autonomous agent can choose from. The agent learns both offline (using historical system data) and online (using real power system data). For learning offline, the system's behavior is reproduced using a simulator. In the same manner (34) does, the agent observes the system's state (which are represented by active and reactive power flows and bus voltage magnitudes and phase angles) and then chooses an action from the set of all possible actions. After the action is executed, the agent receives a feedback allowing it to evaluate its quality and effects. The trained agent is then tested on a simulated system and is capable of controlling voltages in normal and contingency scenarios.

(36) proposed a multi-objective optimization approach to tackle the voltage control problem. The meta-heuristic is used together with a fuzzy system in order to improve its performance. The main objectives are to reduce the voltage deviation on each bus from a certain target while keeping active power losses at a minimum. The objectives are pursued while taking into account operational restrictions and limitations. The voltage is adjusted using the generator output voltages, shunt capacitor banks and transformer taps. The equipment configuration is optimized for a static scenario and not through time. After sufficient training, the technique is able to improve the voltage profile keeping it more balanced and closer to 1 p.u. across all buses.

In (25) voltage is optimized online for systems with a high penetration of photovoltaic generators. The inverters associated with each photovoltaic panels are fitted with a control curve that associates the amount of reactive

power that must be injected on the bus connected to the panel depending on its voltage. The control is executed locally, with no coordination between each inverter. For simulation of the proposed technique, a open source distribution system simulator, namely OpenDSS is used. The results show that the system's voltage profile is improved.

On (32) a traditional reinforcement learning approach, more specifically Q-Learning is used in order to control reactive power and thus consequently voltage on power systems. The control is performed for a static scenario of two, 14 and 136 bus power systems. The voltages are controlled using transformers with commutable taps and capacitor banks. The technique is able to find optimal equipment settings in the proposed scenarios with some advantages over other traditional approaches.

In (35) a Deep Reinforcement Learning approach to voltage control in real time is proposed. The technique is modified to deal differently with certain kinds of equipment. The considered equipment, capacitors and smart inverters work in different timescales, therefore, two slight different methodologies are used. The main differences are on the power flow solutions used in order to determine the settings for each type of equipment and on the techniques used. While for the capacitors a DRL approach is used, the inverters are optimized using a more traditional optimization technique. When tested against two, 47 and 123 bus power systems, the technique is able to outperform a randomized approach and a no-action approach.

(30) uses a traditional reinforcement learning approach to control tap settings of OLTC transformers in distribution networks with a high presence of photovoltaic generation. The system state, which is represented by the voltage on certain points of interest is discretized by voltage levels in order to fit with the reinforcement learning model (which can only deal with a limited number of states). The reinforcement learning rewards are proportional to the squared difference between the bus voltages and 1 p.u., which is in general, desirable in distribution networks. The technique is trained both online and offline and is then tested on a 5000-bus real network which shows satisfactory results after sufficient training.

Overall, most works approach voltage control in a offline scenario, optimizing for static settings. In the majority of works, the voltage is controlled using capacitor banks and transformer taps. In some cases, when available, renewable energy resources are used to control the amount of reactive power injected in system buses, which in turn controls voltages. When controlled online, some adaptations are usually made when representing the systems' states, especially regarding load levels which are mostly discretized. Regarding

techniques, a wide variety of methods are used. When dealing with the online problem, most methods require some kind of adaptation either by simplifying the problem or modifying the technique while other methods are well suited to deal with the real time problem out of the box, especially reinforcement learning and its variations.

4
Reinforcement Learning for Voltage Control

In this section, a reinforcement learning (RL) methodology is proposed to deal with the voltage control problem on distribution grids. Because power systems are composed by multiple discrete and continuous variables, the amount of possible configurations it can attain is *quasi-infinite*, rendering the classical tabular reinforcement learning not feasible. Therefore, a deep reinforcement learning technique is developed to deal with this characteristic.

Besides the usual deep reinforcement learning (DRL) approach, three different methodologies are proposed. These propositions involve slight modifications to the reinforcement learning technique that intend to adapt the procedure to specific characteristics of power systems operation.

4.1
Reinforcement Learning

According to (29), reinforcement learning is a machine learning paradigm that revolves around learning through interaction much like humans and other animals learn. In RL, a learning agent which is capable of interacting with an environment both by changing its condition or state through actions and by sensing or observing it. These interactions allow the agent to evaluate the effect that its actions has on the environment when progressing towards a goal and therefore find the actions that produce the most desired results.

There are certain elements that are essential to reinforcement learning. In a way, the main goal of reinforcement learning is creating a map of **state** to **actions** that contain the best action to be taken at a specific state. In the process of creating this map, the **agent** follows a **policy**, which tells it how actions should be taken depending on the state. After taking actions, the effect it has on the **environment** is transmitted to the agent through a **reward signal** which is to be maximized. Environment's states also have a **value** that is intrinsic to them and represents the benefit of being at that state (which is essentially the reward that can be obtained by the actions that can be taken at that state). Finally, in some cases a **model** of how the environment works is necessary in order to plan which actions are to be taken.

By repeating enough times this process of interacting, observing and adapting, the agent is expected to learn the best actions to be taken in a certain activity or process (Figure 4.1).

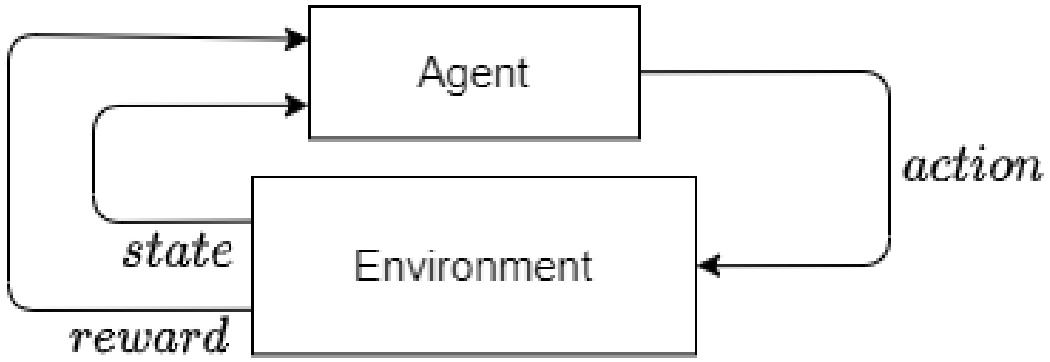

Figure 4.1: Reinforcement learning process (29)

4.1.1 Q-Learning

Many different kinds of reinforcement learning solutions exist depending on the type of problem that is being solved. For the voltage control problem, Q-Learning is a well suited method. Besides being very simple to implement, it is a model-free method, meaning that the model of the environment does not to be fully known (which would be very hard for power systems). Furthermore, it can account for the effect that an action may have on a future state, that is not immediately achieved after taking said action.

Q-Learning works by initializing and building a table composed by every state-action pair possible in a environment and containing the "worth" of taking that action in that state. This "worth" is also known as Q-Value and the main objective of Q-Learning is to find an approximation for these Q-Values which are as close as possible to their true values (Q^*) which represent optimal actions that when taken according to a certain policy, lead to situations that when following the same policy, the chosen action is also optimal. This approximation is achieved through interacting and observing the environment. After creating a table mapping every state to every action, the Q-Values are initialized to zero (Figure 4.2).

The agent then starts interacting with the environment, taking actions by following a policy. In the case of Q-Learning, the policy is the greedy policy. This means that the agent chooses the action with the highest Q-Value for each state. This however is not always the best choice. While during execution, choosing the best action seems like the most reasonable choice (as the agent is supposed to choose the optimal route), during the training procedure while the Q-Values are still being calculated, if the most valuable action is always chosen, the agent may not discover good actions that are only available when a not-so-good action is taken beforehand. In this case, it's a good idea to follow

State-Action Mapping					
	a_1	a_2	...	a_{m-1}	a_m
S_1	0	0	0	0	0
S_2	0	0	0	0	0
$\vdots$	0	0	0	0	0
S_{n-1}	0	0	0	0	0
S_n	0	0	0	0	0

Figure 4.2: Initial Q-Table

a policy that sometimes doesn't choose the best possible action so as to better explore the state space. A policy that accomplishes that and is widely used is called ϵ-*greedy* policy (Equation 4-1). When following it, the agent has a probability (ϵ) of choosing an action at random.

$$a = \begin{cases} argmax(A), \text{ with probability } \epsilon \\ random(A), \text{ with probability } 1 - \epsilon \end{cases} \tag{4-1}$$

Where a is the chosen action, A is the set of all possible actions and ϵ is an arbitrary number in the interval $[0, 1]$.

After each action taken, the agent observes the system's next (s_{t+1}) state and receives a reward (r). With this data, the Q-Values on the table can be updated. This update is done following Equation 4-2. This process is executed until a state is terminal, that is, no further changes can be made in the environment by taking an action. After reaching a terminal state, the so called episode (or iteration) is concluded.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \times [r_t + \gamma \times \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{4-2}$$

Where $Q(s_t, a_t)$ is the Q-Value of taking action a on state s at time t, α is the learning rate which determines how much of the old Q-Value is kept and how much of the new is learned, r_t is the reward obtained by taking action a on state s at time t, γ is the discount factor which determines how much importance is given to future rewards and $\max_a Q(s_{t+1}, a)$ is the estimate of the optimal Q-Value at the next state. That is, what is the best possible action that can be taken at the state that is achieved after taking action a_t at state s_t.

After enough episodes, the values on the Q-Table tend to converge to their true values (Figure 4.3) and allow the choice of the optimal actions. The algorithm is shown on (Algorithm 1)

Although there are many different methods and techniques well suited

State-Action Mapping					
	a_1	a_2	...	a_{m-1}	a_m
S_1	1.233	-1.330	0.550	2.445	0.110
S_2	-3.564	3.224	5.733	9.225	0.625
⋮	2.334	5.854	-9.775	0.002	-0.013
S_{n-1}	7.346	-6.885	-0.032	2.663	4.544
S_n	-4.832	1.332	2.635	9.334	-3.122

Figure 4.3: Final Q-Table

Algorithm 1: Q-Learning

Initialize all Q(s, a) to zero
for $i = 0$ **to** *number of episodes* **do**
  Initialize s
  for *each step of the episode* **do**
    Choose a for the current s from the Q-Table using a policy (e.g ϵ-greedy) Take the chosen action a and observe r and s′
    Update $Q(s, a)$ using equation 4-2
  until s is terminal

to "perform" reinforcement learning, most are limited in a sense, including Q-Learning. Because the main objective is to create a map of state to actions, these have to be finite, otherwise the map will be infinite and therefore impossible to store and consult. While actions are usually limited or can be easily discretized in most cases, states depend on the environment's peculiarities. If the state is composed by many different continuous variables, discretizing the state may be infeasible or lead to great inaccuracies. This is where deep reinforcement learning comes in to play. Its differences and functionality will be further explained in the next section.

4.1.1.1 Hyperparameters

The two main parameters of Q-Learning are the learning rate α and the discount factor γ. They can directly affect the quality and convergence of the learning process.

α or learning rate behaves as in most machine learning techniques. It is also called the step size and determines how fast the model learns. A learning rate of zero means the agent doesn't learn and considers only what is already known when making decisions, while a learning rate of one means only the most recent acquired information is considered. Depending on the problem,

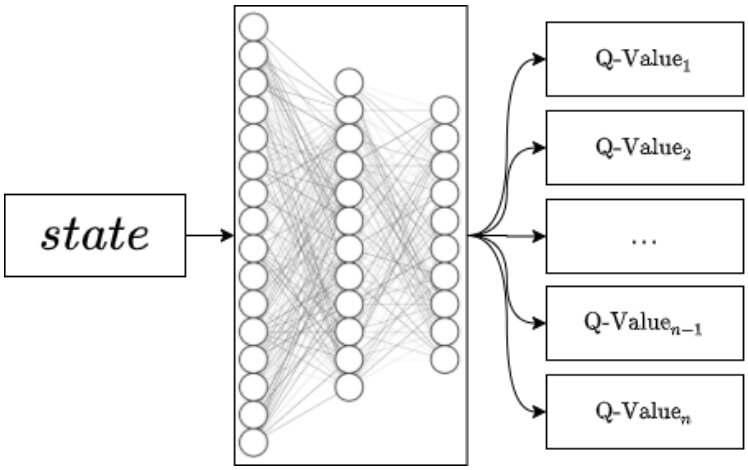

Figure 4.4: Deep Q-Learning Structure

different values of α can achieve good results, albeit small values (such as 0.1) are used.

γ or discount factor is a parameter more commonly seen in reinforcement learning methods. It defines how important is the future for the agent. In simpler terms, the agent can prioritize immediate rewards (γ closer to zero) choosing a very good action that leads it into a state with not so good actions and therefore have a very good short term reward or aim for a better long term reward (γ closer to one) by choosing actions that are immediately not so good but causes the accumulation of a higher reward in the long run. This value doesn't have to be fixed along the iterations. It is possible to start with a lower value and increase it as the training process approaches its end.

4.2 Deep Reinforcement Learning

As stated in the previous section, one of the main disadvantages of Q-Learning is its inability to deal with environments with infinite number of states which are very common in real world problems. This is where deep reinforcement learning, more specifically deep Q-Learning becomes useful. By replacing the Q-Table with a deep neural network, the agent is capable of generalizing for states it has never seen before.

In deep Q-Learning, the neural network's inputs are every variable that represents the system's state and its outputs are the Q-Values for every possible action (Figure 4.4).

The operational part is exactly the same as Q-Learning except instead of consulting the table for Q-Values, the network is consulted. The training procedure is also very similar. Equation 4-2 is used to perform gradient descent on the network and update its weights and biases which translates directly to

updating the table. However, as it turns out, trying to learn the Q-Values with a neural network doesn't work out very well as it is very likely to have instabilities and diverge (20). Because of this, several adaptations have been made in order to stabilize and improve the training process of deep Q-Networks (21, 12). The two main ones called experience replay and double Q-Learning will be used in this work and further explained in the next sections.

4.2.1 Experience Replay

Online learning is difficult for several reasons. One of them is the availability of data used in training the model. Differently from Q-Learning where the Q-Table values are updated at each step of the episode, if the network is trained with only one state sample the achieved results will not be satisfactory. Experience replay can mitigate this problem and provide several other advantages (21).

The way it works is quite simple: at every step of an episode as the agent interacts with the environment, the so called experiences which are composed by the state the agent was in, the action it took, the reward it received and the state it went to, are accumulated in a type of memory. After a certain number of experiences are gathered, this memory is sampled and the chosen experiences are used as a mini-batch to train the neural network.

Besides dealing with the lack of data to train a neural network online, experience replay comes with other advantages. First, because the memory is sampled randomly, the likelihood of sampling consecutive experiences is low. This is good because training with consecutive experiences can be inefficient due to strong correlations that may exist between them which can introduce bias on the network. Second, because after being used in training the experiences are not immediately removed from the memory they can be used multiple times over the course of training the network. This presents great advantages mainly because of two reasons: data efficiency since gathering data in real time may be costly therefore using it multiple times makes better use of it; and because the updates on the network are incremental, using the same data multiple times is beneficial. Third, because the memory is a *deque-like* structure with a **limited size**, older experiences are discarded in favor of new ones which may be more relevant to the learning process.

Experience replay is an essential part of deep Q-Learning as it makes the training process more stable and increases the chances of convergence.

4.2.2

Double Q-Learning

The need for double Q-Learning comes from the *max* expression on Equation 4-2. In this expression, the obtained result is the estimate value of the best possible action in the state that the agent navigated to after taking an action. In deep Q-Learning, in order to calculate this value the neural network is used: the next state (s_{t+1}) is given as an input to the network and the maximum value from all actions is chosen. There are two main problems with this approach: first, by using the same network that is being updated to find a value that is going to be used to update it, the training process essentially becomes chasing a moving target. Second, by using the same network to select and evaluate actions the agent is more likely to select overestimated values (31).

Double Q-Learning proposes using a separate network in order to estimate this value and decouple the process of choosing and evaluating the actions. This second network called *target* network is updated less frequently than the main network, called *online* network. By delaying the training of the *target* network the parameters used to calculate the value estimation are different from the ones used to choose the action which reduces the chances of overestimation.

There are mainly two ways of using the *target* network and two other ways to train it. Regarding usage, the *target* network (θ_t) can either be used to directly find the action value (Equation 4-3) or used to find the action with the highest value which is then calculated by the *online* network (θ_o) (Equation 4-4).

$$\max_a \theta_t(s_{t+1}, a) \tag{4-3}$$

$$\theta_o(s_{t+1}, \text{argmax}_a \theta_t(s_{t+1}, a)) \tag{4-4}$$

The *target* network training can be done either by a *hard* update, which means periodically (every n episodes) copying the weights and biases of the *online* network to it or by a *soft* update, which updates the weights and biases following Equation 4-5 at every step of the episode.

$$\theta_t = \tau \times \theta_o + (1 - \tau) \times \theta_t \tag{4-5}$$

Where τ is the rate of which the parameters are copied over.

Both ways of using and updating the *target* network can have satisfactory results and may vary from application to application.

Both experience replay and double Q-Learning are essential parts of deep Q-Learning as they greatly improve the convergence and training time of the

model. Several other modifications to the regular deep Q-Learning approach exist, as shown in (12). Although technically they could be used on any deep Q-Learning framework, they will not be addressed in this work. The pseudo-code for deep Q-Learning using both experience replay and double Q-Learning is shown in Algorithm 2.

Algorithm 2: Double Deep Q-Learning with Experience Replay

Initialize the online network q
Initialize the target network q'
Initialize the experience replay memory M with size N
for $i = 0$ **to** *number of episodes* **do**
    Initialize s
    for *each step of the episode* **do**
        With probability ϵ *choose a random action* a_t *otherwise choose* $a_t = \max_a q(s, a_t)$
        Take the chosen action a_t *and observe* r *and* s'
        Store (s, a_t, r, s') *in M*
        if $len(M) >$ *minimum number of samples* **then**
            Take m random samples from M into ϕ
            Compute the Q targets $Q^*(s_t, a_t) = r_t + \gamma \times q(s'_t, q'(s'_t, a_t))$
            In q, perform ***gradient descent*** *on* $(Q^*(s_t, a_t) - q(s_t, a_t))^2$
            Update weights w' *and biases* b' *from* q' *following Equation 4-5 or hard copy the parameters every n steps*
    until s is terminal

4.2.3 Novel Proposed Techniques

In addition to the modifications shown in previous sections, two new different additions are proposed to the deep Q-Learning framework. Differently from techniques described in Sections 4.2.1 and 4.2.2 the modifications proposed here do not modify how the learning process works, bur rather change how the techniques are applied to the problem at hand.

The first technique will be named *windowed* Q-Learning. In problems where episodes are too long, the agent may take longer to learn optimal actions for the entire length of the episode. Also, by being long, the episode may have very distinct behaviors on its course. That is, in a certain problem with a constant number of steps per episode, steps i to j may be very different in behavior from steps m to n. The proposed windowed Q-Learning methodology attempts to address this characteristics by dividing the episode into windows (Figure 4.5) and for each window a different agent is trained. That way, each agent only sees states relative to a certain interval and can learn the specificities of each window better when training. During operation, depending

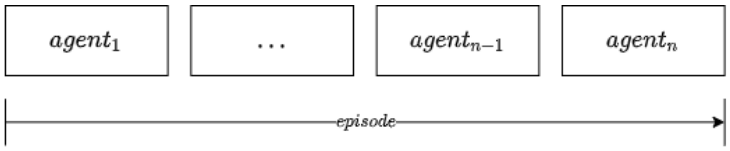

Figure 4.5: Proposed Windowed Q-Learning

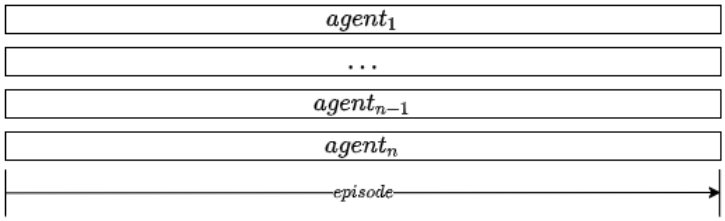

Figure 4.6: Proposed Ensemble Q-Learning

on which window the episode falls into, a different agent is consulted for making decisions.

The second technique will be called *ensemble* Q-Learning and takes inspiration from the way that humans learn. Because when subjected to different experiences people learn things differently, this technique proposes that multiple agents are trained for the same problem (Figure 4.6). These agents can either be equal and rely on the variability of the environment in order to experience things differently or can perceive the environment differently for example by having different reward functions. When operating, every agent is consulted when deciding which action to take. The decision can be made by following several criteria such as averaging every agent's value for the actions and taking the highest average (Equation 4-6) or by taking the action with the maximum value over every agent (Equation 4-7).

$$a = \text{argmax} \begin{pmatrix} \text{avg}(\phi_{11}, \phi_{12}, \phi_{1j}) \\ \text{avg}(\phi_{21}, \phi_{22}, \phi_{2j}) \\ \text{avg}(\phi_{i1}, \phi_{i2}, \phi_{ij}) \end{pmatrix} \tag{4-6}$$

$$a = \text{argmax}(\max \phi_{ij}) \ \forall \ i, j \tag{4-7}$$

Where ϕ_{ij} is the value given to action i by agent j.

The third and final technique is in reality a combination of the other two and is called windowed ensemble Q-Learning. In addition to splitting the episode into windows, for each window multiple agents are trained (Figure 4.7). This allows for the combination of both methods' advantages. One disadvantage may be increased training times. During operation, the actions are also chosen by combining both methods: when the episode moment falls into a certain window, all agents trained for that window are consulted when choosing the action.

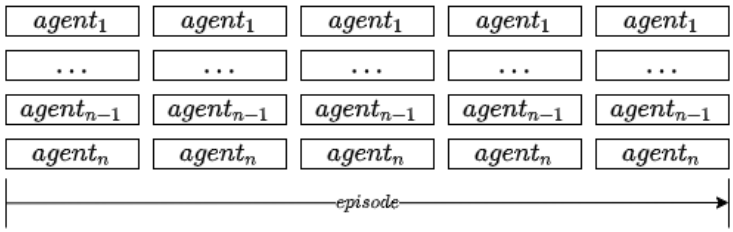

Figure 4.7: Proposed Windowed Ensemble Q-Learning

4.3 Voltage Control as a DRL Problem

Finally, in this section, the voltage control problem will be modeled using the methodologies described above.

As to further justify the necessity of using Deep Q-Learning instead of its tabular counterpart, studies such as (30) show that in order to accurately represent the system state in the latter, considerable effort is needed in order to discretize the representation. Also, in this case, some simplifications are made which are not ideal for a system wide control.

First, the episodes will have a fixed length. This length is 1440, corresponding to every minute in a 24 hour period. At each step, corresponding to a minute the system's loads are updated following a normalized load profile and the agent has the chance of executing an action on the system that can be: changing a capacitor bank stage up or down, changing a transformer tap up and down or doing nothing. The amount of actions vary from system to system as there may be more or less of these type of equipment installed. The action only takes effect on the next minute. This is to account for the delay that may exist between executing the action and the physical equipment actually changing. This also means that the action effect can only be observed after the loads have changed on the next minute. The states both before and after the action are determined by:

- the system's total active and reactive loads;
- the current minute of the simulation;
- the states of the transformers and capacitor banks.

It is important to note, that the state representation has a direct impact on how the agent learn and therefore changing it, can directly affect the learning process. For example, representing the system loads as the sum of all loads may be inaccurate in some cases, since even though the sum may be the same, the load distribution may be fairly different. Also, in preliminary tests was observed that considering the voltages in the state representation led to an unstable training process.

After this process, what is left to observe is the reward received for taking the action. The rewards model the objective that is being pursued by the agent. In this case, the agent is expected to take actions in order to drive the voltage on all system buses closer to a certain target and to keep them between certain operational limits. The rewards are defined following Algorithm 3. The training procedure is no different than that described in the previous sections.

Finally, in order to conduct the training process, since a simulator is needed for the environment, a open-source power system simulator was used: *OpenDSS*. OpenDSS is widely used both academically and commercially by several energy regulation agencies. Its COM interface makes the integration process with most common programming languages easy.

All parameters used in the model such as memory size, neural network structure, reward values, learning rate, discount factor, etc. will be detailed in Chapter 5.

It is important to note that while the training procedure may take a long time, when operating, the results are almost instant since it's only necessary to input the state to the neural network and execute the corresponding output action.

Algorithm 3: Rewards

Initialize $r = 0$
if *action a was the opposite of an action taken in the last 30 minutes* **then**
    $r = r - v_1$
for *each bus* **do**
    if *voltage got closer to the target* **then**
        $r = r + v_2$
    else if *voltage got further from the target* **then**
        $r = r - v_3$
    else if *the action chosen was different from "do nothing"* **then**
        $r = r - v_4$
    else if *voltage is at* $\pm 1\%$ *from the target* **then**
        $r = r + v_5$
    else if *voltage violates upper or lower limits* **then**
        $r = r - v_6$
    else
        $r = r + v_7$

5
Computational Experiments and Results

In this chapter the proposed techniques will be tested on simulated power systems. The utilized systems are the 13, 37 and 123 bus IEEE test circuits (28). For each system, four models were trained: a pure reinforcement learning model, a windowed model, a ensemble model and a windowed ensemble model. Additionally, for comparison effect the system was simulated without any form of control and also with the control capabilities available at the simulation software (OpenDSS). The topology of these systems is shown on Figure 5.1.

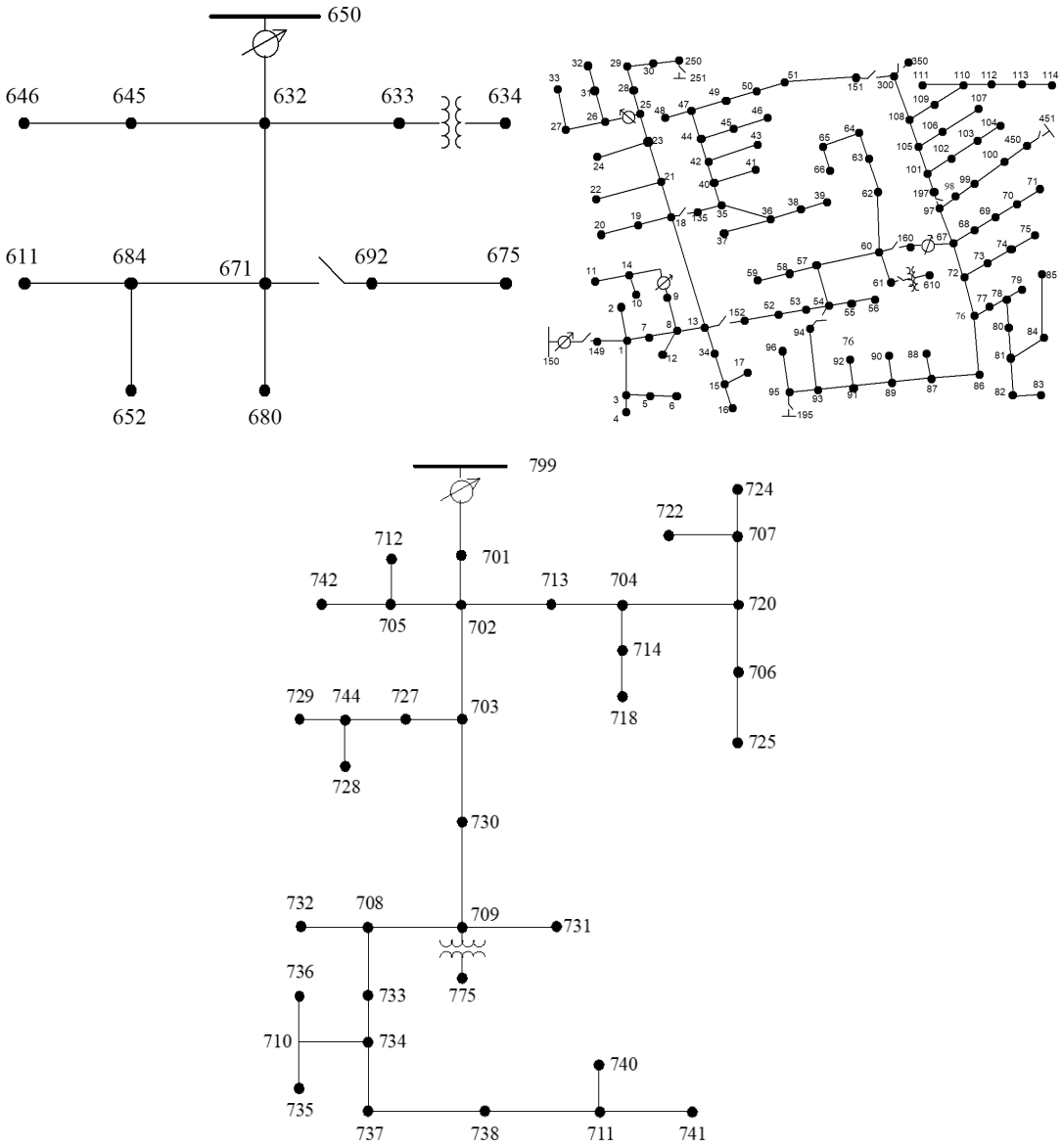

Figure 5.1: IEEE Circuits: (a) 13-bus; (b) 123-bus; (c) 37-bus;

5.1 Training

For every system, the parameters used during training were very similar with the exception of the number of episodes which was 100 for both the 13 and 123 bus systems and 200 for the 37 bus system. Regarding the parameters shown on Algorithm 2, the used memory size N was 1024, the minimum number of samples to start training was 512 and the batch size, 256. The discount factor γ was 0.9. For training the target network, the method on Equation 4-5 was used. For epsilon, instead of keeping it static, a decay technique was used. That way, epsilon decays linearly with the episodes from 1.0 to a minimum of 0.3. The neural network structure is shown on Figure 5.2 and its learning rate α was 0.001. The chosen architecture is quite simple when considering its size and activation functions. This is due to the nature of deep reinforcement learning which does not require complex structures in order to predict the action values and also because of the tested systems' sizes. Its input and output layer sizes depends on the system's characteristics, that is the size of its state and the number of available actions, both of which depends on the number of capacitors and tap changing transformers that can be controlled. This data is shown on Table 5.1. Finally, for the rewards as shown in Algorithm 3, the values are presented in Table 5.2. The upper and lower voltage limits considered are 1.05 and 0.92 p.u. (2) while the target is 1.0 p.u. An exception is the ensemble approach where two alternative agents with more rigorous limits are used. The first alternative agent uses 1.03 and 0.95 p.u. for the upper and lower limits while the second uses 1.02 and 0.98 p.u. Also, for the ensemble agents the "vote" on the best action is conducted by following Equation 4-6.

Table 5.1: Test Circuits' State and Action Space Sizes

	State Size	nº of Actions
13-bus	10	15
37-bus	7	9
123-bus	15	25

The training process was conducted with the parameters and iterations described above. The neural network loss as well as the accumulated reward resultant from the training process for each circuit is shown on Figures 5.3, 5.4, 5.5, 5.6 for the 13-bus system, Figures 5.7, 5.8, 5.9, 5.10 for the 37-bus system and Figures 5.11, 5.12, 5.13, 5.14 for the 123-bus. In deep Q-Learning the network loss not necessarily converges to zero but it must show a converging behavior to any value. The most important part is that the accumulate reward

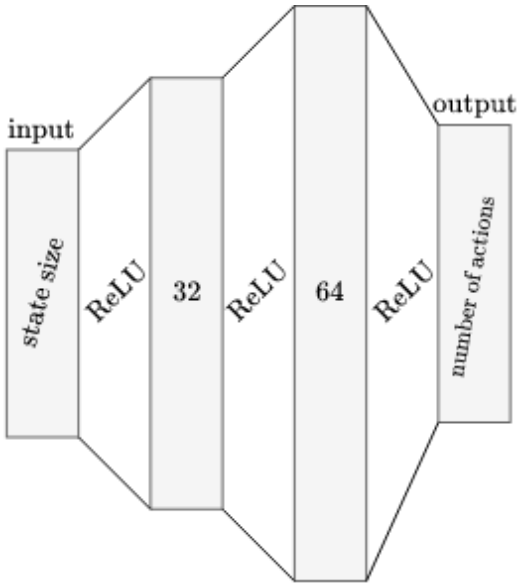

Figure 5.2: Neural Network Structure

Table 5.2: Reward Values

	Value
v_1	-1
v_2	0.7
v_3	-0.8
v_4	-0.8
v_5	1
v_6	-1
v_7	0.2

gets bigger (and positive) as the training progresses. The training time is shown on Table 5.3.

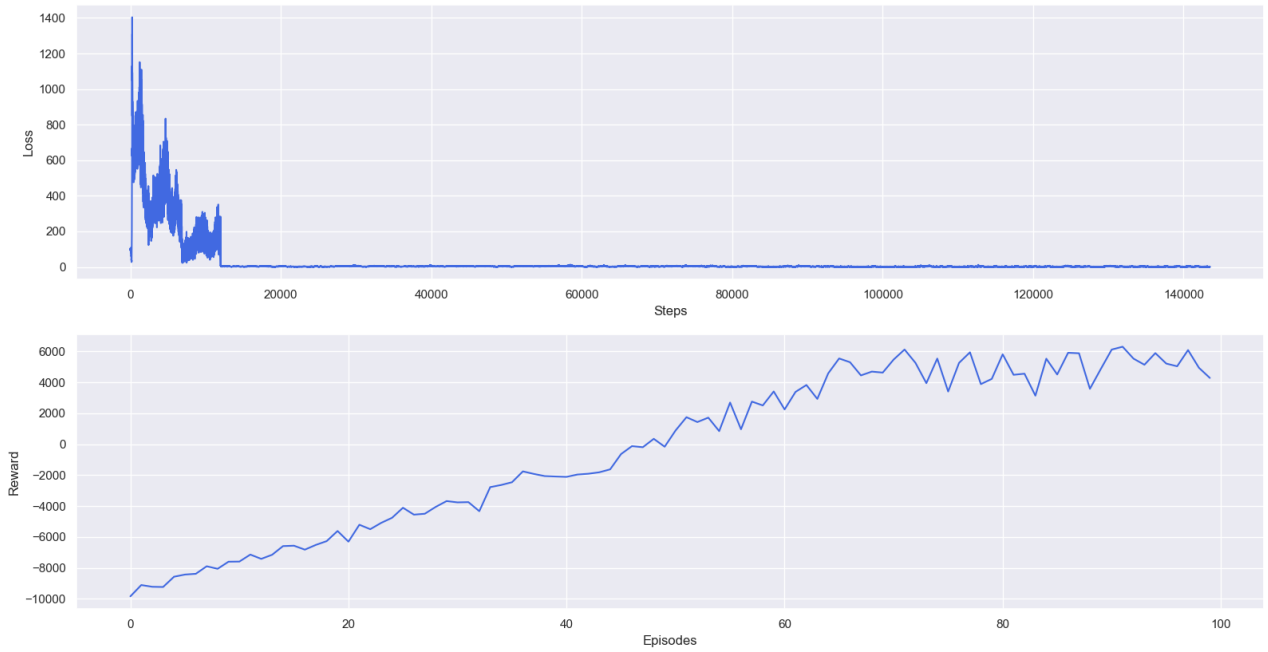

Figure 5.3: IEEE 13-bus network loss (above) and accumulated reward (below) (Reinforcement Learning)

Figure 5.4: IEEE 13-bus network loss (above) and accumulated reward (below) (Windowed Reinforcement Learning)

Figure 5.5: IEEE 13-bus network loss (above) and accumulated reward (below) (Ensemble Reinforcement Learning)

Figure 5.6: IEEE 13-bus network loss (above) and accumulated reward (below) (Windowed Ensemble Reinforcement Learning)

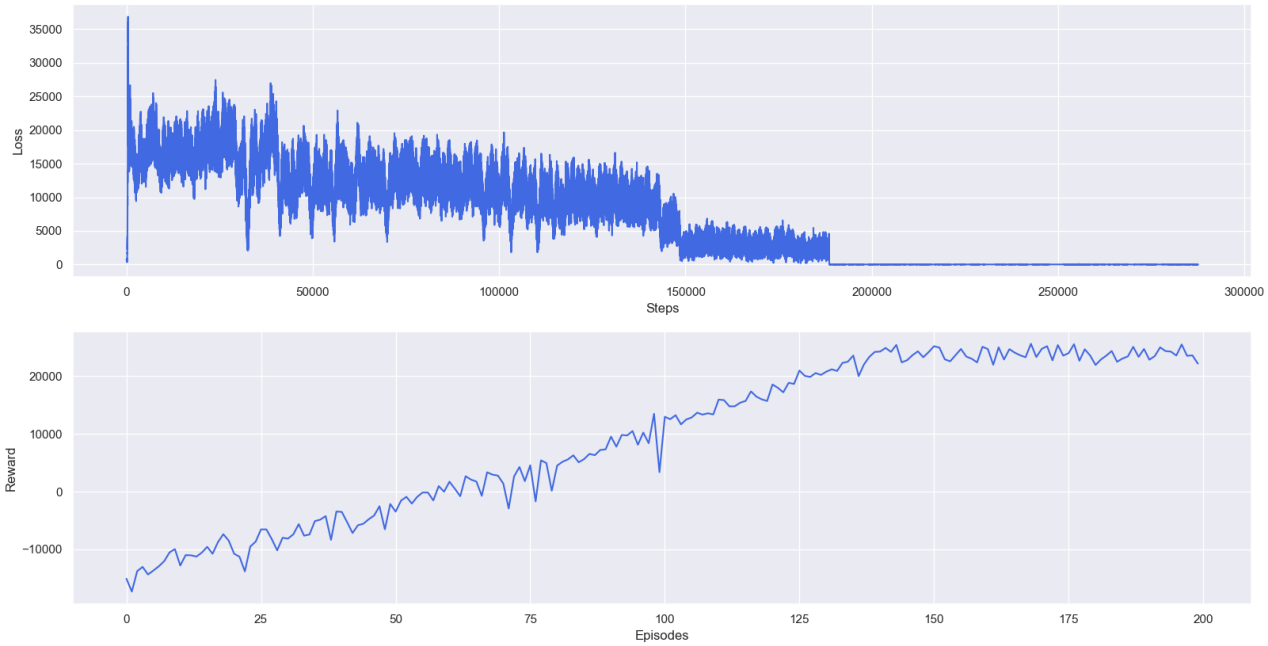

Figure 5.7: IEEE 37-bus network loss (above) and accumulated reward (below) (Reinforcement Learning)

Figure 5.8: IEEE 37-bus network loss (above) and accumulated reward (below) (Windowed Reinforcement Learning)

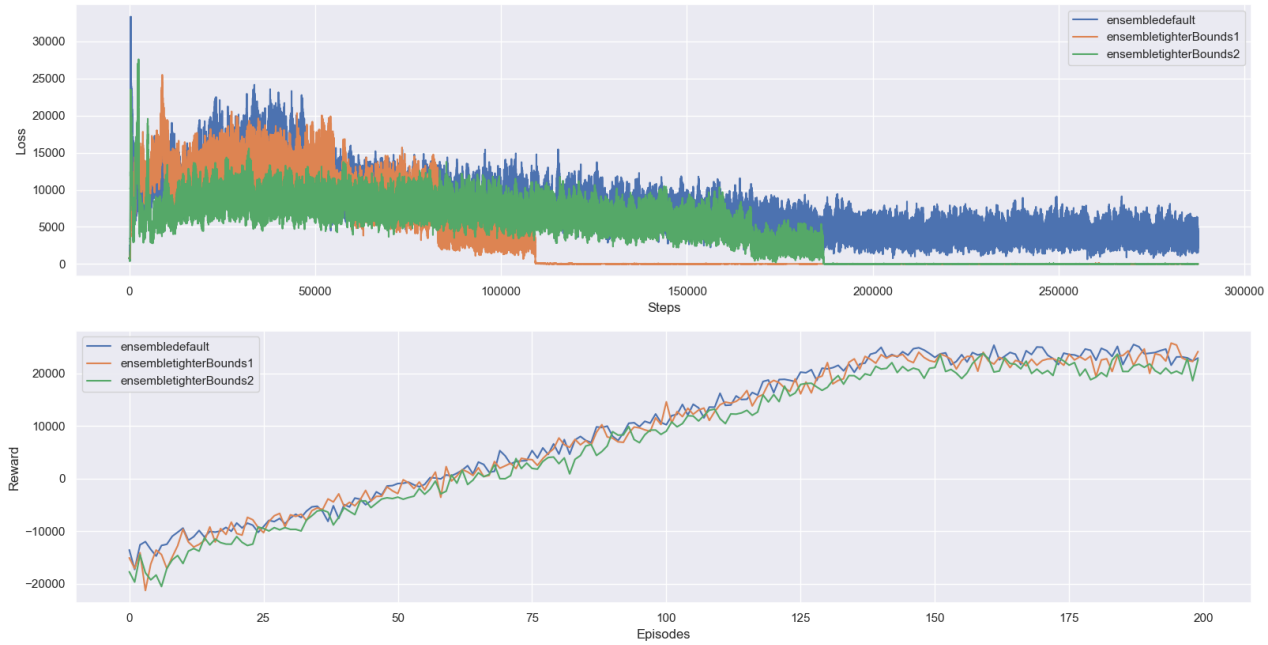

Figure 5.9: IEEE 37-bus network loss (above) and accumulated reward (below) (Ensemble Reinforcement Learning)

Figure 5.10: IEEE 37-bus network loss (above) and accumulated reward (below) (Windowed Ensemble Reinforcement Learning)

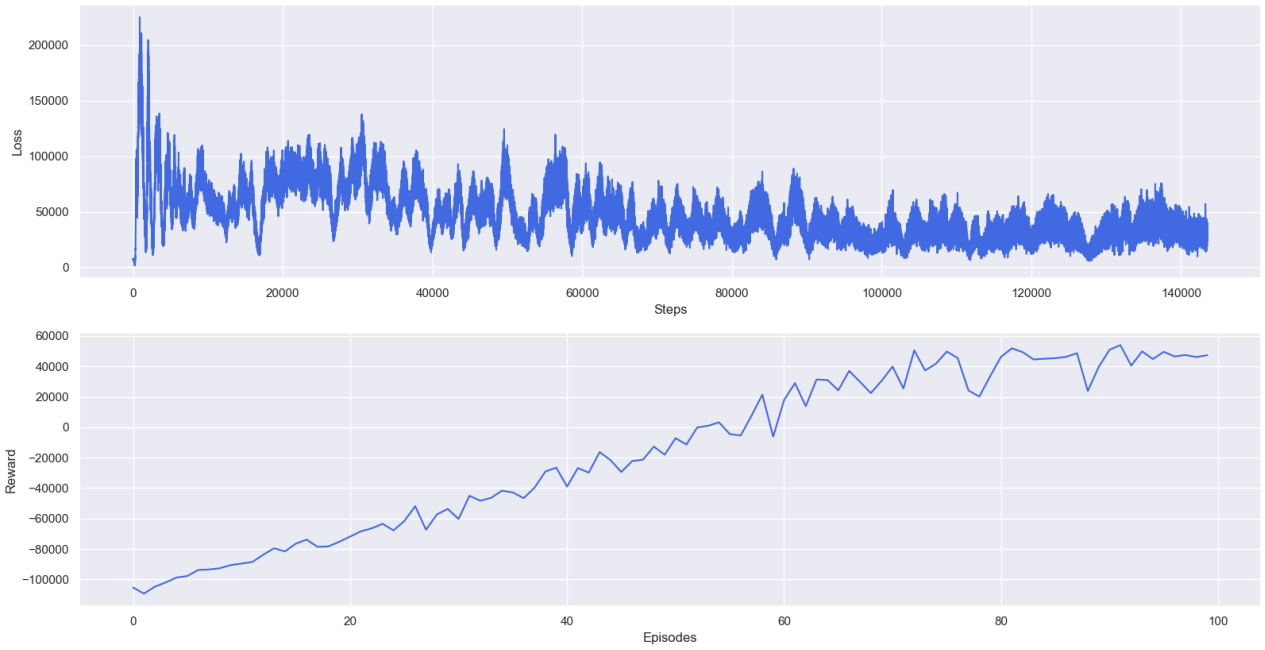

Figure 5.11: IEEE 123-bus network loss (above) and accumulated reward (below) (Reinforcement Learning)

Figure 5.12: IEEE 123-bus network loss (above) and accumulated reward (below) (Windowed Reinforcement Learning)

Figure 5.13: IEEE 123-bus network loss (above) and accumulated reward (below) (Ensemble Reinforcement Learning)

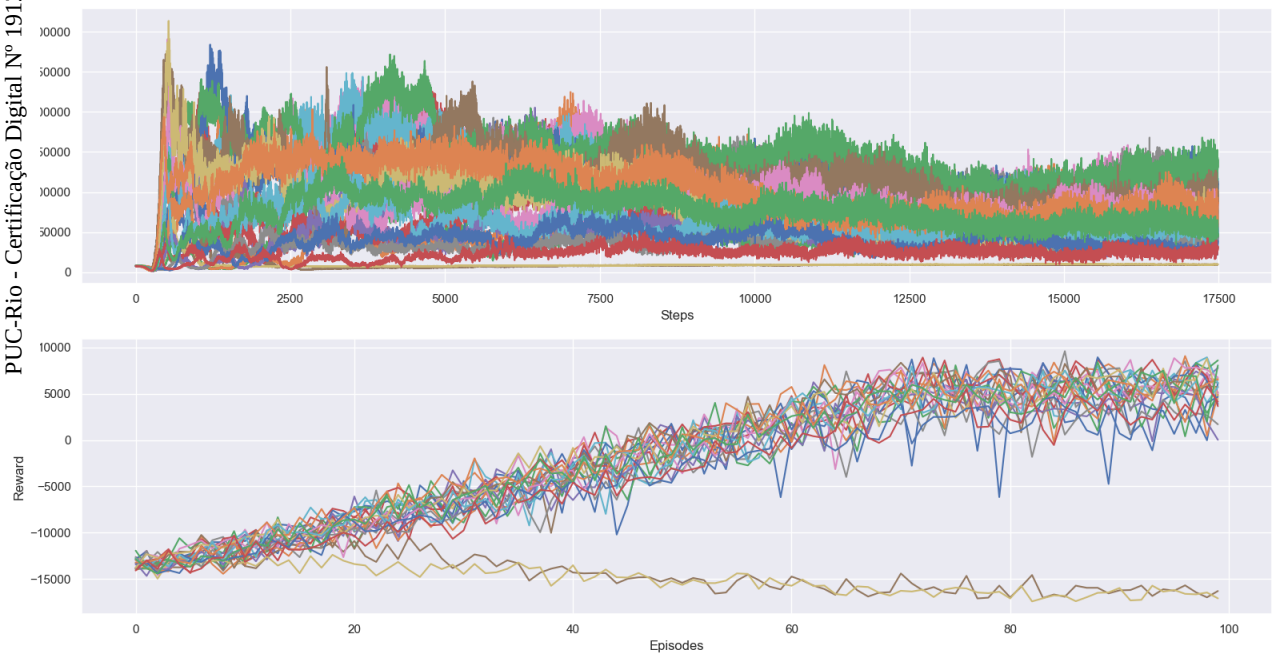

Figure 5.14: IEEE 123-bus network loss (above) and accumulated reward (below) (Windowed Ensemble Reinforcement Learning)

5.2

Table 5.3: Agents' Training Time in Seconds

	13-bus	37-bus	123-bus
Reinforcement Learning	1169.2	4167.5	4903.1
Windowed	1166.5	3343.2	3038.3
Ensemble	3604.2	14524.2	11541.2
Windowed Ensemble	3580.5	9999.9	26908.2

Results

After the training procedure, the agents were tested on the same systems on different days. The days were simulated by randomly choosing a load profile and introducing some random noise into the loads. As stated in the previous section, the agents' performance is compared to the systems' without any form of control, with the control present on the system itself and also with a "pure" deep reinforcement learning approach as proposed by (20, 21, 12, 31) and implemented, with slight variations, by (34), (7), (32), (35) and (30).

In order to obtain the results, a total of 50 days for each technique and system were simulated. A few metrics are used to show the achieved results (Tables 5.4, 5.5 and 5.6):

- the total number of violations (the number of times the voltage surpassed the limits of 1.05 and 0.92 p.u.);
- the maximum and minimum voltage achieved across the 50 days;
- the average real power loss across the 50 days;
- the number of actions the agents took.

For the number of actions, the values are only available for the reinforcement learning techniques, since for the no control no actions are taken and for the system control method, it is not possible to obtain this value from the OpenDSS simulator. Besides the 50 days simulated in order to obtain the metrics described above, a single day was simulated separately in order to closely observe how the systems' voltages behave during this period for each of the proposed techniques. These results are shown on Figures 5.15, 5.18 and 5.21 which show the average voltage (across all buses) on the system as the day progresses and on Figures 5.16, 5.19 and 5.22 which show the voltages on each bus at every moment of the day. Also, Figures 5.17, 5.20 and 5.23 show the voltages at each bus, at each moment of the day. The green areas mean that the voltage at that point is within 1% of the desired target (1 p.u.) whereas the blue areas mean that the voltages are more than 1% away from the target.

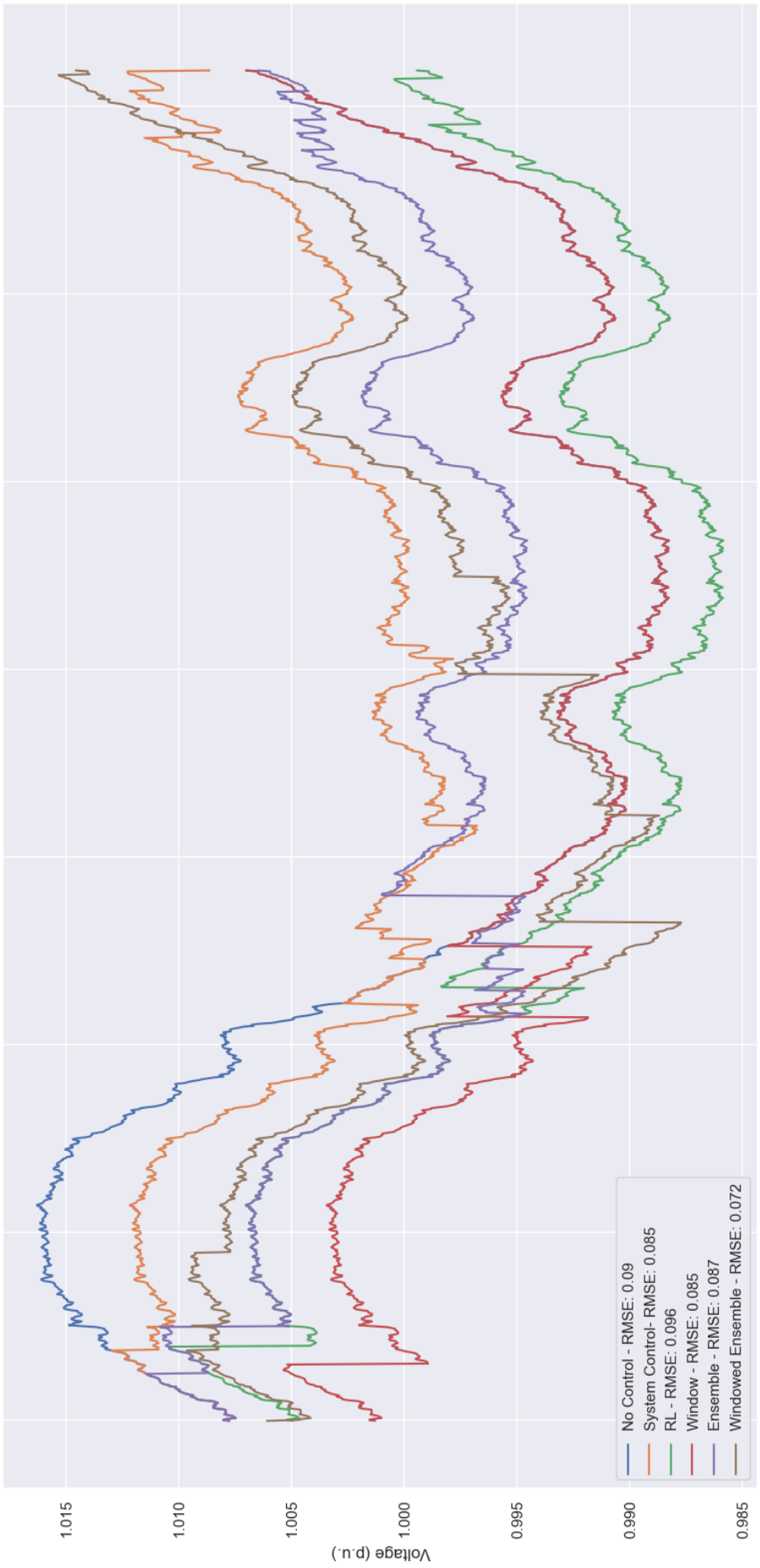

Figure 5.15: IEEE 13-bus single-day results: Average system voltage during the day

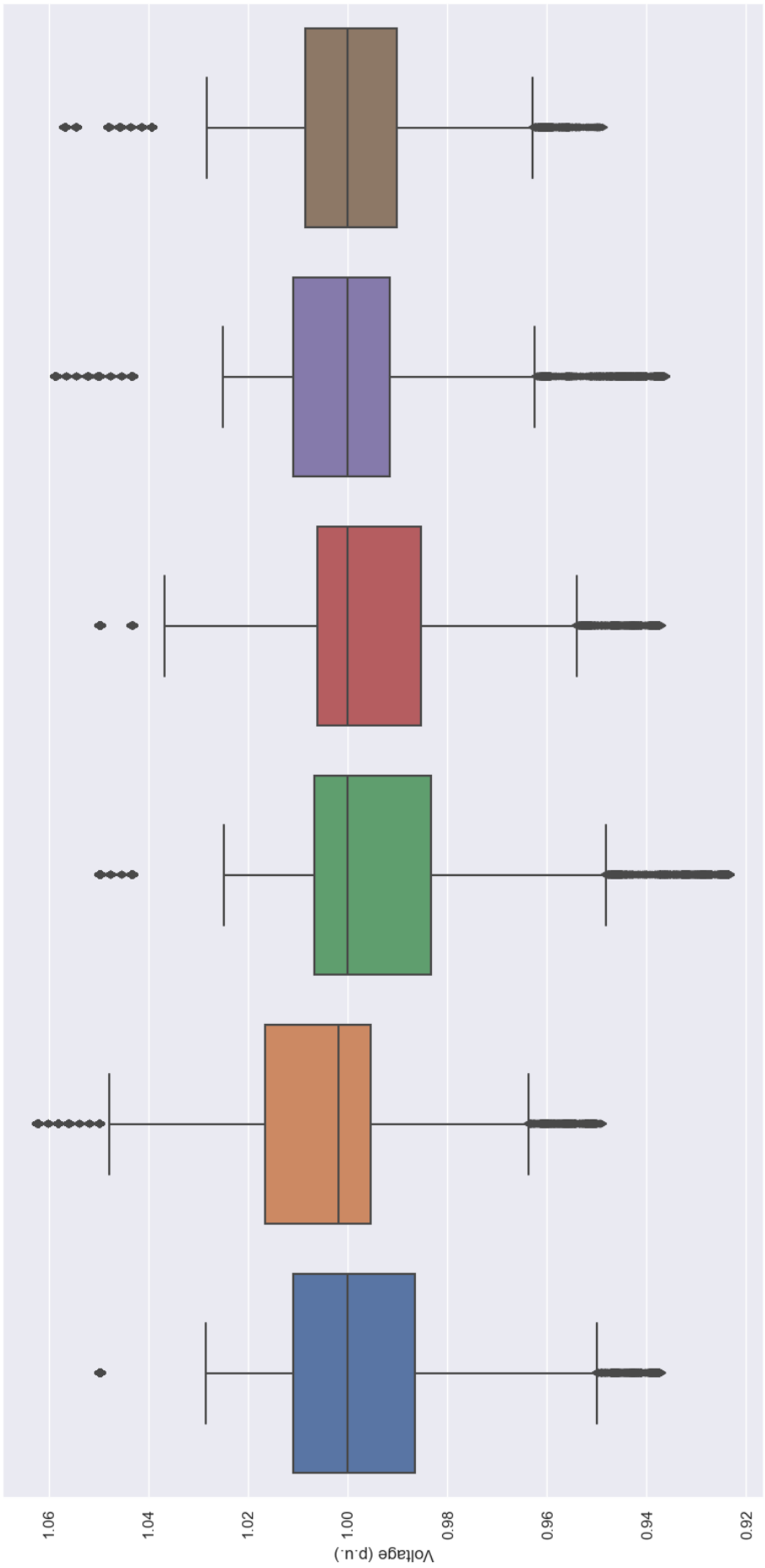

Figure 5.16: IEEE 13-bus single-day results: Voltage distribution during the day

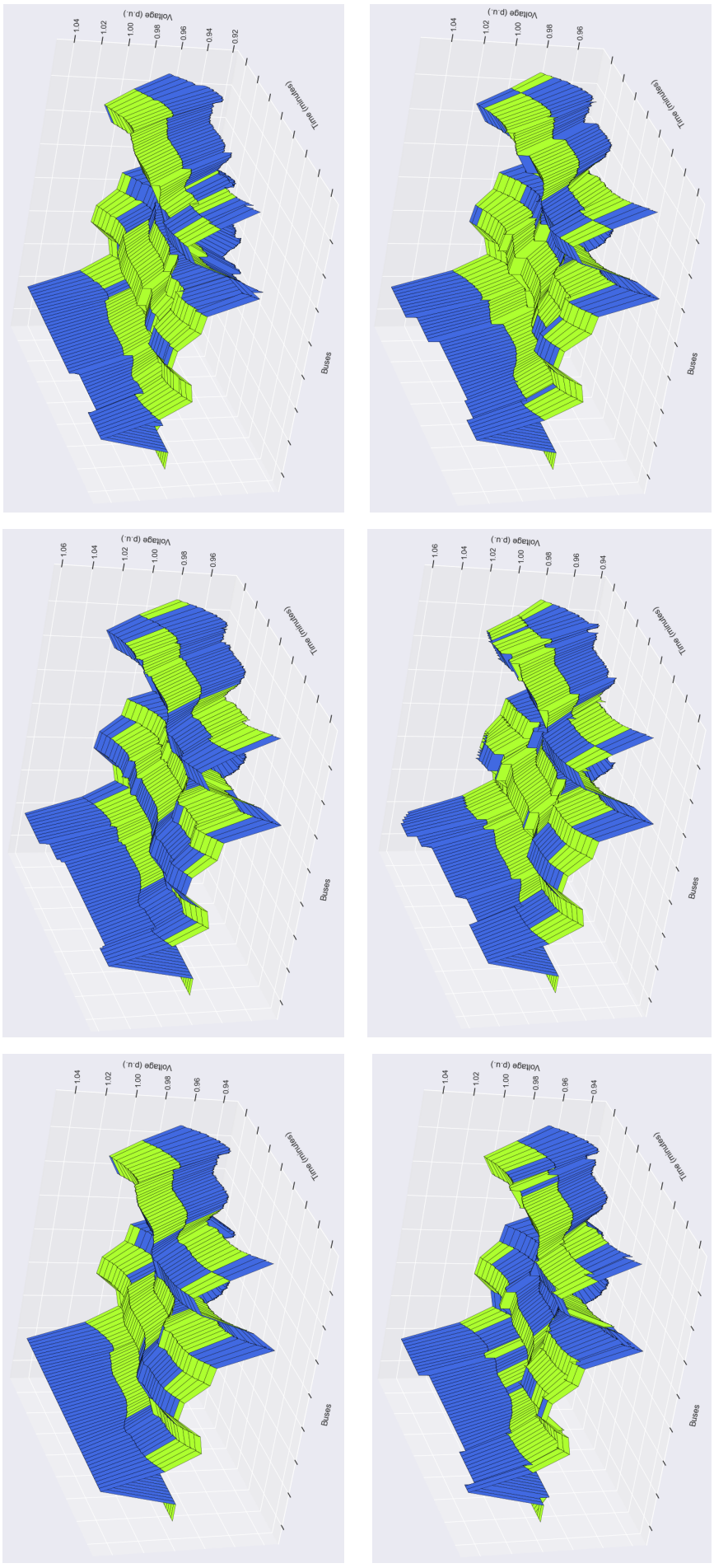

Figure 5.17: IEEE 13-bus single-day voltage profile: (a) No Control; (b) System Control; (c) Reinforcement Learning; (d) Windowed; (e) Ensemble; (f) Windowed Ensemble;

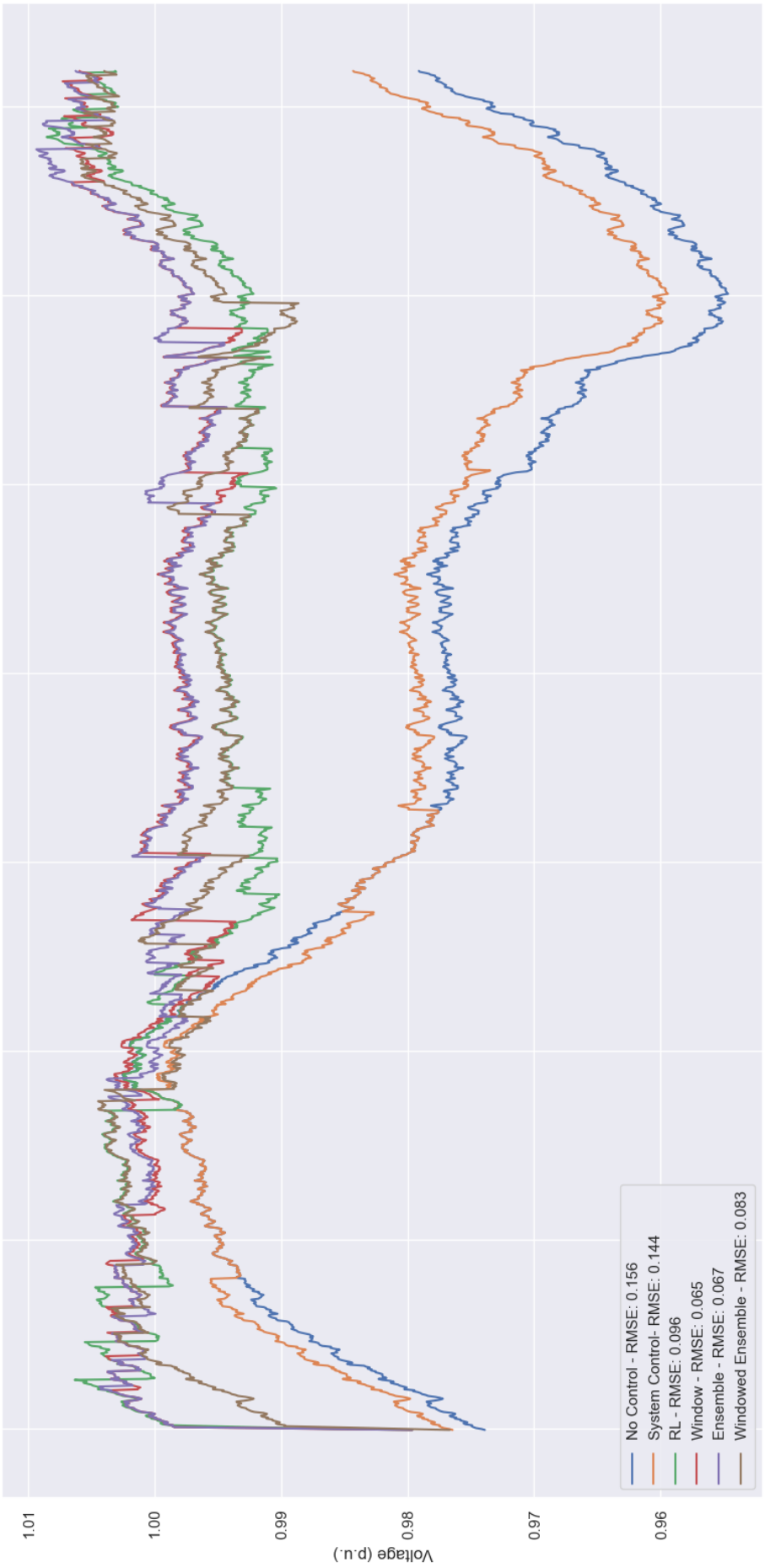

Figure 5.18: IEEE 37-bus single-day results: Average system voltage during the day

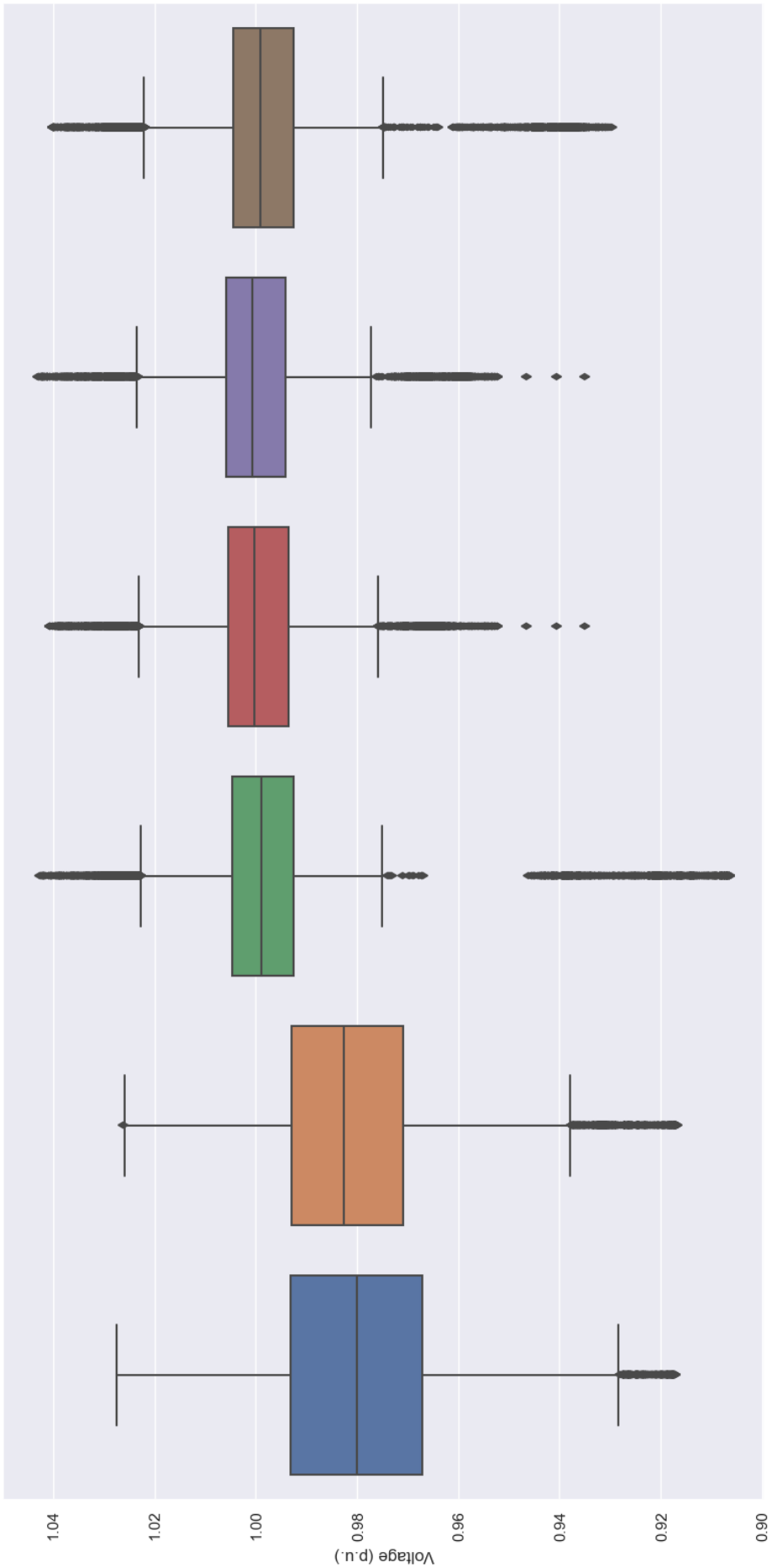

Figure 5.19: IEEE 37-bus single-day results: Voltage distribution during the day

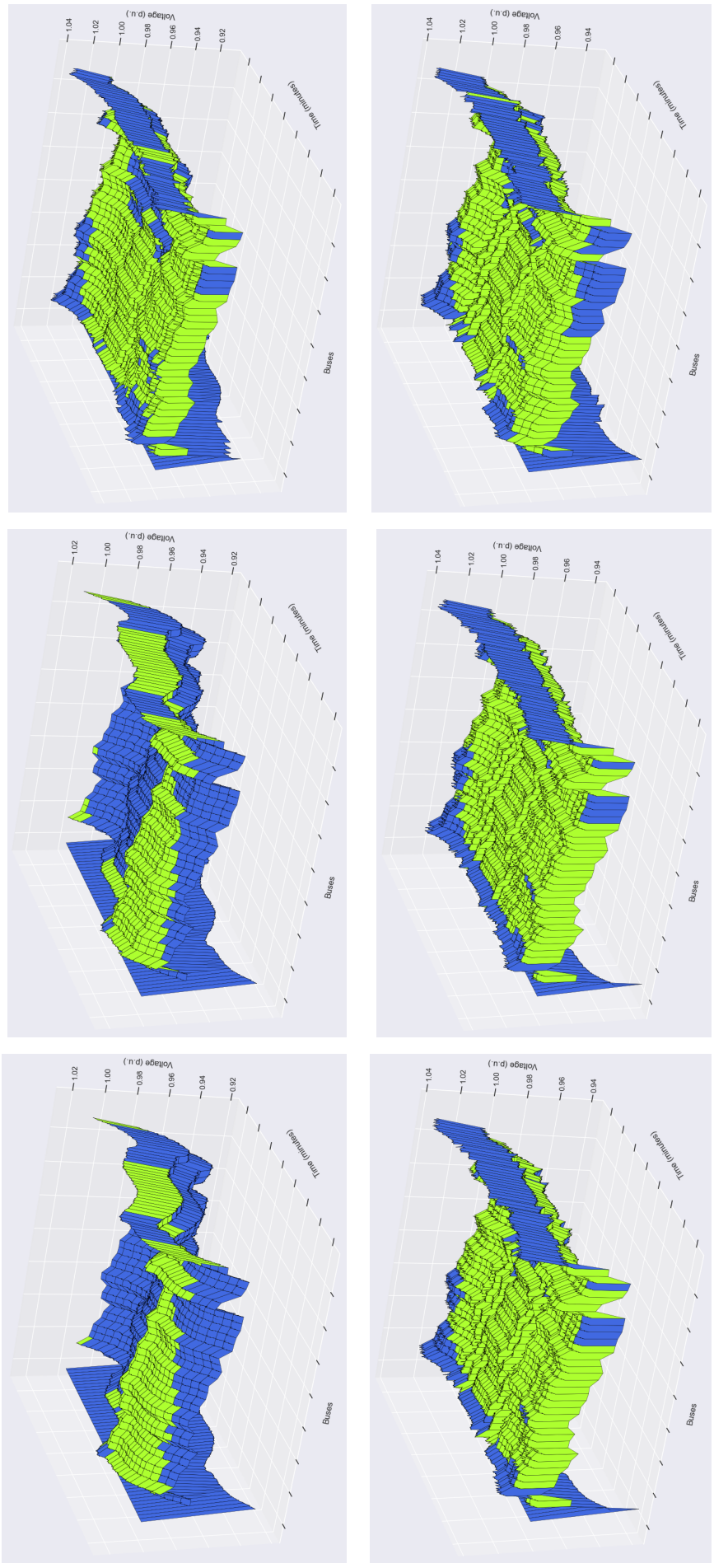

Figure 5.20: IEEE 37-bus single-day voltage profile: (a) No Control; (b) System Control; (c) Reinforcement Learning; (d) Windowed; (e) Ensemble; (f) Windowed Ensemble;

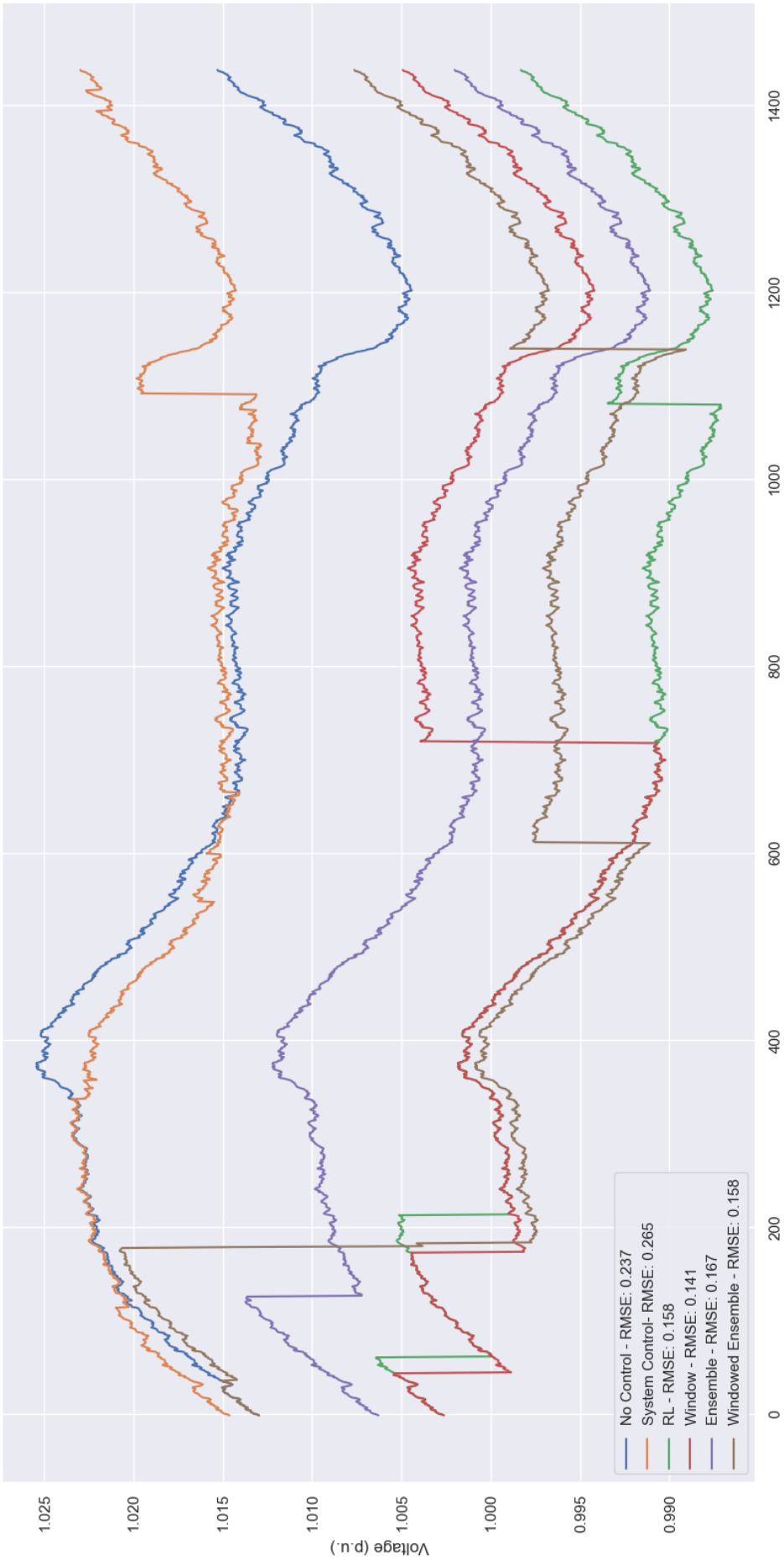

Figure 5.21: IEEE 123-bus single-day results: Average system voltage during the day

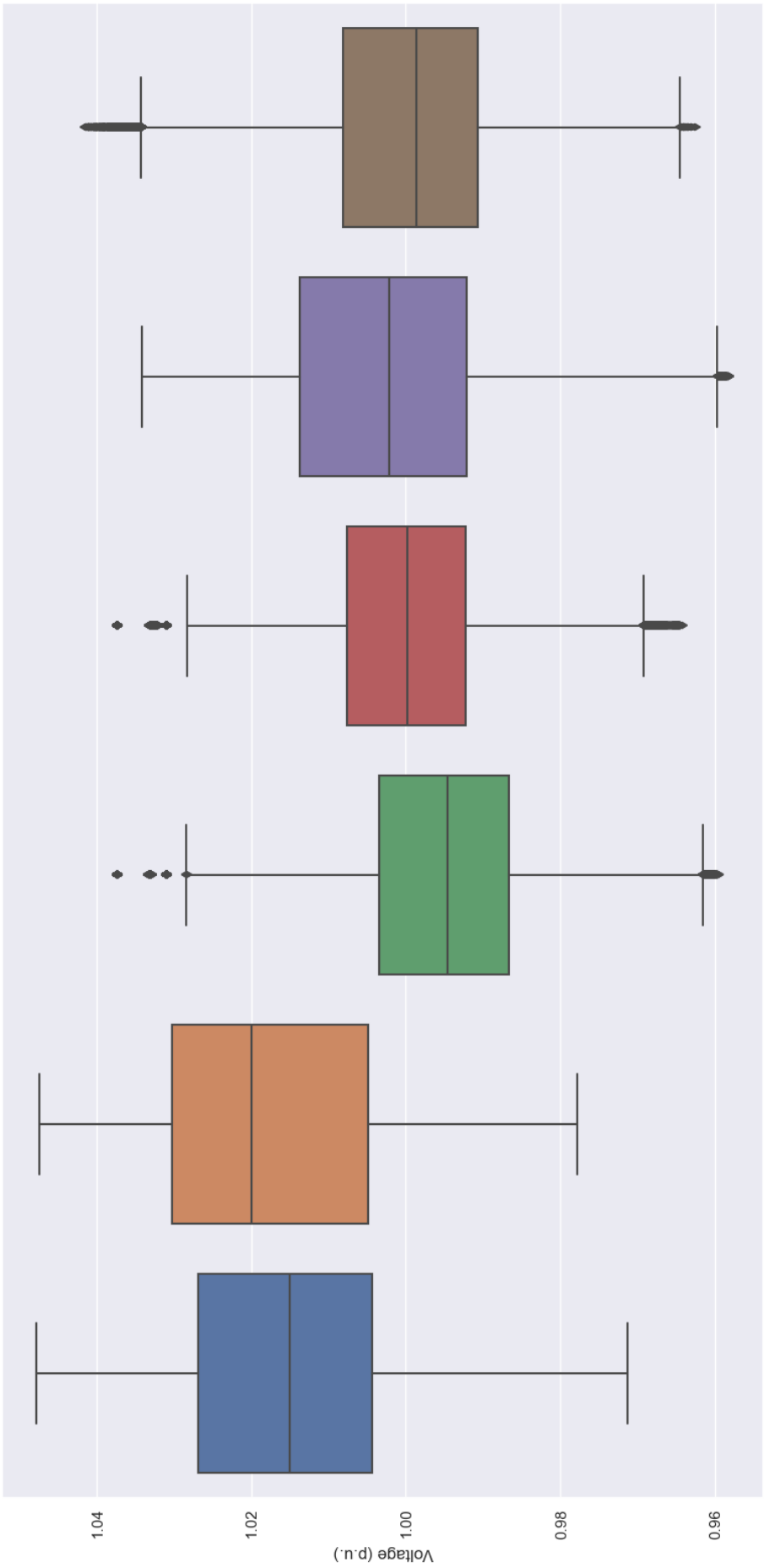

Figure 5.22: IEEE 123-bus single-day results: Voltage distribution during the day

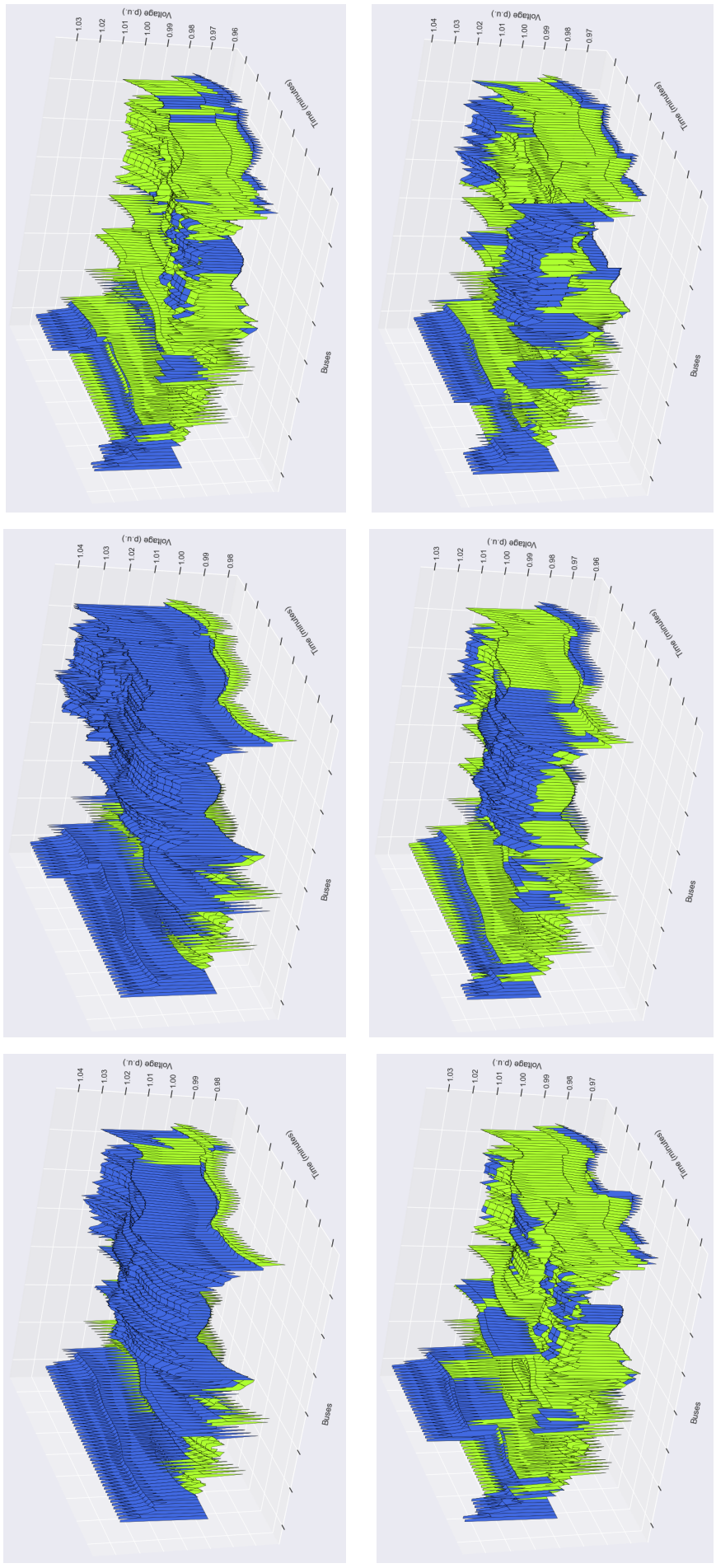

Figure 5.23: IEEE 123-bus single-day voltage profile: (a) No Control; (b) System Control; (c) Reinforcement Learning; (d) Windowed; (e) Ensemble; (f) Windowed Ensemble;

Table 5.4: 13-bus System Results

	Violations	Max. Voltage	Min. Voltage	Avg. Losses (kW)	Avg. Actions
No Control	0	1.049	0.925	115294.8	not applicable
System Control	68201	1.066	0.940	114300.1	not applicable
Reinforcement Learning	3552	1.052	0.912	117684.6	3.5
Windowed	0	1.050	0.926	116669.5	3.6
Ensemble	21789	1.062	0.934	115054.9	6.4
Windowed Ensemble	33434	1.059	0.944	114875.3	11.3

Table 5.5: 37-bus System Results

	Violations	Max. Voltage	Min. Voltage	Avg. Losses (kW)	Avg. Actions
No Control	66	1.034	0.919	154368.0	not applicable
System Control	326	1.031	0.919	154992.1	not applicable
Reinforcement Learning	31026	1.045	0.899	155482.1	29.0
Windowed	0	1.042	0.935	155700.8	24.5
Ensemble	0	1.042	0.932	154746.1	23.1
Windowed Ensemble	0	1.040	0.942	158196.7	20.3

Table 5.6: 123-bus System Results

	Violations	Max. Voltage	Min. Voltage	Avg. Losses (kW)	Avg. Actions
No Control	0	1.048	0.969	97619.3	not applicable
System Control	0	1.050	0.976	97532.7	not applicable
Reinforcement Learning	0	1.037	0.958	109314.9	4.8
Windowed	0	1.037	0.961	109851.9	7.5
Ensemble	0	1.034	0.958	97665.4	2
Windowed Ensemble	105	1.051	0.959	112275.1	7.4

Regarding the training process, all agents were satisfactorily trained as is shown on Figures 5.3 to 5.14 by the convergence of the network loss and positive accumulated reward. The training times were varied but the *windowed* methodology has shown a reduced training time even when compared to the traditional *reinforcement learning* approach.

The results indicate that the proposed techniques are capable of controlling the voltage on power systems.

For the 13-bus system, because it's already a fairly balanced and small system, the results were marginal. Figures 5.15 and 5.16 show that the average voltage has changed very little with the proposed methodologies although with slightly lower peaks and Figure 5.17 presents a somewhat more balanced voltage profile (more green areas on the chart), which is validated by Figure 5.16 (means closer to 1 p.u.) for the proposed methodologies. Nevertheless, the *windowed* methodology was capable of removing all voltage violations

when compared to the control already present on the system while keeping the number of required actions low and not affecting the real power losses significantly (Table 5.4).

For the 37-bus system, while the pure *reinforcement learning* approach actually increased the number of voltage violations, the other three proposed methodologies completely eliminated them. For all the methodologies, the average losses were kept fairly close to its original values and the *windowed ensemble* methodology has executed the task in the least number of actions (Table 5.5). On Figures 5.18 and 5.19 it is possible to see that for the proposed methodologies the voltages are much closer to the target of 1 p.u. and with higher valleys. Figure 5.19 shows a significantly tighter spread for the voltages. Also, the voltage profile in general is much more balanced and closer to $\pm 1\%$ of the target, as shown by the increased number of green areas on Figure 5.20 and by a mean closer to 1 p.u. on Figure 5.19.

Finally, for the 123-bus system, while there were no violations on the system both with and without control, the *windowed ensemble* methodology introduced some, although minimal violations. For this system, the average losses were increased by a significant margin by the three of the four proposed methodologies, with the *ensemble* technique being the exception (Table 5.6). Figures 5.21 and 5.22 show that the average system voltage was kept close to 1 p.u. for the proposed methodologies. Figure 5.23 shows that the voltage profile for the proposed methodologies was improved since the number of green areas was increased and the means on Figure 5.22 are close to 1 p.u. In this system, the *ensemble* technique, has shown a better performance overall, controlling the voltage satisfactorily while also keeping the losses at a better acceptable value and the number of actions at a minimum.

6
Conclusion

Reinforcement Learning has been around for some time and has shown great results across many different scientific and real-world problems. When combined with the power of deep neural networks, deep Q-Learning can tackle a plethora of problems. In this work, a deep Q-Learning methodology was proposed to solve the voltage control problem on electrical power systems. Besides the regular reinforcement learning approach, three other novel methodologies were proposed with the goal of improving the technique's performance on this specific problem. The results show that the application of the techniques was successful and has shown great value when compared to the traditional DRL approach and also with the systems' own control. The trained intelligent agents are capable of controlling the system voltage in a completely autonomous way while keeping the number of actions taken low and having little effect on the real power losses.

6.1
Future Work

As future improvements that can be made on the methodology and technique, the following are suggested:

- Include other type of equipment besides capacitors and transformers;
- Examine the effect of changing the state representation;
- Parallelize the agent's training process;
- Test other reward functions for the ensemble methodology;
- Test the effect the number of windows has on the agent;
- Train for systems with a high number of faults and defects;
- Train the agent to control the equipment while also letting the system's automatic control equipment actuate.

7
References

[1] AL-AMERI, A.. **Méthodes analytiques d'étude pour la diminution des pertes de puissance dans les réseaux électriques maillés en utilisant des techniques d'optimisation pour le dimensionnement et l'emplacement des générateurs décentralisés**. PhD thesis, 04 2017. (document), 1.1

[2] ANEEL. **Procedimentos de distribuição de energia elétrica no sistema elétrico nacional - prodist - módulo 8 - qualidade da energia elétrica.** 5.1

[3] BARAN, M.; MING-YUNG HSU. **Volt/VAr control at distribution substations**. IEEE Transactions on Power Systems, 14(1):312–318, Feb. 1999. 3

[4] BORGHETTI, A.. **Using mixed integer programming for the volt/-var optimization in distribution feeders**. Electric Power Systems Research, 98:39–50, May 2013. 3

[5] BOROZAN, V.; BARAN, M. ; NOVOSEL, D.. **Integrated volt/VAr control in distribution systems**. In: 2001 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.01CH37194), volumen 3, p. 1485–1490, Columbus, OH, USA, 2001. IEEE. 3

[6] CORSI, S.. **Voltage control and protection in electrical power systems from system components to wide-area control**. Springer London : Imprint : Springer, London, 2015. OCLC: 925511668. (document), 1, 2, 2.1, 2.1

[7] DIAO, R.; WANG, Z.; SHI, D.; CHANG, Q.; DUAN, J. ; ZHANG, X.. **Autonomous voltage control for grid operation using deep reinforcement learning**. 3, 5.2

[8] FRANCO, J. F.; RIDER, M. J.; LAVORATO, M. ; ROMERO, R.. **A mixed-integer LP model for the optimal allocation of voltage regulators and capacitors in radial distribution systems**. International Journal of Electrical Power & Energy Systems, 48:123–130, June 2013. 3

[9] GALLEGO, R.; MONTICELLI, A. ; ROMERO, R.. **Optimal capacitor placement in radial distribution networks**. IEEE Transactions on Power Systems, 16(4):630–637, Nov. 2001. 3

[10] GRAINGER, J. J.; STEVENSON, W. D. ; STEVENSON, W. D.. **Power system analysis**. McGraw-Hill series in electrical and computer engineering. McGraw-Hill, New York, 1994. 2

[11] GU, Z.; RIZY, D.. **Neural networks for combined control of capacitor banks and voltage regulators in distribution systems**. IEEE Transactions on Power Delivery, 11(4):1921–1928, Oct. 1996. 3

[12] HESSEL, M.; MODAYIL, J.; VAN HASSELT, H.; SCHAUL, T.; OSTROVSKI, G.; DABNEY, W.; HORGAN, D.; PIOT, B.; AZAR, M. ; SILVER, D.. **Rainbow: Combining improvements in deep reinforcement learning**. 4.2, 4.2.2, 5.2

[13] HOMAEE, O.; ZAKARIAZADEH, A. ; JADID, S.. **Real-time voltage control algorithm with switched capacitors in smart distribution system in presence of renewable generations**. 54:187–197. 3

[14] HU, Z.; WANG, X.; CHEN, H. ; TAYLOR, G.. **Voltvar control in distribution systems using a time-interval based approach**. 150(5):548. 3

[15] KHANABADI, M.; GHASEMI, H. ; DOOSTIZADEH, M.. **Optimal Transmission Switching Considering Voltage Security and N-1 Contingency Analysis**. IEEE Transactions on Power Systems, 28(1):542–550, Feb. 2013. 3

[16] LEVITIN, G.; KALYUZHNY, A.; SHENKMAN, A. ; CHERTKOV, M.. **Optimal capacitor allocation in distribution systems using a genetic algorithm and a fast energy loss computation technique**. IEEE Transactions on Power Delivery, 15(2):623–628, Apr. 2000. 3

[17] LIANG, R.-H.; CHEN, Y.-K. ; CHEN, Y.-T.. **Volt/Var control in a distribution system by a fuzzy optimization approach**. International Journal of Electrical Power & Energy Systems, 33(2):278–287, Feb. 2011. 3

[18] LIU, Y.; ZHANG, P. ; QIU, X.. **Optimal volt/var control in distribution systems**. International Journal of Electrical Power & Energy Systems, 24(4):271–276, May 2002. 3

[19] MEIER, A. V.. **Electric power systems: a conceptual introduction**. Wiley survival guides in engineering and science. IEEE Press : Wiley-Interscience, Hoboken, N.J, 2006. OCLC: ocm62616191. 1

[20] MNIH, V.; KAVUKCUOGLU, K.; SILVER, D.; GRAVES, A.; ANTONOGLOU, I.; WIERSTRA, D. ; RIEDMILLER, M.. **Playing atari with deep reinforcement learning**. 4.2, 5.2

[21] MNIH, V.; KAVUKCUOGLU, K.; SILVER, D.; RUSU, A. A.; VENESS, J.; BELLEMARE, M. G.; GRAVES, A.; RIEDMILLER, M.; FIDJELAND, A. K.; OSTROVSKI, G.; PETERSEN, S.; BEATTIE, C.; SADIK, A.; ANTONOGLOU, I.; KING, H.; KUMARAN, D.; WIERSTRA, D.; LEGG, S. ; HASSABIS, D.. **Human-level control through deep reinforcement learning**. 518(7540):529–533. 4.2, 4.2.1, 5.2

[22] NIKNAM, T.. **A new approach based on ant colony optimization for daily Volt/Var control in distribution networks considering distributed generators**. Energy Conversion and Management, 49(12):3417–3424, Dec. 2008. 3

[23] NIKNAM, T.; FIROUZI, B. B. ; OSTADI, A.. **A new fuzzy adaptive particle swarm optimization for daily Volt/Var control in distribution networks considering distributed generators**. Applied Energy, 87(6):1919–1928, June 2010. 3

[24] Rashid, M. H., editor. **Electric renewable energy systems**. Elsevier/AP, Academic Press is an imprint of Elsevier, Amsterdam, 2016. 1

[25] RIBEIRO, L. C.; SCHUMANN MINAMI, J. P. O.; BONATTO, B. D.; RIBEIRO, P. F. ; DE SOUZA, A. C. Z.. **Voltage control simulations in distribution systems with high penetration of PVs using the OpenDSS**. In: 2018 SIMPOSIO BRASILEIRO DE SISTEMAS ELETRICOS (SBSE), p. 1–6. IEEE. 3

[26] ROYTELMAN, I.; WEE, B. ; LUGTU, R.. **Volt/var control algorithm for modern distribution management system**. IEEE Transactions on Power Systems, 10(3):1454–1460, Aug. 1995. 3

[27] SARIC, A. T.; STANKOVIC, A. M.. **A robust algorithm for Volt/Var control**. In: 2009 IEEE/PES Power Systems Conference AND Exposition, p. 1–8, Seattle, WA, USA, Mar. 2009. IEEE. 3

[28] SCHNEIDER, K. P.; MATHER, B. A.; PAL, B. C.; TEN, C.-W.; SHIREK, G. J.; ZHU, H.; FULLER, J. C.; PEREIRA, J. L. R.; OCHOA, L. F.; DE ARAUJO, L. R.; DUGAN, R. C.; MATTHIAS, S.; PAUDYAL, S.; MCDERMOTT, T. E. ; KERSTING, W.. **Analytic Considerations and**

Design Basis for the IEEE Distribution Test Feeders. IEEE Transactions on Power Systems, 33(3):3181–3188, May 2018. 5

[29] SUTTON, R. S.; BARTO, A. G.. **Reinforcement learning: an introduction**. Adaptive computation and machine learning series. The MIT Press, Cambridge, Massachusetts, second edition edition, 2018. (document), 4.1, 4.1

[30] CUSTÓDIO, G.; OCHOA, L.; TRINDADE, F. ; ALPCAN, T.. **Using q-learning for oltc voltage regulation in pv-rich distribution networks**. 11 2020. 3, 4.3, 5.2

[31] VAN HASSELT, H.; GUEZ, A. ; SILVER, D.. **Deep reinforcement learning with double q-learning**. 4.2.2, 5.2

[32] VLACHOGIANNIS, J.; HATZIARGYRIOU, N.. **Reinforcement Learning for Reactive Power Control**. IEEE Transactions on Power Systems, 19(3):1317–1325, Aug. 2004. 3, 5.2

[33] WOOD, A. J.; WOLLENBERG, B. F. ; SHEBLÉ, G. B.. **Power generation, operation, and control**. Wiley-IEEE, Hoboken, New Jersey, third edition edition, 2013. 1, 2

[34] XU, H.; DOMÍNGUEZ-GARCÍA, A. D. ; SAUER, P. W.. **Optimal tap setting of voltage regulation transformers using batch reinforcement learning**. 35(3):1990–2001. 3, 5.2

[35] YANG, Q.; WANG, G.; SADEGHI, A.; GIANNAKIS, G. B. ; SUN, J.. **Two-Timescale Voltage Control in Distribution Grids Using Deep Reinforcement Learning**. arXiv:1904.09374 [cs], Dec. 2019. arXiv: 1904.09374. 3, 5.2

[36] ZHANG, W.; LIU, Y.. **Multi-objective reactive power and voltage control based on fuzzy optimization strategy and fuzzy adaptive particle swarm**. 30(9):525–532. 3