City University London

MA/MSc in Data Science

Project Report

2018/2019

Cost Optimization In Frequency Control Of Unbalanced Distribution System

Flavio R. de A. F. Mello

Student Id: 180037799

Supervised by: **Dr. Eduardo Alonso & Dr. Dimitra Apostolopoulou**

Date of Submission:

26 September 2019

By submitting this work, I declare that this work is entirely my own except those parts duly identified and referenced in my submission. It complies with any specified word limits and the requirements and regulations detailed in the assessment instructions and any other relevant programme and module documentation. In submitting this work I acknowledge that I have read and understood the regulations and code regarding academic misconduct, including that relating to plagiarism, as specified in the Programme Handbook. I also acknowledge that this work will be subject to a variety of checks for academic misconduct.

Flavio R C F Mello

Signed: Flavio R. de A. F. Mello

Abstract: Introduction of new technologies and increased penetration of renewable sources is altering the power distribution landscape. Gradually, this ecosystem will move away from few, large scale generators and start including increasingly larger numbers of micro generators. The centralized strategies currently employed for performing tertiary control will need to be revisited and decentralized to conform to the increase in generating nodes in the grid. This project demonstrates the use of Multi-Agent and Multi-Objective Reinforcement Learning techniques to train models to perform frequency control through decentralized decision making. In the Multi-Objective realm, this proposes a comparison between reward composition and action composition techniques. Both techniques for multi-objective proposed in this study are able to perform the task in general terms, but not to the extent to conform to industry standards. Parallelly, the software developed to perform the study was done to allow for reuse in future studies.

Keywords: *Multi-Agent Reinforcement Learning — Multi-Objective Reinforcement Learning — Frequency Control — Economic Dispatch — Deep Deterministic Policy Gradient*

Table of Contents

1. Introduction and Objectives

1.1 Background and Motivation

Over the recent years, the field of electrical power systems has been experiencing the beginnings of what may prove to be a structural transformation. Renewable sources of energy have been increasing their penetration in the marketplace, which may displace traditional sources such as fossil fuels based matrixes. Diminishing costs of solar panels lead to increased adoption in households, to the extent that there are already legal provisions for household customers to sell stored energy back into the electrical grid (Ambrose, 2019b). Vehicle to grid and smart charging technologies are posed to enable electric cars to contribute to balancing the power grid they are connected to, either by providing energy stored in their batteries or by throttling their power consumption(Ambrose, 2019a)(Steitz, 2019). UK's National Grid projects that “"By 2030 we could see as many as nine million electric vehicles on the road”(Leggett, 2017). Taken together, these phenomena will mean an increased capillarity in the sources which are able to inject power into the electrical grid. This represents a significant increase on diversification & complexity of the grid, shifting away from a small number of large scale producers to include an ever increasing number of micro-sized sources in the form of individual households, electric cars, etc. This increase in complexity, in turn, will generate increasing demand for intelligent, automated and decentralized control solutions.

In light of these projections, this study proposes the use of reinforcement learning techniques for tackling the increased complexity and required decentralization of control. In this context, electrical system control signifies identifying and issuing the necessary commands to electrically balance a given power system.

1.2 Document Structure

This document is arranged into five major sections of content, one section for the bibliography and additional sections for the appendices. Section 1 describes the broad context of the study, the motivations behind it as well as the objectives it ultimately aims to achieve. Section 2 provides in-depth technical context for both electrical and reinforcement learning concepts dealt with in the analysis. Section 3 is reserved for the methodology used in this work, it includes descriptions of the exact settings used to perform the experiments as well as any relevant design choices made over the course of the study. Section 4 presents the results of the experiments ran as well as interprets their individual and joint consequences. Section 5 covers a broader discussion of the findings presented in section 4. Finally, section 6 provides a meta analysis of this work, critically evaluating some of the choices taken over the course of the project and how these impacted the final results. The final section also proposes future directions of work and assesses the results of this work with regard to the initial objectives.

With respect to the appendices, appendix A contains the original project proposal as submitted for evaluation. Appendix B covers the codebase provided, including the folder structure, libraries used and instructions for running the project.

1.3 Objectives

The primary objective of this study is to investigate the feasibility of leveraging Reinforcement Learning (RL) techniques for training autonomous agents able to perform frequency control in an electric power system according to two distinct hierarchical objectives. The primary objective would be to have the system's operating frequency be within the tolerated deviation range of the system's nominal frequency. The secondary objective would be to minimize the joint cost of production. As a means of performing the investigation, this study aims to incorporate Multi-Objective RL on top of a previous work which leveraged a Multi-Agent RL algorithm to produce satisfactory results in single objective frequency control (Rozada, 2018).

Fundamentally, the primary directive of this study is to answer the following question:

"Can we leverage Multi Objective, Multi-Agent Reinforcement (MOMARL) techniques to control the power output of multiple power generators to efficiently balance a dynamic load while accounting for the secondary objective of minimizing the cost of energy production?"

A secondary goal of this study is to structure the developed software in such a way as to ease further experimentation and learning both in terms of the particular problem studied (frequency control of electric power systems) as well as multi-agent and multi-objective reinforcement learning scenarios in general. As a consequence, the codebase resulting from this effort should be didactically readable, robust enough to enable further experimentation with different setups, and flexible enough to enable future maintenance and development.

1.4 Methods

The immutable deadline for this project, and the uncertainty associated with dealing with fields and algorithms not fully mastered at the start of the project, lead to a seemingly conflicting set of strategies. While waterfall methods usually yield better results in dealing with set deadlines, agile methodologies emerged in the software engineering industry as a means of tackling uncertainty. While these techniques seem somewhat contradictory, the author's prior experience in dealing with similar circumstances in the software engineering sector indicate that it is possible, to some extent, to mix components of both strategies. Waterfall-like planning is used to break down the task into broad tasks and establishing a coarse timeline which can be used to track the project progress and raise alerts if the observed progress deviates from the planning. Complementally, agile-like techniques are used to gradually decrease uncertainty and break down the broad timeline into actionable tasks as the project progresses. Figure 1 shows the planned timeline for the total execution of this study.

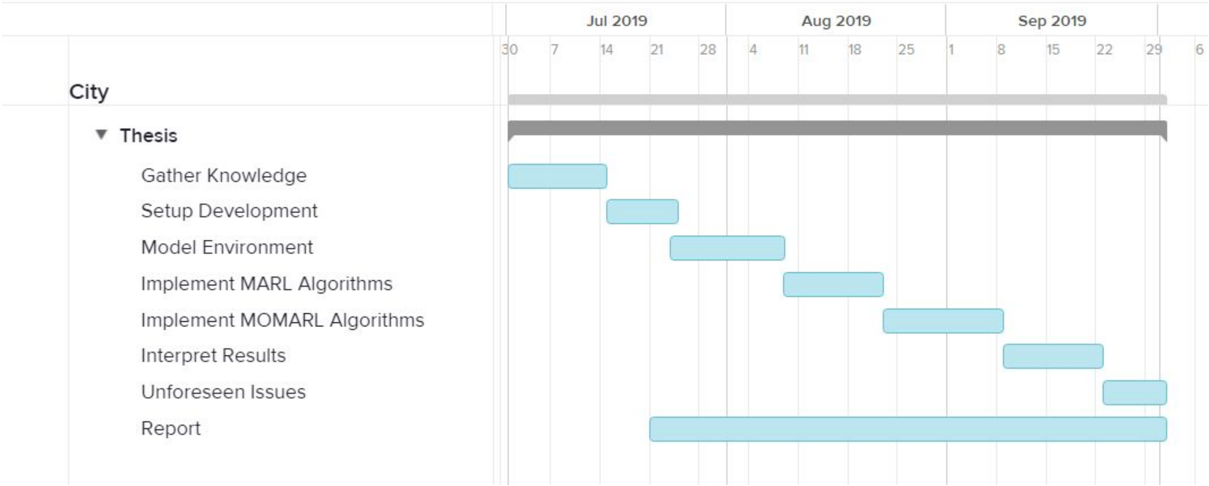


Figure 1: Scheduled work plan for completing the project within the provided time frame.

1.5 Beneficiaries

Being a multidisciplinary effort, this analysis is posed to be of utility to a wide range of groups. Scholars in electrical engineering field may find the results directly useful from the more evident angle of decentralized control through Reinforcement Learning. Scholars in reinforcement learning and artificial intelligence fields may find usefulness in the proposed strategies for performing multi-objective learning and combining multi-objective techniques with the multi-agent algorithm employed. Academics from the aforementioned fields may find further use in the codebase which resulted from this study to perform additional analyses. Individuals associated with electrical network operators may rely on these findings to guide future strategies.

1.6 Unplanned Changes

The most significant change that occurred throughout the project was in the action composition model (Experiment III). Initially, it was envisioned as being composed of two submodels: one for balancing frequency and one for optimizing cost while keeping the power constant. The latter submodel was originally designed as its own multi-objective model, it was to minimize a compound reward function whose components were designed to a) reward the total output being kept constant and b) reward the cost being minimized. However, during the course of the project, there were some issues with finding a combination of constants in the compound reward function that properly trained a model able to achieve both objectives. In view of that, it was decided to change the cost submodel's reward function to a single objective one. The objective chosen, as described in section 3.7.3, encapsulates the same goals (minimize cost while keeping output constant), in light of that, the ultimate impact in the study essence was minimal. The biggest impact, in fact, was the time spent tuning the initial design before moving on to the one currently employed. It is important to note that, even this original component was removed, this study still performs reward composition in experiment II the results of which are ultimately compared against the action composition strategy employed in experiment III.

2. Context

2.1 Electrical Energy Production

Modern electrical energy distribution is largely done by means of wide ranging synchronous grids, which distribute electricity in the form of three phase alternating current. Being synchronous means the entirety of the grid is electrically connected and thus every element attached to the grid share the same observed operating frequency. This is true for both the electrical consumers (households, offices, industry, public infrastructure, etc) - deemed as loads in the electrical systems - as well as the electrical producers, traditionally in the form of large-scale power plants relying on a multitude of energy sources (oil, coal, nuclear, hydroelectric, wind, solar, etc), these are deemed as generators in the shared electrical systems. In these wide ranging systems, there are two main nominal frequency standards around the world, 50Hz - which is the standard in the U.K., continental Europe, India and Russia, among others - and 60Hz which is the standard adopted in North America, Brazil and others. It is important to note that these frequencies are the nominal operating frequencies in such grids. The actual operating frequency in fact changes over time according to i) the total power being injected into the system by all the generators; and ii) the total power being consumed by all loads. If the total power injected into the system is greater than the power being consumed, the operating frequency rises. Conversely, if the total power injected is lower than the consumed one, the system's operating frequency lowers. With the exception of a few select cases[1], it behoves the generators to adjust their output to match the current load, thus electrically balancing the complete system. For the purpose of simplification, this study disconsiders the effects of reactive power and the ensuing phase shift. Consequently, the term power refers uniquely to the apparent power.

[1] While some extremely large-scale consumers such as energy intensive industries are required to ramp-down their power usage in specific circumstances, in general the power generators are the ones which in fact continuously change their output to match the system load.

This process of relying on the observed system frequency to adjust the power generation to match the total system load is named frequency control, and can be divided into three hierarchical layers: Primary, Secondary and Tertiary control.

Primary Control: Primary control acts to counterbalance changes in the total system load by adjusting the output levels of all generators attached to the grid by an amount proportional to the difference between the observed and nominal frequencies. Droop Control(Miller and Malinowski, 1994) would be the most commonly applied form of primary control. Primary control has the benefit of being completely decentralized as each generator is able to observe individually the current frequency in the system. However, this technique has its limitations as it results in steady state errors and disconsiders any economical implications.

Secondary Control: Secondary control aims to further balance the power grid by acting upon the steady state error resulted by the limitations in primary control. To this end, Automatic Generation Control (AGC) algorithms(Miller and Malinowski, 1994) are often employed. However, these algorithms are often centralized to some extent, with an individual entity overseeing the entire grid and issuing commands for the individual generators.

Tertiary Control: Differently from primary and secondary, the objective of tertiary control is not to minimize frequency deviation from nominal, but to minimize the total production costs of the grid. As such, it is also referred to as Economic Dispatch. Doing so also requires a centralized entity with knowledge of each generating node's power output and cost of production curve, as well as theirs and relevant transmission facilities' physical limits with regards to minimum and maximum output levels(Congress, U. S., 2005).

According to their cost structure and ability to ramp up/down the production electrical power plants can be divided into three categories:

Base Load: These power plants usually operate at constant production levels. Collectively, they provide energy to the system enough energy to match it's base load - the load that is constantly demanded in the system, regardless of mid-high frequency fluctuations that happen over the course of a day. Typically these plants have either limited ramp ramp up-down capabilities, or few incentives to do so, as it's cost structure is such that the power output has a limited role in the overall cost of construction and operation. Examples of base load power plants are nuclear, hydroelectric, geothermal, solar and wind.

Peak Load: The common characteristic among this group is the ability to startup and ramp up/down the power production significantly quicker than base load plants. Peak load plants are not necessarily active throughout the whole day, and are activated to match large increases in total load during peak demand.

Load Following: Load following plants are in-between base and peak load plants. They do not have the capability to startup and ramp their outputs as fast as peak load ones, but can - and have cost structure incentives - to do so, differently from base load plants. Load following plants, as the name implies, do continuously adjust their output to match (or follow) the observed system load.

The results of this study would therefore apply to both Load Following and Peak Load plants, the types of plant that in fact change their output according to the variations in the system load. The form with which the system reacts to these changes in output by the generation plants is described with the following set of equations:

$$P_{t+1} = P_t + (Z_t^{total} - \Delta\omega_t / R_D - P_t) / T_G \qquad (1)$$

$$\omega_{t+1} = \omega_t + (P_{t+1} - L_{total} - d * \Delta\omega_t) / m \qquad (2)$$

$$Z_t^{total} = \sum_i^I Z_t^i \qquad (3)$$

$$L_{total} = \sum_j^J L_j \qquad (4)$$

$$\Delta\omega_t = \omega_t - \omega_{nominal} \qquad (5)$$

Each term is further explained in the following table:

Term	Name	Constant Value
P	Power generated	-
Z	Secondary control action	-
ω	Frequency	-
R_D	Droop coefficient	0.1
T_G	Time constant	30
L	Power consumed (Load)	-
d	Damping coefficient	0.0160

m	Electrical inertia	0.1
pu	Per Unit	100 MVA

Table 1: Definition of the terms used in the electrical system equations used

2.2 Production Cost and Economic Dispatch

There are multiple cost factors associated with the production of electricity, they can be broadly divided into three larger segments: capital, fuel and additional costs.

Capital costs are typically one-off costs associated with the electrical production enterprise. Land purchase, site construction, and equipment purchase are classified as capital costs. Fuel costs encompass the obtention and transportation of the fuel used for running the power plant. Finally, additional costs are recurring costs other than fuel - insurance is a good example of an additional cost.

Different means of energy production may present vastly different characteristics with respect to their cost profiles. Nuclear plants capital costs include expenses related to nuclear waste management and plant decommissioning, as such their capital costs are extremely high. Solar and Wind plants also present high capital costs, but their fuel costs are, as expected, null. On the other side of the spectrum, fossil fuel based generators capital costs are comparably low, but their fuel costs are considerably higher. Power plants are long-lived enterprises, often operating for decades, additionally the amount of variable factors related with operations ultimately render any precise calculation of costs nearly impossible. As a means of comparing the approximate costs of different types of power plants, one can make use of the Levelized Cost of Energy (LCOE)(U.S. Energy Information Administration, 2019). The LCOE is based on projections of capital, fuel and additional costs, along with the expected operational load over the entire operational life of a plant, then averaged to produce an estimated cost or production per unit of energy (e.g. $/MWh). For the purposes of this study, relying on LCOE would not suffice as it provides no insight with respect to the differences in cost associated with operating the plants at different output levels, rendering it impossible to pinpoint the per plant output combination which is cost-optimal for any given total power production.

For estimating the cost associated with reaching a given power output, a typically quadratic or piecewise linear function is used to express the operating cost as a function of power output. The process of optimizing the system for cost is, as mentioned above, named economic dispatch or tertiary control. Traditionally, a multitude of iterative techniques may be employed for dealing with economic dispatch, such techniques include Newton's approximation method, Lagrange Multipliers, and Dynamic Programming, among others (Wood and Wollenberg, 1996). Furthermore, leveraging these techniques requires having a centralized controller calculating the most cost effective production level for each generator in the network and issuing commands with the necessary output adjustments to every one of them. In a scenario with increasing numbers of generators attached to the grid, as well as increasing diversity in terms of production characteristics and capacity, the centralized approach to economic dispatch becomes ever more complex. The complexity of the calculations required to find the optimal arrangement increases dramatically with the number of generators in the system. Issuing individual timely commands for each generator also may become an issue.

In mathematical terms, the following set of equations apply to the economic dispatch problem:

$$C_{total} = \sum_{i} C_i \tag{6}$$

$$C_i = \alpha_i + \beta_i \cdot P_i + \gamma_i \cdot (P_i)^2 \tag{7}$$

$$P_{total} = \sum_{i}^{I} P_i \tag{8}$$

$$L_{total} = \sum_{j}^{J} L_j \tag{9}$$

$$P_{total} = L_{total} \tag{10}$$

$$P_{i\,min} \leq P_i \leq P_{i\,max} \tag{11}$$

In other words, solving tertiary control entails finding the power output combination set $\{P_1, P_2, \ldots P_n\}$ that minimizes the global cost C_{total} while respecting the constraints that the total power produced by the generators must be equal to the total power consumed by the loads (10) (i.e. the system must remain balanced), while keeping every generator output within its operational limits (11). The global cost is the sum of all generator's individual cost of production which is calculated in terms of its current output (P_i) and coefficients (α_i, β_i, γ_i) which vary according to the category of the power plant in question (coal, gas, oil, etc).

2.3 Reinforcement Learning

As a field of study, Reinforcement Learning (RL) emerged from research done on how the learning process takes place in animals and humans. The first instances of RL techniques accomodate for one agent pursuing a single objective while interacting with the environment. Commonly taught examples of such algorithms include Q-Learning and SARSA. From a software agents standpoint, reinforcement learning can be defined as a family of techniques used to train software agents based on their interactions with the environment and the ensuing rewards/punishments observed by the agents. Given enough observations (learning episodes) the goal is to have the then trained agent able to issue commands so as to reach a state which is desired. This approach differs from the procedural, imperative one often associated with computers and software, although the underlying mechanism are still algorithmic and procedural - software agents are still software by nature. Whereas procedurally declared agents are brittle and rely on the programmers to account for every possible scenario, successfully trained agents can be able to generalise and issue proper commands when encountering previously unseen situations. Since learning is reached through experimentation rather than express declaration, there is diminished cognitive load on the part of the programmers to account for each and every scenario imaginable.

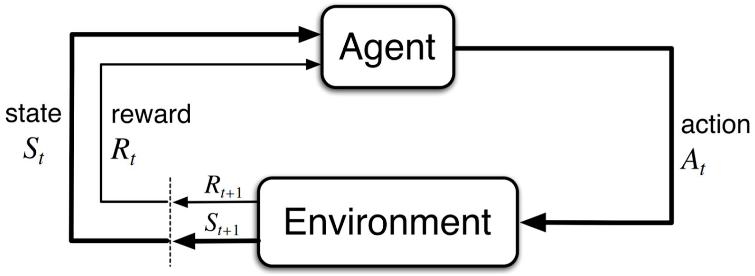


Figure 2: Reinforcement Learning framework of agent/environment interaction (Sutton and Barto, 2018).

Current reinforcement learning techniques can be loosely divided into three major categories: Value Based methods — which rely on approximating the value associated with the possible state-action pairs; Policy Based methods — which rely on approximating the best policy for the possible states; or a combination of both(Kapoor, 2018).

2.3.1 Multi-Agent Reinforcement Learning (MARL)

Multi-Agent Reinforcement Learning is, simply put, the collection of reinforcement learning used for solving problems associated with Multi-Agent Systems (MAS) which, in turn, can be defined as *"an area of distributed artificial intelligence that emphasizes the joint behaviors of agents with some degree of autonomy and the complexities arising from their interactions"* ('T Hoen, P. J. et al. 2006). In simplistic terms, these problems are framed as either competitive (agents seek different objectives and success is measured by each of their individual gains) or cooperative (agents share a common goal and success is ultimately measured by their ability to reach said goal globally). In reality, some problems may not be fully classified using this simplistic dichotomy, and usually fall somewhere within the fully-cooperative to fully-competitive spectrum. The problems approached in this study, however, can be classified as fully cooperative as the agents work together to reach both objectives.

2.3.2 Multi-Objective Reinforcement Learning (MORL)

Multi-Objective Reinforcement Learning (MORL) relates to RL problems which multiple, sometimes conflicting, objectives. Successfully trained MORL agents should be able to perform tradeoffs, sometimes intentionally sacrificing adherence to one objective while advancing towards a more desired global state. This can be, for example, in the form of higher adherence of another higher order objective (if the objectives are hierarchically sorted); or, if there is no ordinality relationship, seeking a middle ground - minimizing the total deviation from all objectives altogether. To this end, there are a number of different techniques that can be employed, ranging from simplistic weighted-sum or winner takes all approaches (Liu, C., Xu, X. and Hu, D., 2015), to more intricate ones such as Pareto dominating policies (Moffaert, K. Van and Nowé, A., 2014). The choice of which approach to take should be based on the particular task at hand and, therefore, becomes an integral part of the design process of the solution.

It is important to observe, here, how MORL and MARL are for solving independent facets of a given problem. Whereas MARL is associated with the multiplicity of agents coexisting and interacting in the same problem space, MORL is tied to the multiplicity of objectives any given agent strives to achieve. Therefore any given system may contain either MA or MO elements, neither, or both.

2.4 Learning Algorithm

The technique used in this study is named multi-agent deep deterministic policy gradient (MADDPG) and is considered an extension of deep deterministic policy gradient (DDPG), combined with some elements of actor-critic RL techniques(Lowe, et al., 2018)(Lowe, et al. 2017). By relying on actor-critic methodologies, this algorithm would be categorized as a combination of both value and policy based methods. As the name implies, actor-critic models make use of two distinct entities: the actor and the critic. The actor plays the role of a policy based entity, while the critic relies on value-based methodologies to evaluate the policy learned by the actor and provide feedback on such in the form of calculated policy gradients.

The MADDPG algorithm applies this concept to multi-agent scenarios by centralizing learning whilst decentralizing execution. Each agent is constituted of both an actor and a critic. Actors remain decentralized in nature - each actor has access only to the same information said agent would have in execution time. The critics, however, are centralized as they have additional information (in the form of the actions taken by all the other actors in the system) which is unavailable to their respective actors in execution time. This additional information is used to shape the policies learned by the actors. Once trained, the agents rely solely on their actors to take actions in the execution environment. Therefore even though this technique relies on information centralization during training, it is able to operate in decentralized manner as the critics are not used post-training.

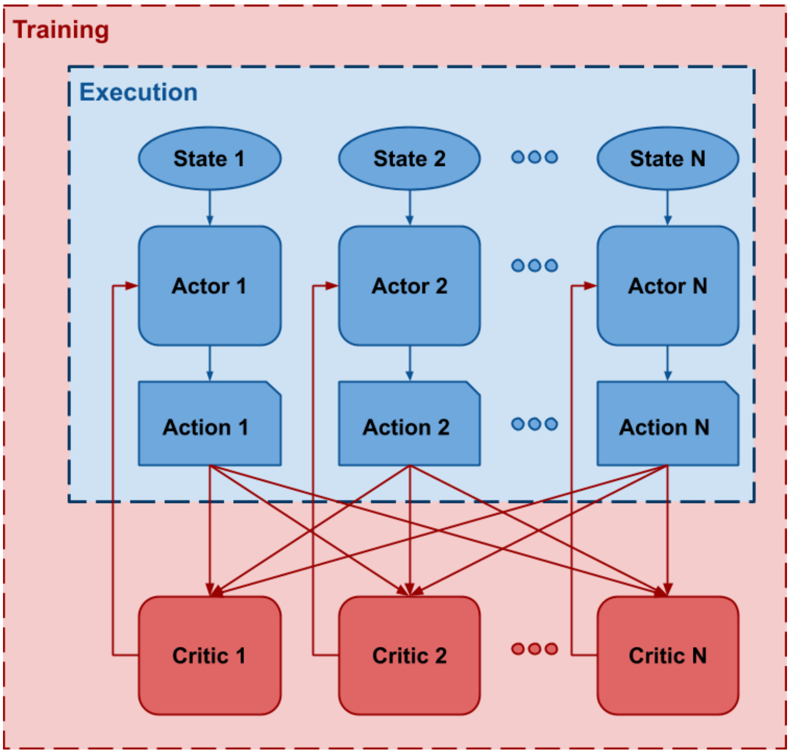


Figure 3: Actor/Critic relationship in MADDPG. Centralized critic entities are used only during training

In order to provide further stability to the learning process, the MADDPG algorithm makes use of the so-called “target” networks, as is the case in the single agent counterpart (DDPG). These target networks are *“time-delayed copies of their original networks that slowly track the learned networks”*(Yoon, 2019). If the algorithms were to forego the use of these target networks, the updates of the actor network would depend on the values produced by the critic network and vice versa. Thus, for both actor and critic networks, their update values would indirectly depend on the values produced by the networks themselves. Effectively this makes the learning process less stable and more prone to divergence. The addition of these target network counterparts breaks this interdependence cycle, as the update value of the actor and critic networks now depend on the target critic and target actor networks, respectively. At the end of the update cycle, after updating the actor and critic networks, their target counterparts are finally updated by adding the respective original network parameters into the target parameters weighted by τ , as shown in equation (12) below. Figure 4, below describes the network update cycle.

$$\Theta^{i}_{target} = \Theta^{i}_{original} \cdot \tau + \Theta^{i}_{target} \cdot (1 - \tau) \quad (12)$$

```
# Sample the experience batch (mini batch)
experienceBatch = EpisodeBuffer.sampleExperienceBatch()

# Get Target Actors' Actions
allTargetActions =
TargetActors.calculateActionsForBatch(experienceBatch)

# Get Target Critics' Q Estimations
allCriticTargetQs =
TargetCritics.calculateQvalsForBatch(experienceBatch, allTargetActions)

# Update the critic networks with the new Q's
Critics.updateModelsForBatch(experienceBatch, allCriticTargetQs)

# Calculate actions for all actors
allActions = Actors.calculateActionsForBatch(experienceBatch)

# Calculate the critic's gradients from the estimated actions
allGradients = Critics.calculateGradientsForBatch(experienceBatch,
allActions)

# Update the actor models with the gradients calculated by the critics
Actors.updateModelsForBatch(experienceBatch, allGradients)

# Update target actor and critic models for all agents
TargetActors.updateModels(Actors, tau)
TargetCritics.updateModels(Critics, tau)
```

Code Block 1: Pseudocode for the MADDPG update cycle.

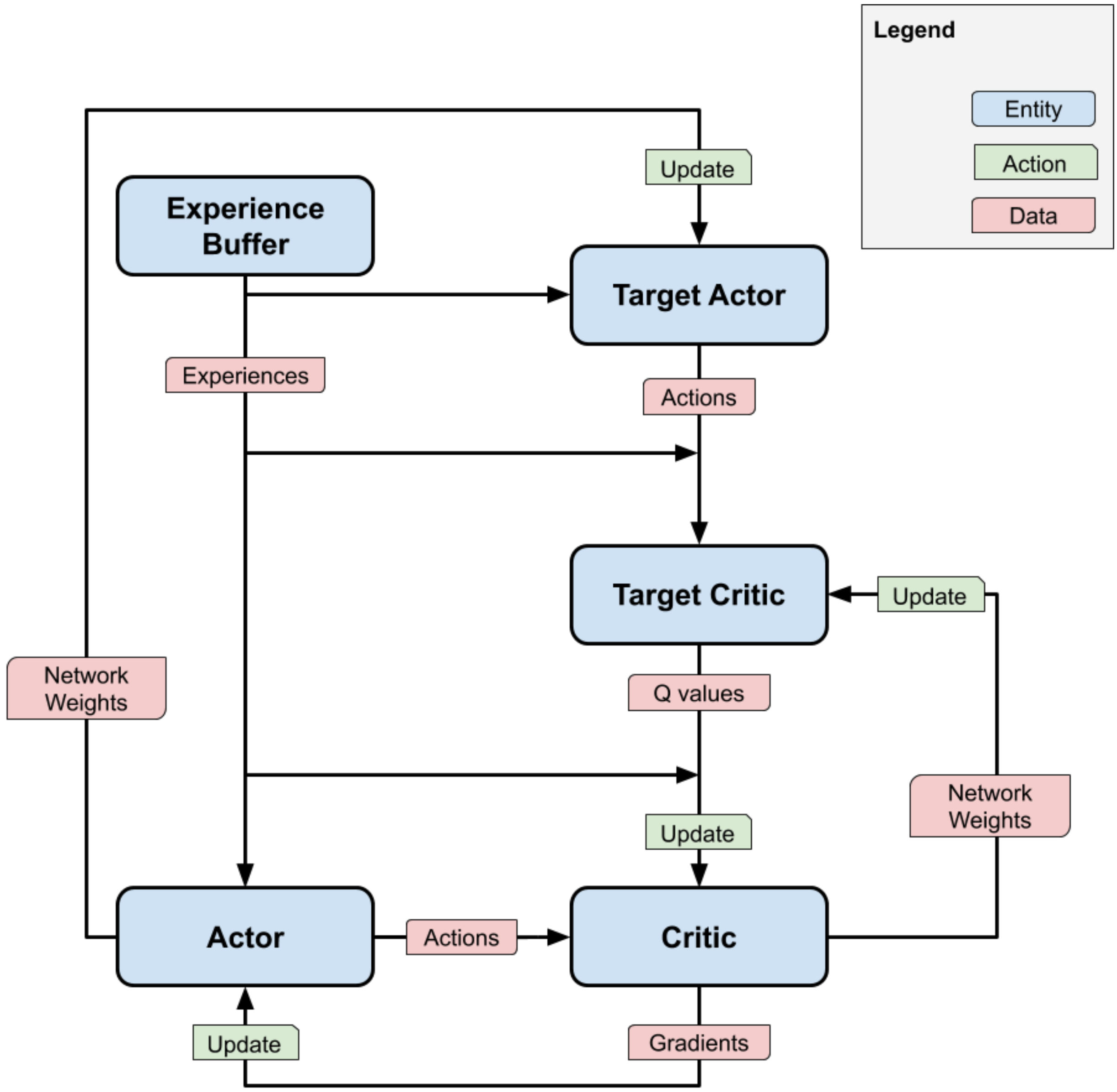


Figure 4: MADDGP update cycle dataflow

Finally, as is the case with contemporary reinforcement learning techniques, MADDPG makes use of the experience replay buffer. Essentially, this buffer stores previous experiences in the form of the state-action-reward triplet. During the network update cycle, which occurs every n steps, these experiences are then randomly sampled and used to perform the update process. Given that the environment studied have an explicit time-dependence relationship between states, the sampled experiences take the form of traces of consecutive states.

2.5 Exploration/Exploitation Tradeoff

A common aspect of reinforcement learning training is the exploration/exploitation tradeoff: how to balance, during the training of the model, the conflicting needs of fully exploring the state-action space in search of global optima rather than local ones (explore); and maximizing the earned reward by relying on the currently best performing policy (exploit). This tradeoff is controlled using a single parameter ε, usually constrained to the interval [0, 1].

In the early stages of training, it is of interest to gather shallow information covering an ample subset of the state-action space, thus the agent is encouraged to explore the state-action space. As training progresses, it becomes a better use of the resources to fine tune the solution by gathering deeper information regarding a smaller subset of the state-action space that is more of interest (i.e. in the vicinity of the best performing policies) thus the agent is encouraged to gradually exploit it's currently assimilated policy to take the best actions. To this end, the learning process begins with a high ε value (e.g. 0.999), indicating higher propensity for exploration, with ε smoothly declining towards 0 over the course of learning.

In the problem encountered in this analysis, the action is a unidimensional, continuous value representing the secondary response of the power generator controlled by the agent. Therefore, the exploratory noise is in the form of a sample from a Gaussian distribution which is then scaled by epsilon and added to each actor's actions over the course of the learning process. The smooth decay of epsilon over time can be intuitively interpreted as gradually focusing the exploration in the ranges of interest of the state-action space. In this study, the epsilon decay function was implemented with a two stage approach: a smaller decay is used until a specified threshold value of epsilon is reached, at which point a larger decay starts being used.

2.6 Reward Function

The reward function plays a crucial role in Reinforcement Learning applications. A well crafted reward functions should be able to distill the degree of which the objective is reached into a single numerical value. Reward functions are expressed as a function of one or more elements of the reached state. In discrete state-spaces, where the objective is associated with reaching one or more terminal state(s), the reward function is often sparse, granting a fixed reward if said state is reached. In continuous state-action spaces, the reward function is often continuous - increasing the reward granted per step to the extent the reached state is more desired.

The number of objectives a given system strives to accomplish is equal to the dimensionality of the reward function. For single objective systems, the reward function depends on a single variable. In such cases, the defining factor of the function would be its general shape. Provided there is a monotonic relationship between increasing adherence to the objective and the earned reward, the scale of the function is not as important, provided there is sufficient variation in the granted reward between different encountered inputs.

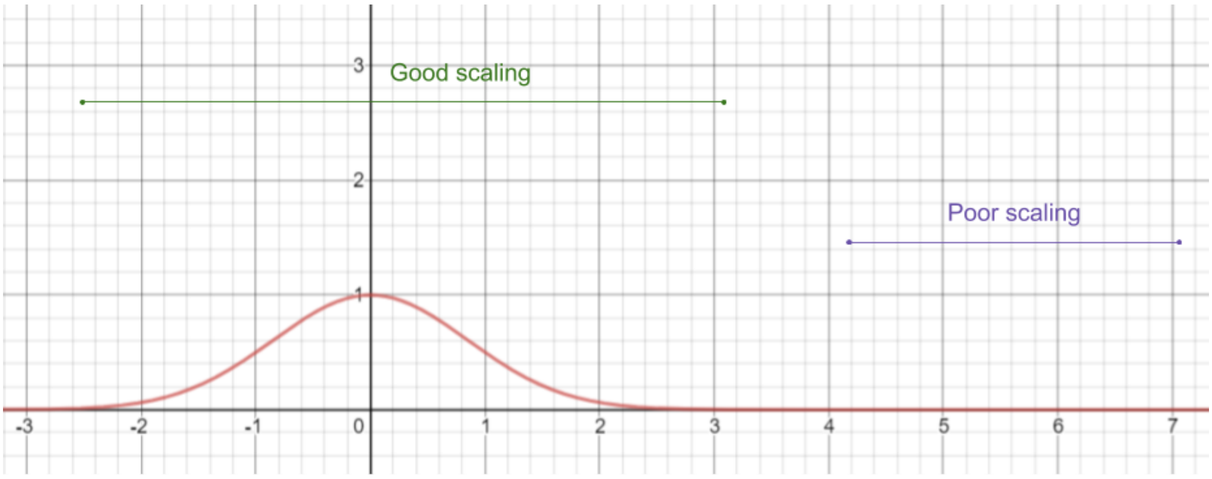


Figure x: Comparison of proper vs poor reward/expected domain values scaling

In multi-objective systems, the scaling of the reward function becomes significantly more important, as in this case, the relationship between the variables(objectives) directly impact the trained agent's resulting policy. One simple example would be if there is a significant difference in magnitude between the objectives' contribution to the final reward (e.g. $r(x, y) = 1000 \cdot x + y$). In such case, any reward obtained by increasing adherence to objective y would be negligible compared to any gain towards x. The scaling between different reward components is then directly tied to the priority given to each objective. Therefore, in multi-objective scenarios, it is not only important to ensure that the reward function and objectives inputs operate on the same scale, the proportion between the objective components also plays an integral role in the design of a successful reward function. When crafting a reward function, it is valuable to have access to a tool able to plot the graphs, especially in the case of multi-objective functions, as the relationship between the variables play an even greater role in the success of the model.

3. Methods

This study takes as starting point and draws inspiration from the results obtained by Sergio Rozada, a former student of City, University of London MSc Data Science programme. In terms of software, the base implementation of the MADDPG algorithm using Tensorflow and electric system simulation was used as a starting point upon which a series of modifications and improvements were made. The reasons, gains and nature of these alterations are further described in section 3.6. But in terms of proportion, the code excerpts which comprise the starting point of this study are the following classes: *actor_maddpg, critic_maddpg, maddpg_secondary, recurrent_experience_buffer, Node_Secondary,* and *Area_Secondary*. Taken together, they amount to approximately 450 lines of code. The codebase produced in this study totals over 2500 lines of code, not including the individual jupyter notebooks used to perform and produce the results of each experiment. Additional libraries and supporting software used in this study are referenced in section B.2 of Appendix B.

This study sets out to investigate the possibility of leveraging Reinforcement Learning techniques for enacting primary, secondary and tertiary control. Rozada's work indicates how Multi-Agent Reinforcement Learning (MARL) can be used to perform primary and secondary control. Specifically, the Multi-Agent Deep Deterministic Policy Gradient Actor Critic algorithm proved able to successfully perform primary and secondary control, but failed to perform tertiary control alongside. In spite of being separate layers of control, both primary and secondary control share a single common objective: to minimize frequency deviation from a nominal setpoint. Tertiary control, however, is associated with a completely unrelated objective: to minimize the total cost of electricity production. These differences in objective alignment could explain why the MAADPG algorithm, as implemented in that study, successfully performed primary and secondary controls but failed with tertiary control. For this end, it is necessary then to incorporate Multi-Objective Reinforcement Learning techniques into the system devised. It is precisely this problem that this study sets out to investigate: to gauge the possibility of leveraging Multi-Objective Multi-Agent Reinforcement Learning (MOMARL) techniques to train agents that are capable of issuing commands that i) adjust the generators' total electrical output to match the system load; and ii) do so in a way that is cost optimal for the system as a whole.

3.1 Electrical System

In order to perform the control experimentation, an electrical system simulator was implemented according to the equations described in section 2.1. The simulated environment consists of a user-defined number of nodes (loads and generators) all attached to the grid in the form of a single bus. For the purpose of this study, a consistent system topology was used across all experiments: three generators and one single load, as shown in figure 5.

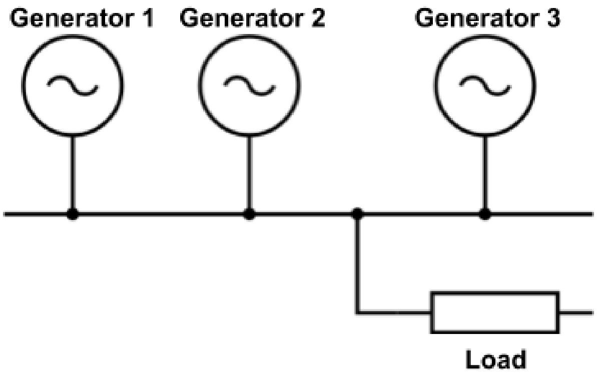


Figure 5: Schematics of the simulated electrical system used in all learning experiments

Being frequency control a continuous matter, each simulation begins considering that the system is fully balanced ($P = L$, $\Delta\omega = 0$) and a perturbation in the form of a change in the total load occurs at t_0. The task being performed then is to balance the system after this initial perturbation.

Considering that the training of RL models rely on multiple learning episodes being run and, in the interest of increasing the robustness of the models trained, the application developed is able to introduce noise in the simulated environment in the form of changing the initial values for the loads and generators power levels. The noise takes the form of a uniform distribution with magnitude defined by the user and can differ from node to node. In practical terms this means that each learning episode is run under slightly different simulation terms, the degree of such difference being defined by the user when specifying the electrical system to be simulated. Furthermore, besides the initial output and its noise, generators are also specified in terms of their minimum and maximum outputs as well as their cost profile. As is the case with the added noise, these specifications are done on a per generator basis.

Despite basing the study in this somewhat simple electrical system, the application developed is able to produce more complex systems with varying numbers of loads and generators. This characteristic of developing a system which is capable of dealing with more complex instances than the ones studied was a recurring theme throughout this study as one of the objectives was to build a software application flexible enough for further experimentation. This is further discussed in section 3.6.

3.2 Economic Dispatch

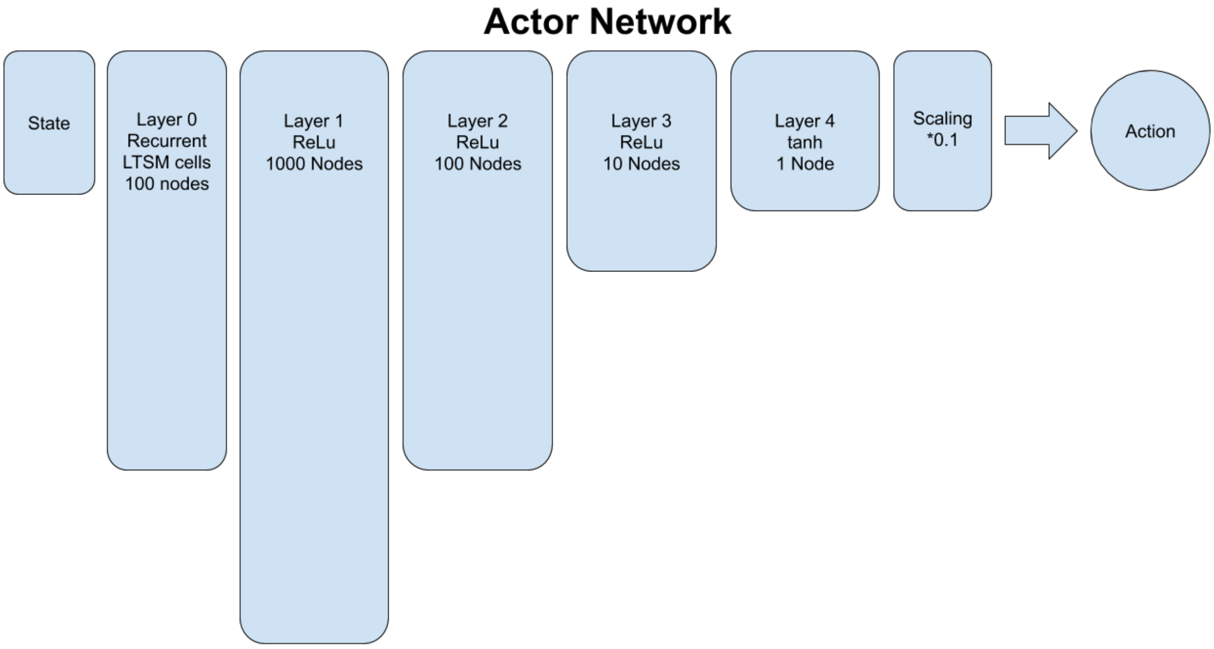
As previously stated, this study proposes the use of reinforcement learning techniques for solving the tertiary control along with the primary and secondary control problems. In order to provide benchmarks for the achieved performance, it was decided to solve the economic dispatch problem using traditional methods. The solutions devised will then be compared against what is the optimal output setup. This was done using SciPy, a well regarded open-source Python library for widely used in science, mathematics, and engineering fields. More specifically, the *minimize* function of the *scipy.optimize* package. This function uses dynamic programming techniques to find the input values which minimize a given scalar function while respecting specified set of constraints. In the context of this study, this function is called every single time step, providing a benchmark of what is the optimal output setup at all points during the experiment. Furthermore, the input cost function used was the total cost of production as the sum of all generators' individual cost of production in terms of their respective output, as specified in equations (6) and (7). The constraints provided were the ones in equation (11), that is all generators' outputs must be within their corresponding minimum and maximum operational outputs. With respect to the constraint in equation (10), it is critical to observe that, in the circumstances encountered in this analysis, solving economic dispatch happens concomitantly and independently to balancing the electrical system. With this in mind, rather than providing a constraint indicating that the total power in the optimized solution must match the total power consumed by the loads, expressed in equation (10), it was decided to specify a similar constraint indicating that the total power output in the optimal solution must match the total output being provided at that current point in time. Therefore, with the provided set of inputs and constraints, the power output setup returned by the *scipy.optimize.minimize* function indicates the optimal setup, with regards to cost, for providing the same total output as is being currently provided at each step evaluated. This data is used to plot a cost-optimal curve indicating, for every step in the experiment, what is the optimal cost and how much the observed power setup deviates from the optimal. Since *scipy.optimize.minimize* provides both the minimum scalar value (cost) and the input set that yields such value, the deviation from optimal cost can be assessed both with respect to the global cost as well as with respect to each individual generator's cost or output.

3.3 Neural Networks

The MADDPG algorithm leverages fully connected deep neural networks to model both the actor and the critic. In the form implemented in this study, both networks follow the same schema, with slight changes in the input/output layers. This section describes both the commonalities and differences in the actor and critic networks. Finally, this study employs the same algorithm to learn different policies to achieve different objectives. Often this requires changes in both the reward function and the set of variables that compose the “state” input. These changes are further described in sections 3.3.1, 3.4 and 3.7 on a case by case basis. The base neural networks used in this study contain 6 layers, described below.

Layer index	Layer Type	Layer Size	Activation Function	Observation
0	Input layer	Variable	-	Number of inputs changes between actor and critic as well as for each objective
1	Recurrent	100	Tanh	LTSM Cell
2	Feed Forward	1000	ReLU	Fully connected
3	Feed Forward	100	ReLU	Fully connected
4	Feed Forward	50	ReLU	Fully connected
5	Output	1	Tanh (Actor) or Linear (Critic)	Output of the actor network is scaled down by a factor of 0.1

Table x: Description of base neural network used in actors and critics in the implemented MADDPG



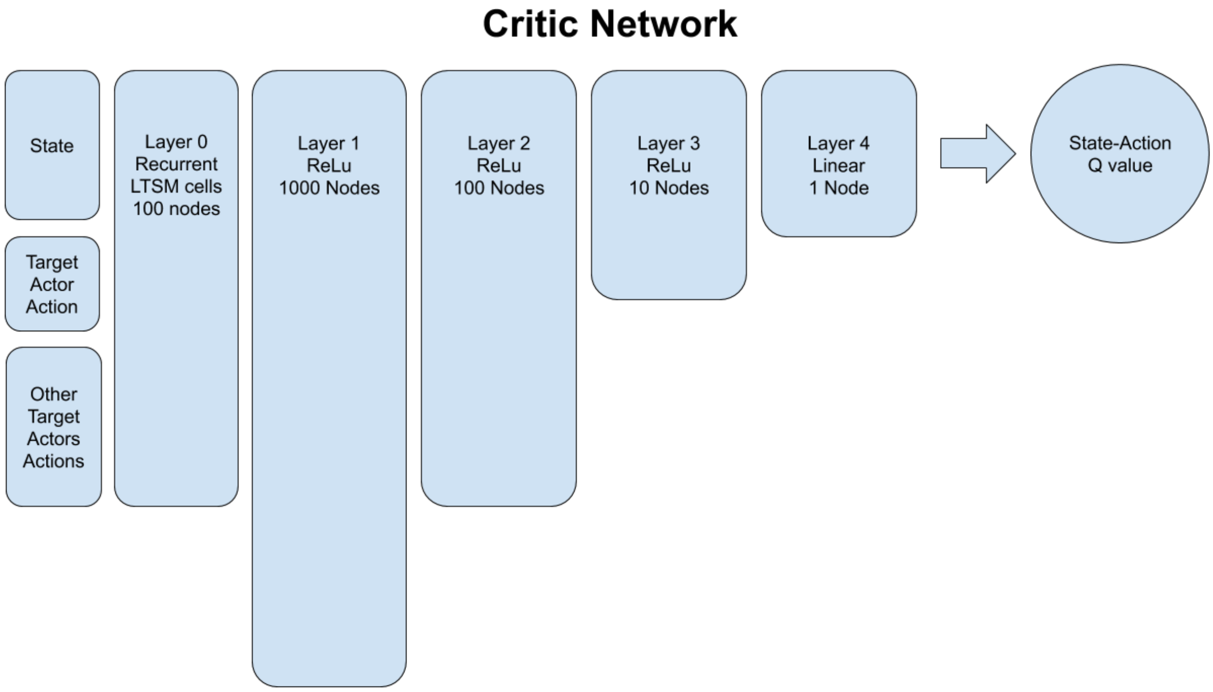


Figure x: Actor and Critic neural networks, respectively

As can be seen, there are 2 main differences in between the actor and critic networks and these rest solely in the input and output layers.

3.3.1 Input Layer

The differences between the set of inputs used in the actor and critic are simple and reflect the differences between these entities in terms of centralization of information. The actors, being decentralized entities have as input solely the observed state of the system. The critics, however, being centralized entities have additional inputs for the actions taken by all actors.

Another fact that affects the inputs of the networks is the objective being pursued. Different objectives rely on different variables of the state being observed. Some objectives rely on a single variable, as if the case of frequency control (input= $\Delta\omega$), while other objectives may require its state being comprised of multiple variables. For example, the cost optimization network uses two quantities as inputs, the respective generator's current output level and the total output of all generators (input= $[Z_i$, $Z_{total}]$). The details of which variables that comprise the state for each objective and the reasons for such are further described in section 3.7.

3.3.2 Output Layer

The output layer, albeit similar between actor and critic, represent entirely different quantities. In the case of an actor network, the output represents the action to be taken by the agent, in the context of this study this is the adjustment to be taken by the generator in its secondary action output. After scaling, such output takes the form of a continuous value in the interval [-0.1; 0.1]. The output being bounded signifies that there are limits to the generators ability to ramp-up/down their power output. In this case, maximum change a given generator is allowed to perform would be 0.1 *pu* per time step. If it was desired to perform tests with different ramp up/down settings, one could perform further scaling and shifting arithmetic operations to the original value produced by the tanh activation function.

The output of the critic network is the estimated quality of the set of state-actions taken by the actor, with regard to the actions taken by the other agents in the environment. This also takes the form of a single continuous value. Unlike the actions taken by the generators, the assessed quality of an action is unbounded, hence the choice of a linear activation function.

3.4 Reward Function Design

As discussed in section 2.6, the reward function plays a pivotal role in the success of a Reinforcement Learning model. More specifically, in multi-objective scenarios where it is decided to combine multiple objectives into a single function, the proportion between each reward component has increased importance. With these characteristics in mind, a collection of guiding principles shaped the design process of reward functions used in single and multi-objective applications:

Be bounded: The finite upper and lower bounds act as points of reference against which rewards can be compared, thus becoming easier to assess the quality of any given reward. Generally the lower bound is 0, while the upper bound is a round positive number such as 1, 10 or 100.

Be at the same scale as the expected state values: As previously mentioned, this is to ensure the the function’s output changes smoothly according to changes in the state.

Function Composition: Craft individual reward functions for each objective and have the global reward be a composition of all individual functions. Such such compositions usually are done by either multiplication or addition.

Monotonically increase: While keeping the adherence to all other objectives constant, increasing adherence to a given objective should monotonically increase the total reward. This is only possible if the objectives are not intrinsically contradictory — that is, by definition, progressing towards one objective entails retreating from the other.

Have a single global peak: A direct result of the previous guidelines. Having the global maxima of all individual objectives reward functions coincide means that the state which provides the maximum reward globally is the same state which provides maximum rewards for all different objectives individually.

For the purpose of streamlining the design process of the reward functions, and conforming to the aforementioned guiding principles, all individual reward functions share the same base function:

$$f(x) = a \cdot 2^{-b \cdot (x^2)} \quad (13)$$

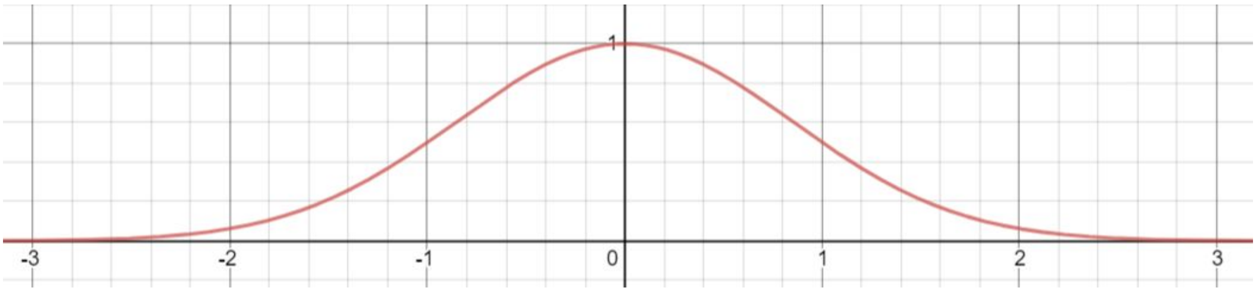


Figure 6: Plot of base reward function $f(x) = 2^{-x^2}$

This function provides some useful characteristics for following the guidelines. It is an even function and thus symmetric with respect to the y-axis, this is useful if the state is encoded in terms of a deviation from a target value (e.g. $x = \Delta\omega$) with the objective to minimize the deviation, as the magnitude of the deviation is the important aspect, rather than its sign. The base function has one single maximum at $x = 0$, this means that composition by either multiplication ($h(x, y) = f_1(x) \cdot f_2(y)$) or addition ($h(x, y) = f_1(x) + f_2(y)$) retains a single global maximum at the origin. The parameters a and b can be used for deforming and scaling while keeping the symmetry and maximum location characteristics instact. These transformations are particularly important in multi-objective reward functions as the proportion between each objectives' rewards plays a significant role in the success of a given reward function.

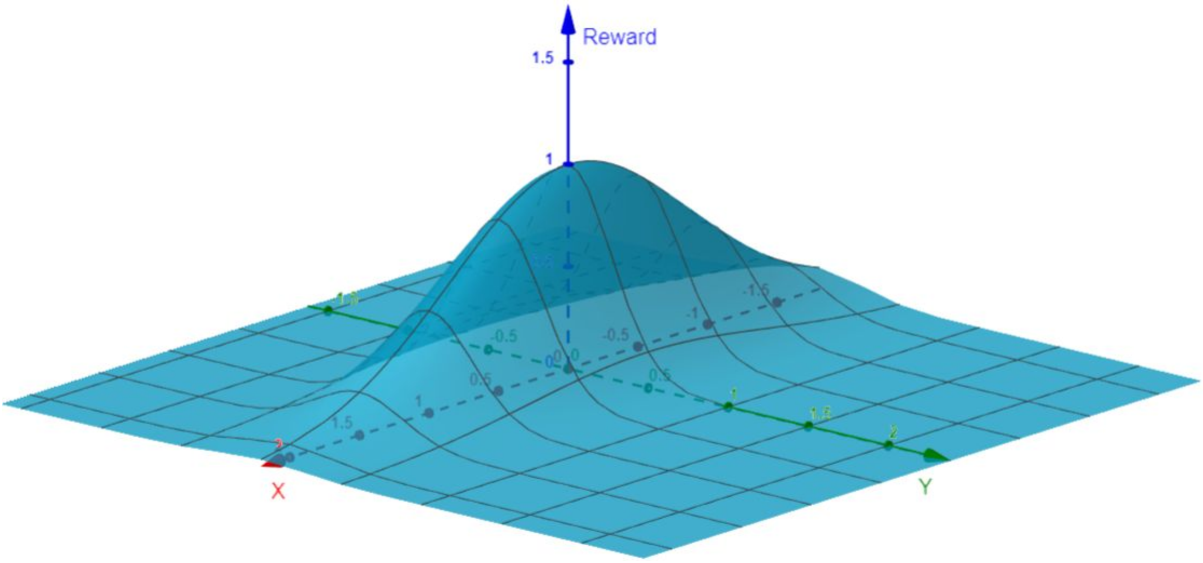


Figure 7: Plot of a sample multi-objective reward function $r(x, y) = 2^{-x^2} \cdot 2^{-10\cdot(y^2)}$

Throughout the design process of the reward functions, this study leveraged the aid of 2D and 3D graphical plots to visualize the individual functions and their resulting compositions. Even in multi-objective scenarios, it is interesting to leverage the use of 2D graphs to garner further understanding of the reward dynamics awarded for each component (i.e. objective) before relying on 3D plots to visualize the joint behaviour of both components with respect to the final reward space. The particular functions used for each experiment and objective are further detailed, along with the rationale behind their choice, in section 3.7 below.

3.5 Multi-Objective

This investigation sets out to test two distinct strategies for obtaining multi-objective optimization: reward-composition and action-composition. The former strives to accomplish it by learning a single policy which is able to fulfill multiple objectives. This, in turn, is achieved by consolidating the adherence to multiple objectives into a single reward function. Said function is purposely designed to contain the hierarchical relationship between objectives. The single value produced by the reward function in this case is already imbued with such preferences. Equation (14) below shows an example of a multi-objective reward function based on two objectives:

$$r(x, y) = 2^{-(x^2)/2} \cdot 2^{-(y^2)/100} \qquad (14)$$

Conversely, the action-composition approach trains multiple single-purpose sets of agents. During execution time, the application then samples actions from all sets of agents, and consolidates the actions into a single action per agent. For a system with I agents and J objectives this composition is expressed in equations (15) and (16) below:

$$A^i = \sum_j^J \rho_j \cdot A_j^i \qquad (15)$$

$$\sum_j^J \rho_j = 1\,,\; 0 \le \rho_j \le 1 \qquad (16)$$

When performing action-composition, the reward functions used for each overarching objective do not intrinsically carry information regarding such preferences. Such preferences are declared in the form of the weight ρ_j assigned for each action component. One important prerequisite for performing action composition is for the action-space to be numerical/quantitative. In categorical action environments, consolidating multiple actions into a single one cannot be done via arithmetic operations.

3.6 Software Engineering

Aside from the experimentation results, one of the overarching objectives of this study concerns the software engineering aspect of data science. In this regard, the implementation and structure of the application was guided to reach the following goals:

- Application should provide ample experimentation power
- Experiments should be easily reproducible
- Diminish the cognitive load required for defining experiments to be run

To this end, particular attention and time were dedicated to producing a codebase which is didactic and easy to understand and expand. The code structure, separating domains into different packages, and each entity in a single file, also reflects the author's years of experience in software engineering and code maintenance.

In didactic terms, the implementation is done in a way to provide separation between different layers of abstraction and domains. The electrical system simulation is contained in a single package which provides means of specifying and instantiating new systems as well as updating said systems via actions, and finally, collecting information regarding the history and current state of a given instance. Likewise, the reinforcement learning aspect of the application is contained in a separate folder, with individual files for each entity. The learning algorithm general flow and update cycle is separated from the detailed implementation of the individual actor and critic neural networks. The episode buffer stores experiences in the form of proper dataclasses, instead of a large matrix of mixed data.

Finally, specific care was given to naming all the variables, entities and functions in such a way that they properly reflect the terminology in both electrical and RL domains.

To facilitate further experimentation, there are two main strategies present in the codebase. Firstly the very environment used in this study should be easily replicated by others. For that reason, the code relies on the PyEnv and PipEnv python libraries to ensure the same python and dependencies versions are locked and reproducible. Additionally, the author opted for performing the experiments using Jupyter Notebooks — a common means of sharing experimentations in the data science field — while the internal workings of the application remain written in plain python. Furthermore, the code is structured in order to enable a declarative approach to running experiments. All the learning parameters are easily configurable via the LearningParams class, as are the electrical system constants via the ElectricalConstants class. The electrical system specifications such as number and configuration of generators and loads are also exposed through the LearningParams class. Taken together, these changes streamline the process of carrying out further experiments under different conditions, which can be readily done simply by changing the relevant values in the Jupyter Notebook. Finally, the code ensures that the trained models are saved in specific folders, as is the information regarding the conditions under which said model was trained (i.e. the LearningParams used). Taken together, these measures help ensure the models are reproducible, avoid loss of information, and minimize costly human errors.

Another example of the effort spent on minimizing human errors is the means by which the epsilon decay was implemented in this study. Instead of manually declaring both decay rates and the threshold value as it is often the case in RL applications, the decay rates are inferred from other user-defined values such as the total experiment duration (episodes x steps per episode), the ε threshold value, and the point during the experiment where this value should be reached (in % of total steps). This is a useful addition from a user experience standpoint as the exact decay values are not as relevant as having the decay happen smoothly and over the course of the complete learning process. Since the experiment duration is a quantity that is frequently changed during the design and test process of a given RL solution, having the ε decay values being automatically calculated as a function of the experiment duration removes the need to continuously adjust the ε decay values according to changes in experiment duration. The practical effect is that such implementation reduces the rate of errors in the specification experiments and, therefore, speeds up the process of reaching a working solution.

This study takes as starting point and draws inspiration from the results obtained by Sergio Rozada, a former student of City, University of London MSc Data Science programme. Although there are definite parallels between both works: both study the use of reinforcement learning techniques for solving the frequency control problem; both studies simulate an electrical system with multiple generators and one load, in a single shared bus; both studies are based on the MADDPG algorithm. However, it is important to address the significant differences and areas in which this study improves upon previous work in breadth, and depth.

In terms of breadth, most of the improvements are a byproduct of the deliberate effort in conducting the analysis through building a high quality codebase, as evidentiated above. This work's implementation has the capability to seamlessly execute experiments with multiple loads and multiple generators, with noise levels individually declared, instead of being bound to a specific electrical system setup, as a consequence, the base scenario in this study uses three generators instead of two. Additionally, the MADDPG algorithm is implemented in a more flexible way. Entirely new models can be created by simply implementing a new *model_adaptor,* specifying the data used as the "state" input of the neural networks and the reward function to used, and running the training process. This implementation also supports partially observable states through improvements the experience buffer: aside from storing episodes in proper data classes, a separate state is stored per agent. Finally, most of the experiment constants, including the neural network shape, are easily configurable.

Regarding depth, this study builds on top of previous work by focusing on solving tertiary control. It proposes two separate strategies of doing so by introducing two distinct multi-objective techniques: reward composition and action composition. Additionally, this analysis relies on different measurements from the system state as inputs for the models.

3.7 Experiments

Built on the foundational methodologies, strategies, and guiding principles stated above, this study set out to perform a multitude of experiments aimed at assessing the feasibility of leveraging multi-objective techniques to perform primary, secondary and tertiary control in an electrical power system. Tables x and y below define the symbols used in the experiment descriptions and the learning algorithm parameters used in all the experiments.

Symbol	Name	Observation
$\Delta\omega$	Delta Frequency	Difference between observed frequency and nominal setpoint. See equation (5)
Z_i	Secondary action of generator i	
Z_{total}	Total secondary action	

C_{total}	Total cost of production	Expressed in \$/h. See equation (6)
ΔP_{total}	Total deviation from the cost optimal output configuration	Calculated as the sum of the individual deviations. Expressed in %. See equation (20)
ΔZ	Output differential	The difference in total secondary action from a given target output. $\Delta Z = (Z_{total} - Z_{target})/Z_{target}$
Z_{target}	Target output	A defined target output

Table x: Definition of symbols used in the state and reward function descriptions

Constant	**Description**	**Value**
γ	Discount factor	0.9
τ	Mixing factor for target network updates. See equation (12)	0.001
ε threshold value	ε value at which the decay rate changes	0.5
ε threshold value	Learning process progress point at which ε threshold value should be reached	0.6
ε final value	ε value at the end of the experiment	0.0001
numEpisodes	How many episodes should the model be trained with	5000
maxSteps	How many steps per episode	200
Replay buffer size	Maximum amount of episodes held by the replay buffer	100
batchSize	How many experiences are sampled from the episode buffer per update cycle	4
traceLength	Size of the sampled experiences (in steps)	8

Table y: Configuration constants used for all experiments

Finally, all experiments were run using the following cost profiles for the generators:

$$C_1 = 510.0 + 7.7 \cdot P_1 + 0.00142 \cdot (P_1)^2 \quad (17)$$

$$C_2 = 310.0 + 7.85 \cdot P_2 + 0.00194 \cdot (P_2)^2 \quad (18)$$

$$C_3 = 78.0 + 7.55 \cdot P_3 + 0.00482 \cdot (P_3)^2 \quad (19)$$

These parameters fall in line with values applied in the industry (Wood, A. and Wollenberg, B, 1996), Furthermore, these parameters and the electrical system configuration were explicitly chosen to ensure that, in the total load interval studied (~300 MVA), the optimal setup is such that so no generator is in either minimum or maximum output values. This helps evaluate the ensuing results as successful models should be able to steady the outputs around given values, rather than relying on the enforcement of minimum/maximum limits and blindly move towards these boundaries.

3.7.1 Experiment I - Frequency Control

State Inputs: $\Delta\omega$

Reward Function: $r_I(\Delta\omega) = (9 \cdot 2^{-(\Delta\omega^2)/2} + 2^{-(\Delta\omega^2)/100})/10$

The first experiment ran was to perform frequency control on an initially unbalanced electrical system in the form of primary and secondary control. This serves the dual purpose of i) reproducing the initial results obtained by Rozada (Rozada, 2018), and ii) in doing so, establishing the correctness of this study's implementation of the MADDPG algorithm.

Naturally, these specifications are quite similar to the ones in the original system that proved to successfully perform primary and secondary control in previous works. The environment state is distilled into a single input in the form of the frequency deviation from the nominal setpoint ($\Delta\omega$) as defined in equation (5). The reward function also draws inspiration from previous work, but tweaked to conform to the base function described in equation (13). Overall this experiment is expected to produce similar results, with the trained agent being able to properly balance the load over time.

3.7.2 Experiment II - Frequency and Cost - Reward Composition

State Inputs: $\Delta\omega$, Z_i , Z_{total}

Reward Function: $r_{II}(C, \Delta\omega) = f(C) \cdot g(\Delta\omega)$

$$f(C) = 2^{-(C_{total}^{2})/50}$$

$$g(\Delta\omega) = 8 \cdot 2^{-(\Delta\omega^2)/2} + 2^{-(\Delta\omega^2)/100}$$

This experiment follows the reward-composition strategy described in section 3.5. Each generator is associated with a single agent which produces actions striving to accomplish both objectives (frequency control and cost optimization). To this end, a single reward function that reflects both objectives was crafted following the guidelines set in section 3.4.

The reward function of choice was a product of two reward components aimed at optimizing for each objective. The cost component — $f(C)$ — strives to minimize the cost, and is expressed in terms of the total cost of production following the base reward function. The frequency component — $g(\Delta\omega)$ — seeks to minimize frequency deviation and is similar to the function used in experiment I, but with a twist. Instead of using the base reward function as is, it expresses the frequency component in terms of a weighted sum of two separate subcomponents — peak and base. The most significant subcomponent ($8 \cdot 2^{-(\Delta\omega^2)/2}$) represents the peak subcomponent and provides the frequency component with a steep peak, which has the effect of prioritizing gains in the frequency component over gains in the cost component. While the least significant ($2^{-(\Delta\omega^2)/100}$) is the base component, which has the sole intent to speed up the learning process by providing a soft slope towards the central peak. The base component is required as a side effect of the peak component. By enforcing a steep slope near the origin, the peak component produces negligible rewards for larger values of $\Delta\omega^2$, additionally it has a near flat slope in that region of the domain. Although learning would still happen, a larger amount of learning time would be dedicated at aimlessly exploring that region of the state space. The presence of a slope in said region of the reward function, albeit a smooth one, effectively helps speeding up the learning process. This reward function is a good example of how parameters a and b are manipulated to shape the global reward function as a composition of individual base functions.

3.7.3 Experiment III - Frequency and Cost - Action Composition

Set [1] - Frequency:

State Inputs: $\Delta\omega$

Reward Function: $r^1_{III}(\Delta\omega) = (9 \cdot 2^{-(\Delta\omega^2)/2} + 2^{-(\Delta\omega^2)/100})/10$

Set [2] - Cost:

State Inputs: Z_i, Z_{total}

Reward Function: $r^2_{III}(\Delta P_{total}) = [2^{-(\Delta P_{total}{}^2)/100} + 9 \cdot 2^{-(\Delta P_{total}{}^2)/2}]/10$

This experiment is aimed at testing the action-composition strategy described in section 3.5. To this end, it trains two sets of agents, one for each overarching objective.

The first set, aimed at balancing the system load, is in fact the same model trained in experiment I. The second set is trained with a single objective reward function aimed at finding the minimum cost of production for every total output. It is trained by beginning episodes with different output combinations and finding output set that minimizes the cost of production while keeping the total output level constant. This is done by calculating the individual power levels (P_0^i) that minimize the cost for the total output observed at the initial episode configuration ($C_0^{total} = C_0^{min}$). Those values are then used throughout the episode to calculate the total power deviation (ΔP_t^{total}), as shown in equation (20) below:

$$\Delta P_t^{total} = \sum_i^I abs(\frac{P_t^i}{P_t^{min}} - 1) \qquad (20)$$

The objective of the model trained in set [2] is then to minimize the sum of the individual generators output deviation from the cost optimal setup, given a total output. In doing so, the individual generators outputs will approximate the cost optimal setup for that given output. By beginning episodes with different combinations of secondary actions, the training process is able to explore the state space and learn the cost optimal output combination for multiple different total outputs.

After finishing both training processes, the application has two sets of agents available, set [1] produces actions that eventually balances the electrical system, while set [2] produces actions that guide the individual output levels toward the cost optimal setting. In this setup, execution of the application samples, for every step, actions from both sets of agents, and combines these actions into individual actions by means of a simple weighted sum. One important point to note is that while set [2] is trained using a fixed target output per episode, during execution the value used as target output is the current total output. By virtue of integrating actions from set [1], the total output is expected to change over time while the system is being balanced, and remain constant once electrical equilibrium is reached. This affects the samples taken from set [2] in the sense that at every step, it provides actions with the intent to optimize the cost of production for the power output observed at that point in time. Furthermore, by virtue of set [2] being trained to minimize costs given an output, it is expected that the actions sampled from this set to produce net zero change in power. This was done intentionally in order to minimize interference between the actions from both sets. While set [1] produces actions with the necessary changes in overall output to properly balance the system, set [2] produces actions which do not change global output, but rather swaps output levels between the generators so that the final setting is more cost effective.

4. Results

4.1 Experiment I - Frequency Control

The first experiment's results are promising. After about 20-30 steps, the load was successfully balanced and the power output and system frequency, in general terms, remains stable. This indicates that i) the implementation of the MADDPG algorithm is properly done and works as expected, ii) the results obtained by Rozada are reproducible and iii) the chosen algorithm works for 3 generators, which further indicates it should work with any number of generators.

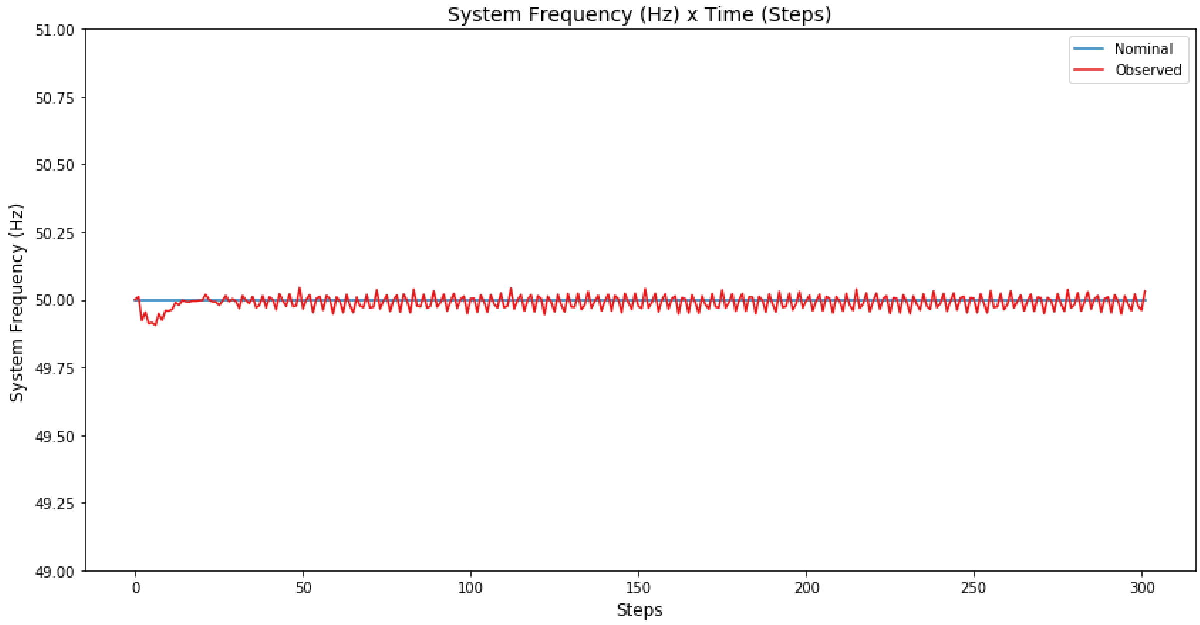


Figure 10: Experiment I - Observed Frequency, test episode progression

If we are to observe closer, at the frequency deviation, it is possible to assess that there still remains a steady state error. The frequency still oscillates within 0.05 Hz (0.1%) of the nominal setpoint. While this value may seem low, in practice it would actually be regarded as too high — normal frequency deviation and AGC control range falls within 0.02 Hz of the nominal value (Kirby, B. J., 2003). This indicates that, although the objective is fulfilled in general terms, upon further scrutiny, it still falls short of the standards deemed as acceptable in the industry. It should be noted, however, that increasing training time and further tuning the reward function by providing increasingly steeper peaks has a positive effect in reducing this steady state error. It is then expected that further tuning could diminish this error until it is within the acceptable standards.

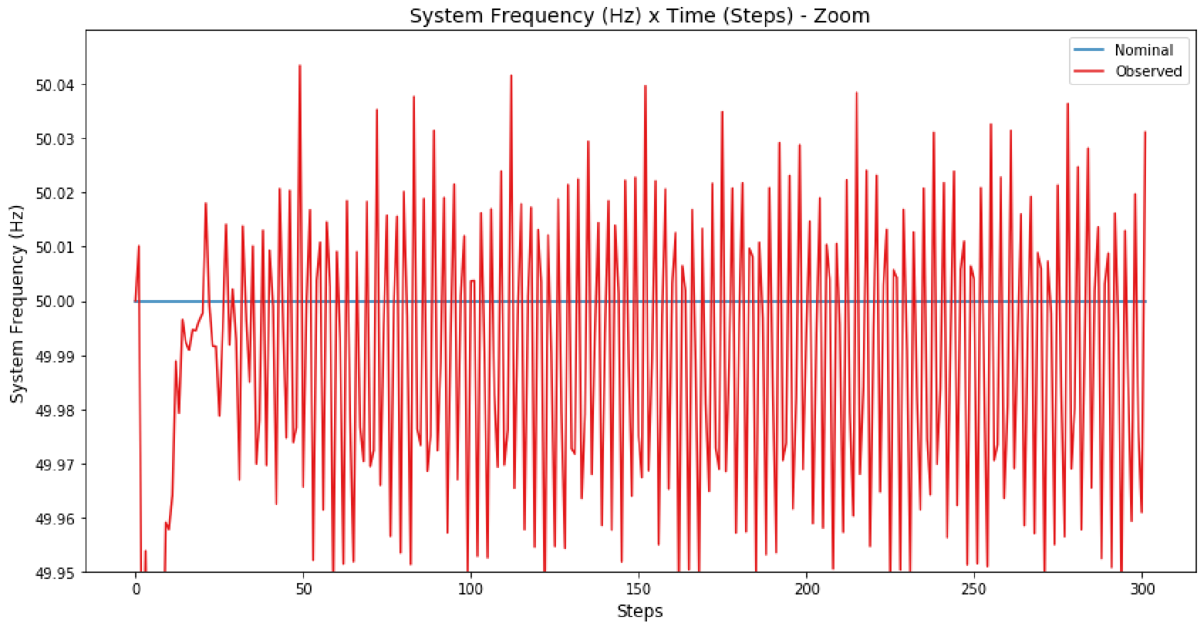


Figure 11: Experiment I - Observed Frequency (zoom), test episode progression

The relative success of this model can be explained by two main components. Firstly, the reward function seems to be properly crafted. By observing the earned reward progression compared with the observed frequency progression it becomes clear that a) the reward increases as the test episode progresses, stabilizing near the maximum value (1) in the first third of the episode; b) the frequency stabilizes near the nominal value simultaneously as the reward reaches the maximum plateau. Together, these observations indicate that the reward function properly reflects the goal, which is to minimize the frequency deviation from the setpoint.

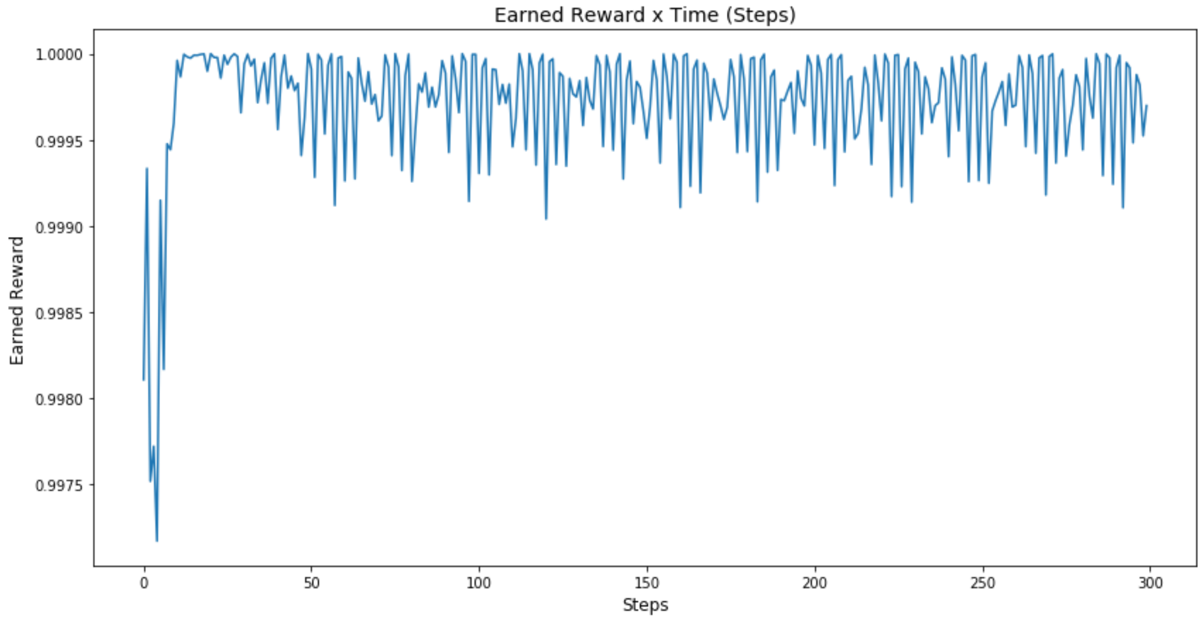


Figure 12: Experiment I - Earned Reward, test episode progression

The second component which explains the success of this model is the algorithm of choice: MADDPG. Agents trained with this algorithm successfully coordinate their actions to reach a common goal. This can be seen by perusing the power output per generator during the course of the experiment. In this case, it appears that the learned strategy seem to be to have two generators reach their minimum output as fast as possible, therefore providing a stable base output, while the third generator controls its output to stabilize the system gradually reducing the steady state error.

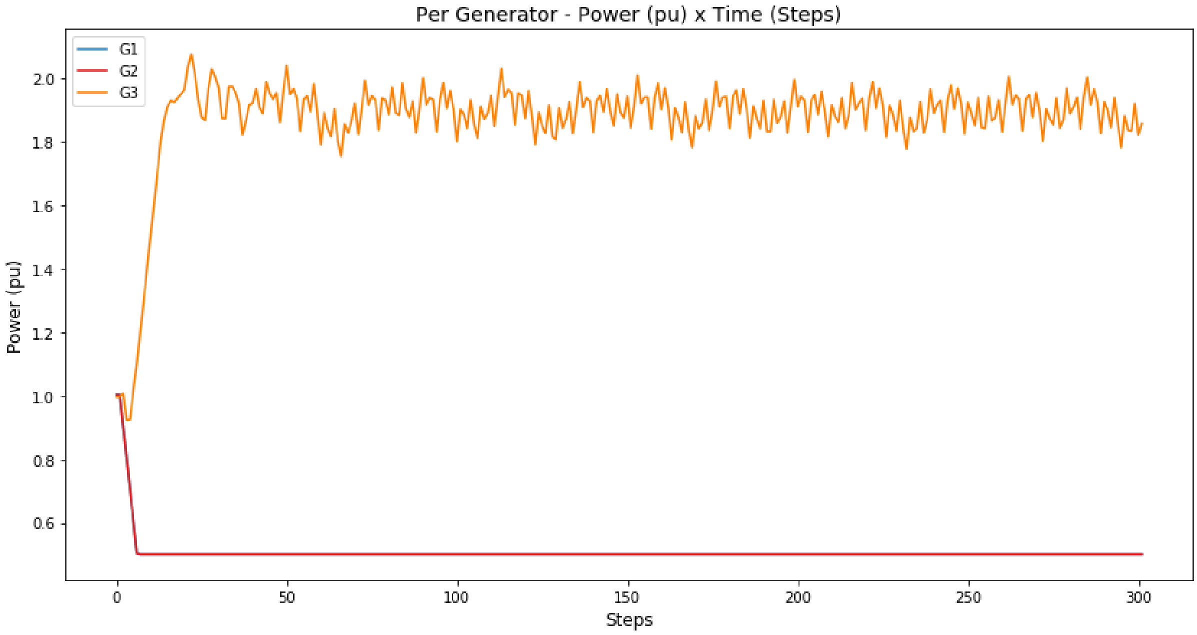


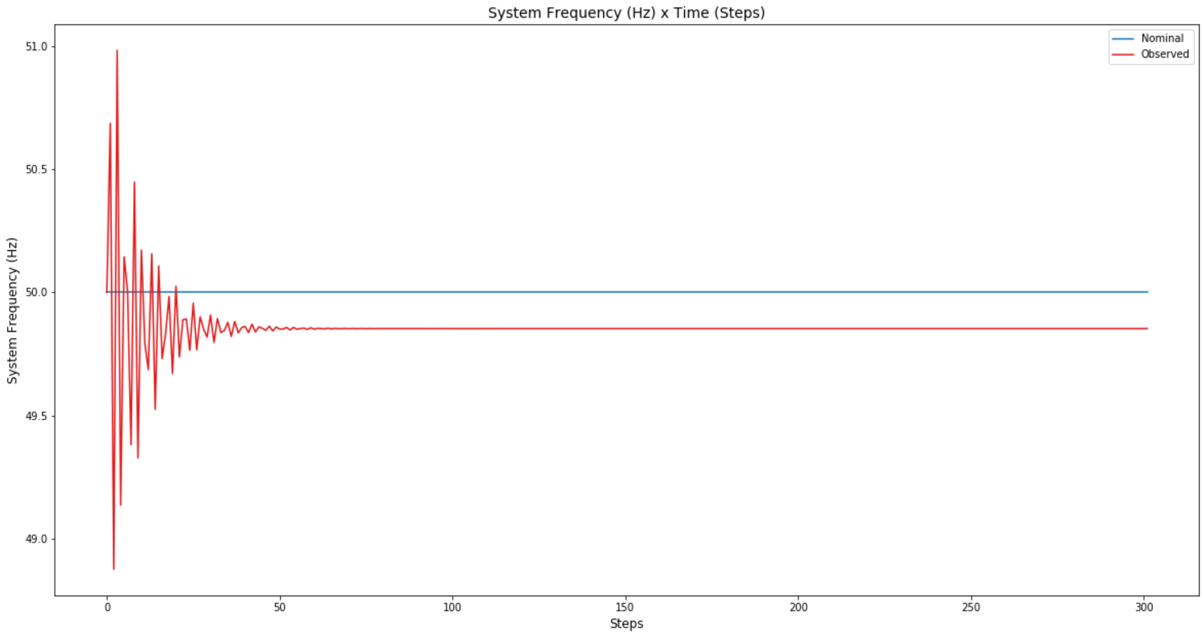
Figure 13: Experiment I - Per generator output, test episode progression

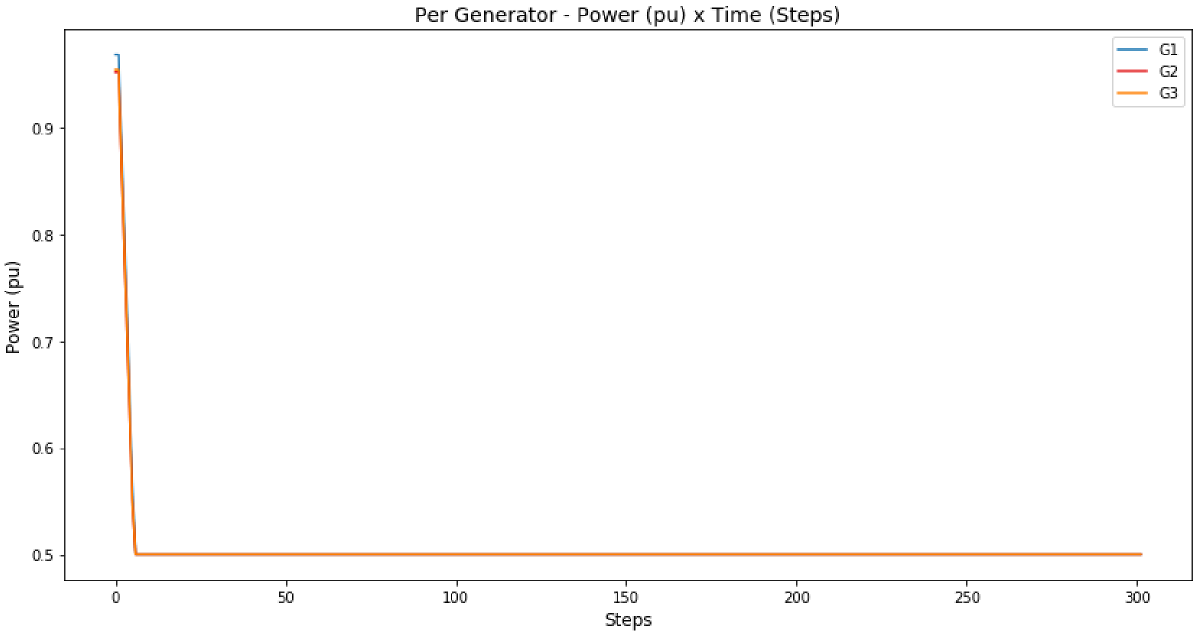
Regarding the strategy learned by the model to perform frequency control, one should note that, even though the final objective is partially fulfilled, this “cooperation by omission” approach does not appear to be the most efficient nor the fastest way to balance the system. In the experiment results, this can be seen in the initial steps as the frequency deviation initially increases before eventually converging near 0. One possible reason for this type of cooperative behaviour could be that reaching the maximum/minimum limits may be the best way to ensure stable output for the other generators, as these limits are enforced in the simulation, and not in the modelled neural networks themselves (i.e. once the secondary action reaches whichever limit, the neural network is still able to issue commands to go beyond such limits, but they are disregarded by the electrical system simulation). Therefore the way these limits are enforced may be one of the root causes of this strategy surfacing, and it would be interesting to observe the effect of changing the way these limits are enforced on the trained strategy. Another possible explanation could be that the total loads observed during training were such that a single generator had power amplitude large enough to properly balance the system, perhaps training with more diverse loads that better cover the full spectrum of the system's total power capacity would lead to more complex and robust cooperative strategies. Finally, one could introduce in the reward function further incentives for more complex cooperation by increaser the reward proportionately to how fast the system is balanced.

On a further note regarding this cooperation by omission, it should be noted that the generator which is elected to effectively perform the balancing seems to arbitrary. Rerunning the exact same experiment multiple times result in different generators being elected by the model to perform this role. This randomness is somewhat expected as all generators are identical with respect to their output capabilities. However, while the impact this choice is nonexistent for the task of enacting frequency control, this characteristic has repercussions when this model is combined with a cost optimization one to perform action composition, as will be shown in experiment III.

4.2 Experiment II - Frequency and Cost - Reward Composition

The results of this experiment show how the scaling between the reward function inputs and the function's underlying characteristics have an increased significance when shaping a reward function for two objectives. In this case, relying on the same function as in experiment I for the frequency component of the global reward results in a poor performing model. Even though the shape of the reward function gives increased importance to minimizing frequency deviation rather than cost, the trained model still fails to properly balance the system. All generators reduce their output to their minimal value, and the only factor acting to balance the system is the droop control. This results in a steady state 0.2 Hz far below the nominal setpoint and well over the acceptable deviation.





Figures 14 and 15: Experiment II-a -Frequency and Per Generator Output, test episode progression

This behaviour is a direct consequence of two factors: the combination of both goals into a single reward function and an indirect relationship between both goals. By melding both objectives into a single function which produces a single numerical output, there is an inherent tradeoff between both objectives (i.e. there exists situations in which the total reward is increased by advancing one goal while hindering the other). As to the indirect relationship between the goals, both are linked by the total output of the generators. In a perfectly balanced system, it is a corollary that the total power output must be equal to the load. Additionally, for decreasing the total cost of production, there are two possible methods: 1) Change the operating output of all generators that results in a lower cost of production while keeping the same total output — this keeps the system balanced and can be done until the optimal setup is reached. 2) Simply lower the total output — this can be done indefinitely, however at the cost of breaking the electrical balance. The reward function, as designed, rewards both methods. The observed behaviour can, therefore, be explained by the reward function being such that sacrificing electrical balance for the sake of reducing costs still achieves considerable rewards.

Delving deeper into this oblique interdependence between the objectives, it should be noted that the guidelines used only provide general directions to craft a reward function. Furthermore, the shape of the reward function may suggest that the highest possible reward lies at the origin, however in this case this is not true because collection of physically possible states is a subset of the surface of the reward function. In essence, there are points in the surface of the compound reward function that are physically impossible to be reached.

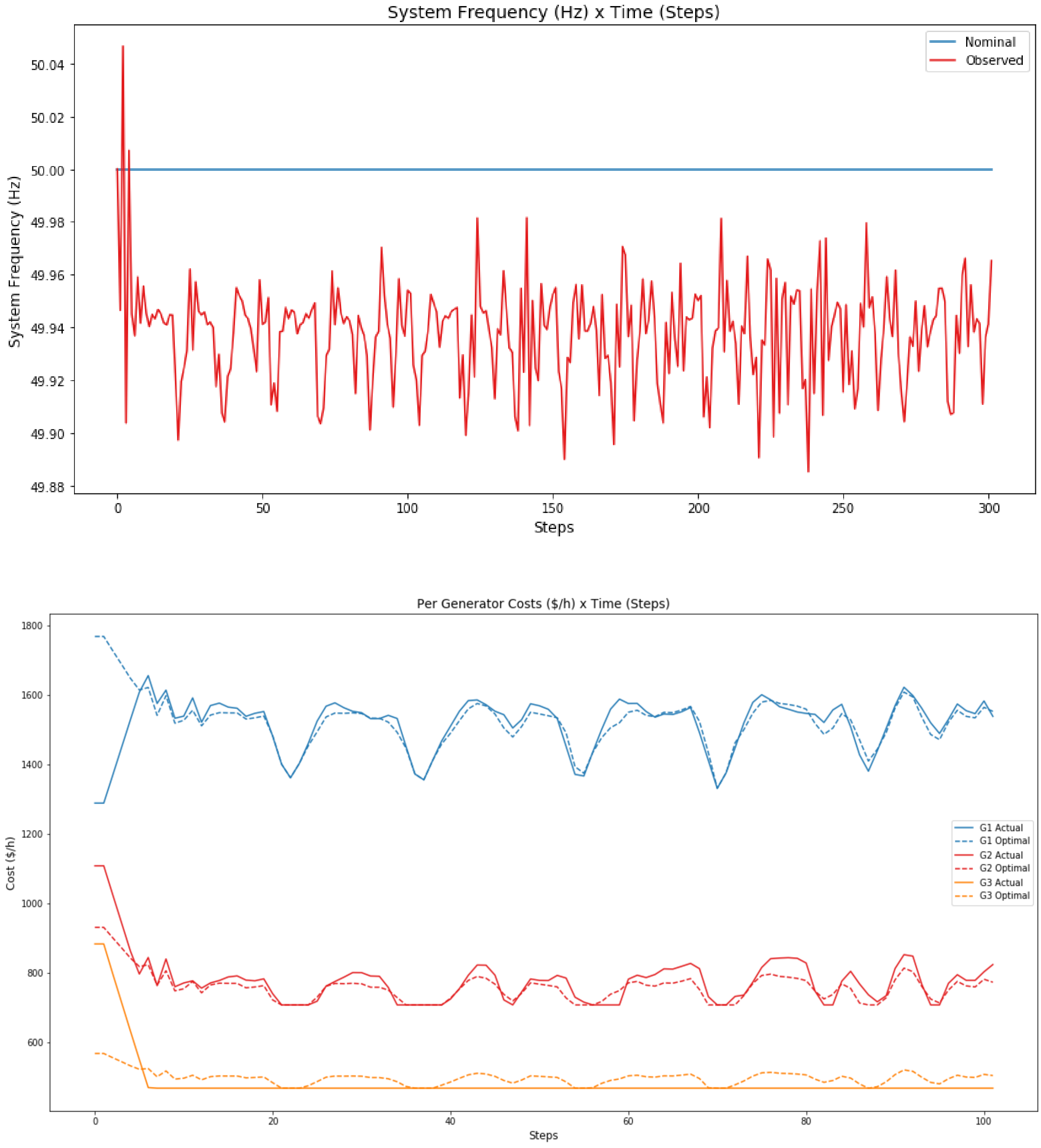
Further tuning the reward function could render the steady state error arbitrarily small by increasing the weights associated with the frequency component. In practice this would mean shaping the frequency component into even steeper peaks and awarding increasingly low rewards for any deviation from the nominal value. This methodology could lead to satisfactory results in practice, however it would fail to address the issues created underlying relationship between the objectives. The downward bias in total output caused by the cost component would still exist, only rendered arbitrarily small. In light of that, instead of performing further experimentation with longer training periods or reshaping the reward function, it was chosen to redefine the multi-objective reward function in a way that breaks this spurious relationship.

Instead of shaping the cost component to simply minimize the total cost, it was decided to declare it as a function of the total power deviation from the cost optimal setup (ΔP_{total}) as seen in equation (20). This leads to a simple, yet profound change: the only way to advance towards the cost objective is to reconfigure the outputs in a way that is optimal. Simply lowering the total output no longer increases the cost component of the reward. This, in turn, has the practical effect of breaking the indirect relationship between both goals (it is possible to advance in one whilst keeping the other constant), and ensures that the origin is a reachable state. That is, the highest possible achievable reward is one in which the frequency is exactly equal to nominal, and the cost is the lowest required to achieve said electrical balance. The revisited reward function is declared in equations (21) through (23). The results of the experiment can also be seen in figures 16 and 17 below.

$$r_{II-b}(\Delta P_{total}, \Delta\omega) = f_b(\Delta P_{total}) \cdot g_b(\Delta\omega) \quad (21)$$

$$f_b(\Delta P_{total}) = 2^{-(\Delta P_{total}^{\ 2})/4} \quad (22)$$

$$g_b(\Delta\omega) = (9 \cdot 2^{-(\Delta\omega^2)/2} + 2^{-(\Delta\omega^2)/100})/10 \quad (23)$$



Figures 16 and 17: Experiment II-b - Frequency and Per Generator Output vs Optimal, test episode progression

The results indicate progress. Generators G1 and G2 no longer operate at the lowest possible output. Instead the follow somewhat closely their optimal outputs for the given total output at any given point. While G3 exhibits similar behaviour to the previous iteration, this could be interpreted as being associated with it being the least cost efficient generator of the setup, and its optimal output being close enough to the minimum that the model as a whole benefits less from G3 actively attempting to follow the optimal value than it does from the decrease in entropy it causes by keeping the output constant at the minimum level.

Despite the generators being able to closely follow the cost optimal levels throughout the episode, one still needs to address that matter of frequency balancing. In this regard, it still performs worse than the single objective model seen in experiment I. In this case the model still exhibits a slight downward shift of approximately 0.05Hz, a shift considerably smaller than the previous iteration of the model but nevertheless consistently present. The manifestation of such bias is unexpected as the changes applied to the cost component input supposedly sever the indirect relationship between cost and frequency control. Although the present experiment was not able to pinpoint the root cause of this bias, it is suggested to be associated with the addition of the cost component to the reward function. Further work should be done to uncover the precise origins of this bias and what further mitigation can be done to overcome it.

4.3 Experiment III - Frequency and Cost - Action Composition

This experiment is somewhat different than the previous ones. Instead of training and testing a single set of models, two different models are used to compose the actions taken by the agents. As such, there are two layers of evaluating the results. The models are first assessed individually on how they perform their assigned tasks. Afterwards, the strategy as a whole is appraised by observing the effects of their joint effort in simultaneously satisfying both objectives.

4.3.1 Individual Performances

For attempting to perform cost-effective frequency control, this solution uses one model trained to perform frequency control, and another to optimize the production cost for a given total power output. For the role of the frequency model it was decided to reuse the same model discussed in section 4.1, as such, its individual performance has already been discussed in the relevant section. The cost model, however, is a new one and trained as specified in section 3.7.3.

The initial results indicate that, in rough terms, the cost model is able to perform somewhat as desired. For different initial power configurations, the agents are able to change their individual power outputs towards the optimal setup. As evidence that learning is taking place, one can point to how all generators monotonically change their outputs towards the optimal setpoint prior to reaching the steady oscillatory state. This can be seen as G1 initially increases its output while generators G2 and G3 start the episode by decreasing their outputs.

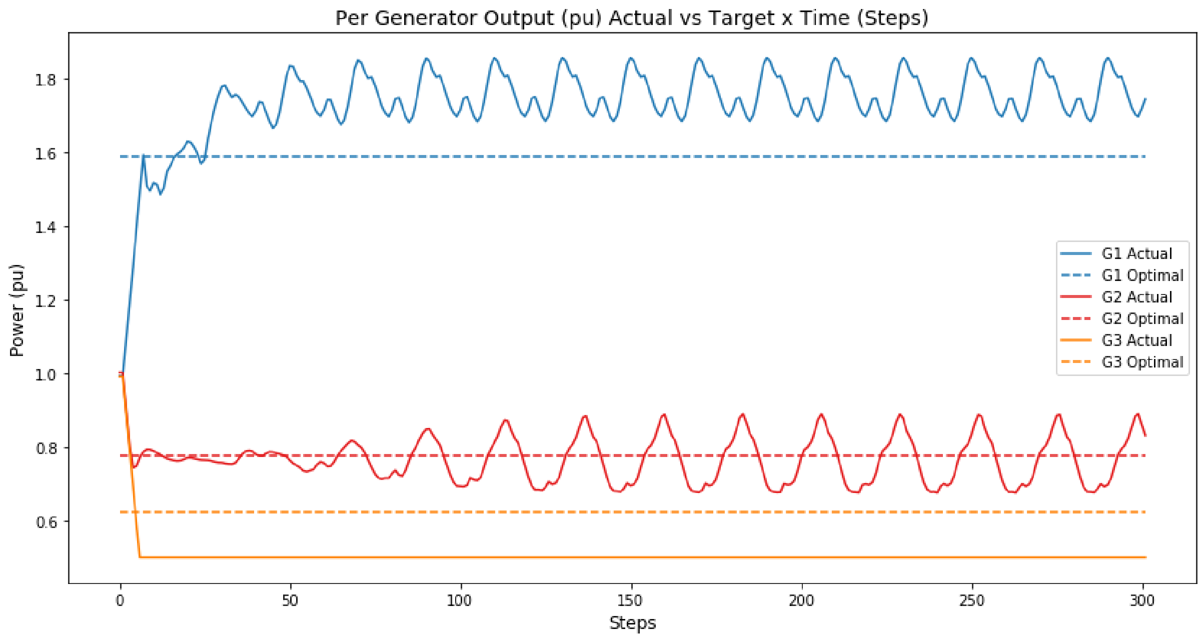


Figure 18: Experiment III-Cost Model - Per Generator Output vs Target, test episode progression

As can be observed, Generator 3 relies on the enforced limits to keep its output constant at the minimum value. Generator 2 has some oscillation but is centered around it’s desired optimal value. Finally, Generator 1 has an oscillatory pattern similar to G2, but offset above its desired value. These deviations as oscillatory patterns are too large for use in a production system.

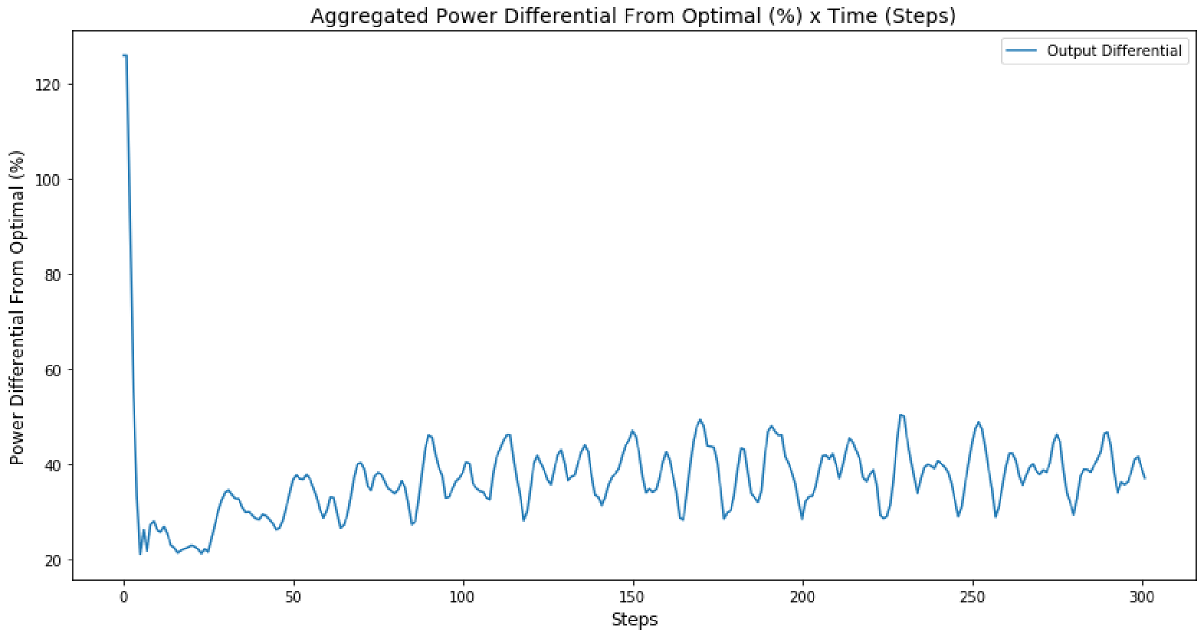


Figure 19: Experiment III-Cost Model - Total output differential from target, test episode progression

One form of measuring the model performance holistically would be to measure the total power differential from optimal as calculated in equation (20), that is the very value used as input in the reward function which the model aims to minimize. As can be seen in figure 16 above, there is significant improvement in that regard. The total deviation of the generators outputs from their optimal values start at over 120% and decreases until stabilizing at an oscillatory state between 50% and 20%. As discussed above, even though this shows some progress, these values are still far from being considered fit for purpose — the deviation is still too large.

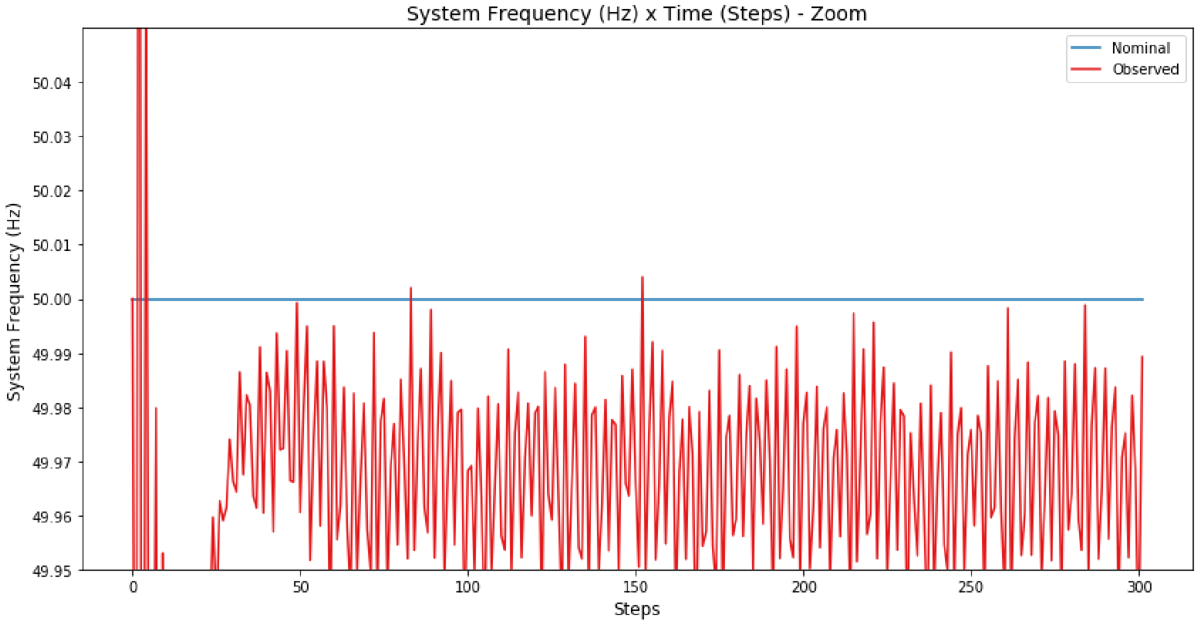
As indicated above, even though there are evidences of learning taking place and that, in general terms the model performs the task it was designed for, the degree to which this task is fulfilled still falls short from being acceptable as is in an industrial setting. In summary, the deviation from the optimal values are still too large, as is the amplitude of the steady oscillatory state. However, it should also be noted that, as was the case in experiment 1, further tuning the reward function constants and prolonging the training period had an observable effect in mitigating these behaviours. One should expect further work on this regard to produce increasingly better results.

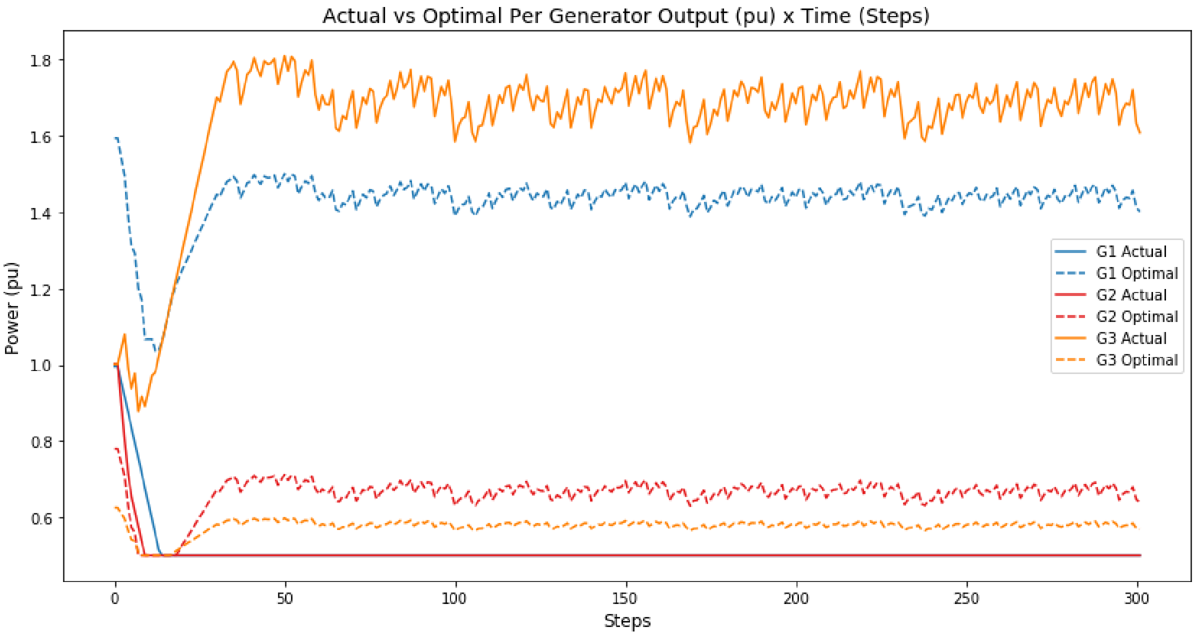
4.3.2 Joint Performance

Even though the individual models performances can be used to inform the final results, the action composition approach should be ultimately evaluated with respect to the joint performance of combining both models. In that regard a few tests were performed with different combinations of weights assigned to the frequency and cost models. These tests have surfaced some interesting behaviours which are further described below.

Frequency Dominant

In this experiment the individual actions taken were calculated using a mixture of 70% incoming from the frequency model and 30% from the cost model ($\rho_{frequency} = 0.7,\ \rho_{cost} = 0.3$). Initially, one would expect this composition to result in harmonious balance between both models. However this is not the case. As observed in experiment I, the trained frequency model relies basically on a single generator to provide most of the output and change its output to gradually balance the system. Furthermore, in this particular instance of the trained model, the generator elected for that role was G3, which also has the characteristic of being the least cost efficient generator among the set. Together, these characteristics result in a clashing behaviour between both models, as can be seen in figures 20 and 21 below.





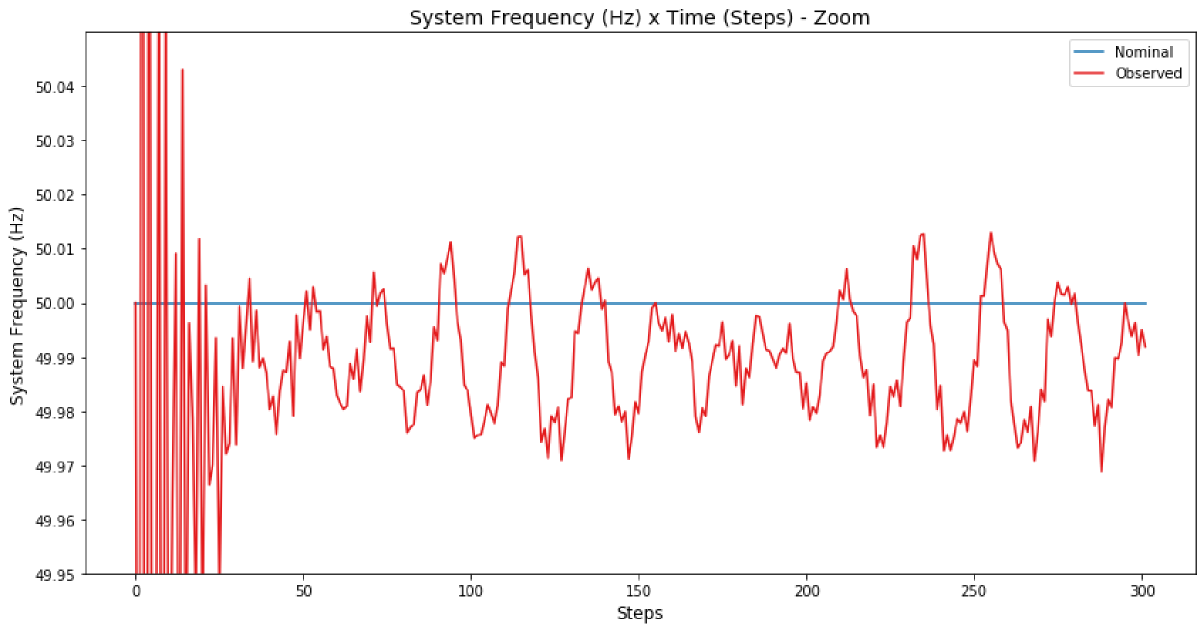
Figures 20 and 21: Experiment III (Frequency Dominant) - Frequency and Per Generator Output vs Optimal, test episode progression

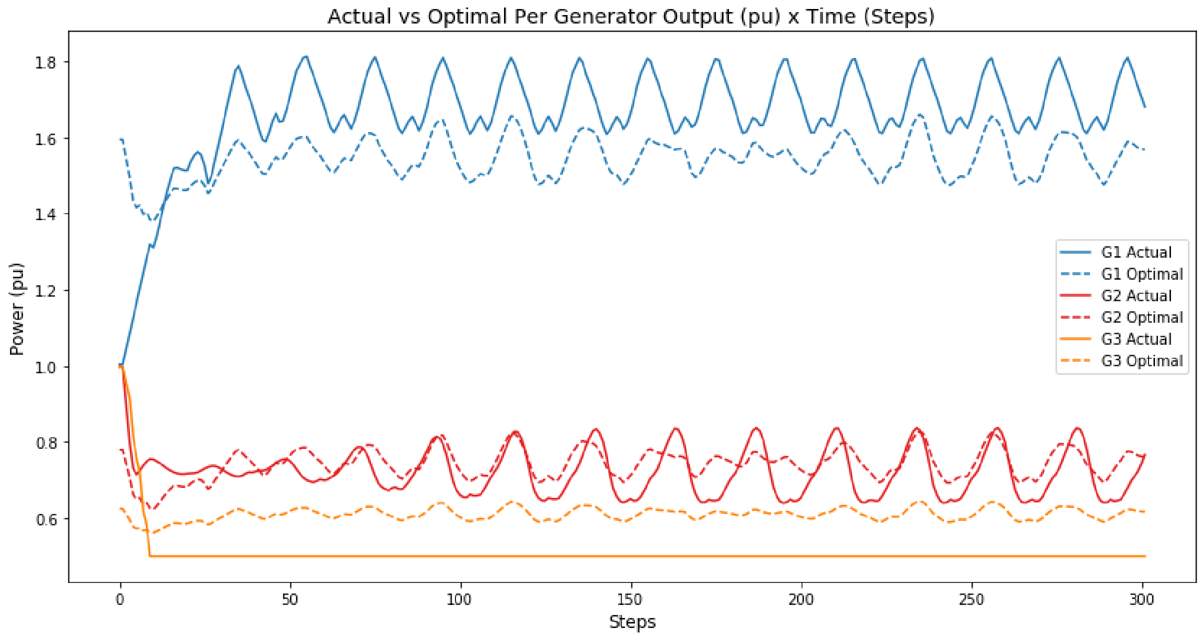
In summary, the cost and frequency action produce contradictory values. For generators G1 and G2, the frequency model simply acts to reduce the power indefinitely, relying on the enforcement of the minimum floor. The mixing weights in this case are such that the frequency model continuously overrides the actions issued by the cost model, resulting in a behaviour much like in the frequency only model discussed in section 4.1. Generator G3, however has a uniquely interesting behaviour. Initially it rises, much like in the frequency model, as it approximates the output which would balance the system, the frequency model issues increasingly smaller actions to perform the fine grained balance of the system. However, the cost model continues to issue actions to dramatically lower G3's by virtue of it being the least cost effective generator and having an output significantly above its optimal value. These divergent actions eventually reach an equilibrium at a point which the frequency is far enough from the nominal so that the magnitude of the frequency and cost actions are counterbalanced. The final result is a downward shift in the observed frequency. The system observes a steady state with and oscillatory amplitude similar to the one produced in experiment I, but centered around 0.03Hz below the nominal frequency.

Cost Dominant

This test is a mirrored version of the previous one. In this instance, the final actions are composed using a ratio of 30% from the frequency model and 70% from the cost model (

$\rho_{frequency} = 0.3,\ \rho_{cost} = 0.7$). This change in weights results in the system performing largely as intended.





Figures 22 and 23: Experiment III (Cost Dominant) - Frequency and Per Generator Output vs Optimal, test episode progression

In order to explain why this change in weights is able to produce better results, one should recall the original design intentions of the cost model. The reward function employed aimed at issuing actions that move the system closer to the cost optimal outputs for that same given total output. As such, a perfectly trained cost model would issue actions that produce a net zero change in the total output of the system. Balancing an electrical system entails changing the total generation to match the total load. In that sense, the actions issued by the cost model produce no effect towards balancing the system. Conversely, the frequency balancing model is trained to issue actions precisely to that end. The final result result is such that the system is balanced within ±0.03 Hz of the nominal setpoint, while the power output levels approach those that lead to minimum cost of production.

5. Discussion

5.1 Results Summary

As can be seen, both approaches are able to approximate the desired behaviour to some extent. The frequency remains within 0.8Hz and 0.5Hz of the nominal value in the reward composition and action composition models, respectively. Meanwhile, although the individual output values largely follow the optimal ones throughout the testing episodes, there is still room for improvement which could be addressed with further training.

5.2 Reward Composition vs Action Composition

This study proposes two different methods of achieving multi-objective learning, reward composition and action composition. In the experimental results, the action composition approach presented superior performance in balancing the frequency, while both strategies had similar results with respect to finding the cost optimal setup. The increased flexibility provided by action composition facilitated the overcoming of the objective divergence which manifested itself in both strategies.

Observed performance is generally regarded as the defining single factor when comparing techniques, especially in a theoretical setting. In the industry, although observed performance remains an important indicator, when implementing systems in the industry there are a number of other factors that are taken into consideration when choosing a technique to be implemented. Such aspects include, but are not limited to, ease of implementation, ease of maintenance, computational resources requirements. In that sense, it can be argued that the action composition approach is superior form a systems design standpoint. Among the benefits provided by this strategy, one can single out the following:

Separation of Concerns: A fundamental principle in software engineering. Breaking down the global model into smaller, single objective ones results in decreased coupling between the models, facilitates the reuse of individual models, and simplifies debugging.

Simplified Modeling: As was observed in this study, crafting bespoke multi-objective reward functions is a time consuming enterprise. Even armed with the guidelines to narrow down the vast blank slate of possible function compositions, it is a process which involves some amount of trial and error due to the delicate balance of ratios between the rewards. If possible, breaking down into single objective rewards should speed up since the reward functions behave in a more predictable way in single objective scenarios.

Variable Priorities: Declaring the objective priorities at the action composition state means that these priorities can be seamlessly changed, even during the course of single execution. Furthermore, finding the optimal priorities ratio can be done faster as the test feedback loop is tighter — the models are pretrained therefore testing a ratio involves just running the test episodes, which is orders of magnitude faster than retraining the models.

Separate Data Sources: A consequence of separation of concerns and variable priorities. Individual models can have different inputs, if different objective of the system are associated with different SLAs, the information sources which provide the inputs can be designed to match these SLAs. If all objectives are joined into a single model, all inputs are necessary to sample the actions, therefore all inputs would have to provide an SLA that is compatible with the most critical objective. Using the studied scenario as an example, balancing the system frequency is critical at all times, while optimizing for cost albeit still important is something that can be overlooked in critical situations. If those objectives are tackled by individual models, the inputs for balancing the system (frequency deviation) should be kept available and real time at all times. Conversely, the inputs for optimizing the cost (individual and global secondary actions) can have its requirements relaxed — if they become offline, the system still can be operated at a degraded level by relying only on the frequency balancing model. This is not be possible if both objectives are tied into a single model which takes as input the intersection of the individual objectives inputs.

For the reasons mentioned above, in situations where both strategies present comparable performance, the action composition approach would be preferred. Furthermore even in some situations in which the performance is lower than the alternatives, this technique may be preferable due to better fulfilment of those non-functional requirements.

5.3 Objective Divergence

Both reward composition and action composition techniques are susceptible to divergence, that is the objectives produce conflicting incentives which negatively affect the model's observed performance. In the action composition approach, divergence was manifested in the form of conflicting actions issued by each of the two models. In this scenario, such conflicts may be more evident as one is able to inspect the individual actions issued by each model and measure their disagreement. In the reward composition approach, as was seen in section 4.2, such conflicts manifested in the form of a downward bias in the total output, shifting the observed frequency below the nominal setpoint. Circumventing these issues may require reframing the manner in which the overarching objectives are distilled down into reward functions. Experiment II took steps to overcome this obstacle by relying on a function which strives to minimize the individual generators output deviation from the cost optimal setting, rather than simply trying to minimize the absolute cost. The final result, however still presented the down bias to some degree — this behaviour should be subject to future studies. Meanwhile, experiment III circumvented this issue by changing the weights used to compose the final actions.

5.4 Points of Improvement

In reviewing the work done throughout this project, there are some instances in which changes could have been made to generate better final results and strengthen the conclusions taken away from it.

The first point of improvement would be the performance of the neural network implementation. In implementing the MADDPG algorithm and the accompanying neural networks, this project strived for clarity rather than performance. In that sense, the neural networks structured in this project are not fully optimized. Although the use of GPUs to accelerate the weight propagation through the networks can be achieved by substituting the *tensorflow* library by the *tensorflow-gpu* library (along with the proper CUDA driver installation), the current structure of the implementation means that little speedup is observed when relying on GPUs. That is because the iterative loop associated with reinforcement learning negates the gains of GPUs by producing a bottleneck in the data which is fed to the GPUs. Even though this problem is not trivially solved, and doing so was beyond the scope of this study, if the networks were implemented in a more efficient way, this would tighten the feedback loop between declaring an experiment and observe the results. In that sense, by improving the efficiency of the implementation, one would also advance in the didactic nature of the project.

Another point that, in hindsight, could be improved is the choice of parameters used for the learning algorithm. Further work could be done to test the learning process with different settings of batch size and trace length to gauge the impact on learning. Additionally, the target networks' mixing factor (τ) could be defined in a way similar to what was done for the exploration parameter (ε). That is, instead of declaring an absolute value for the experiment, this value could be calculated as a function of other parameters that control the total experiment duration (e.g.: number of episodes and steps per episode).

Finally, the analysis could have tested the models with even more scenarios to gauge the generalizability power of the trained models. For example testing with the models with larger noise rates and in different combined power levels.

6 Evaluation, Reflections and Conclusions

6.1 Evaluation

This project proposes the inclusion of multi-objective RL techniques to solve tertiary control in a multi-agent based model which performed the task of fully distributed primary and secondary control. The end-goal in this scenario would be to achieve tertiary control in a fully decentralized way. Even though the proposed implementation is not yet fully decentralized, a consequence of the use of total secondary control as input for optimizing cost, it should be regarded as an important step in the right direction since the method is able to decentralize the decision making. Future work in this line of study could improve upon the results by further relaxing the real time constraints associated with the need for centralized information.

On the topic of the project's initial planning, I would evaluate it as successful. The general timeframe reached by waterfall-like methodologies was mostly followed over the duration of the project. Meanwhile, the agile methodologies guided the decision making for which problems to tackle first to decrease uncertainty and increase the chance of success of the project. An example of such decision making would be the decision to forego work in the report for some periods of time in order to dedicate time fully to the code implementation. This was done because at certain points in time, the uncertainty associated with some aspects of the code were considered to be much higher than working on the report. In light of that, it was opted to deprioritize a subtask of low uncertainty (e.g. write section 2 of the report) and prioritize a task with high uncertainty and high impact (e.g. have complete understanding of the inner workings of the MADDPG algorithm and have a working implementation).

Regarding the literature used in this study, I would consider it to be thorough. More specifically in the context of multi-objective RL techniques, as described in section 2, there is a vast number of strategies that can be employed to that end and this study opted for electing two and comparing them in terms of performance and implementability. The literature on that topic was what informed this decision and shaped the guidelines employed for reward composition.

This project set out to achieve two objectives, i) to provide an analysis regarding the feasibility of using multi-objective reinforcement learning techniques for enacting frequency control while simultaneously optimizing for cost, and ii) produce a codebase which could be reutilized for further experimentation both in the particular problem studied as well as reinforcement learning in general.

6.1.1 Multi-Objective Frequency Control

One of the objectives this study set out to achieve was to answer the following question:

"Can we leverage Multi Objective, Multi-Agent Reinforcement (MOMARL) techniques to control the power output of multiple power generators to efficiently balance a dynamic load while accounting for the secondary objective of minimizing the cost of energy production?"

With respect to primary, secondary and tertiary control (Multi Objective), the performance achieved in this study, shown in experiments II and III can be considered an important progress towards fully functional systems. Upon further scrutiny, the degree to which said models accomplish each of the two objectives would still fall short of currently required standards. However, it should be mentioned that as discussed in section 4, the author observed significant improvements in the performance by both further tuning the constants in the reward functions and prolonging the length of training.

In relation to decentralization, the methods employed in this study to perform tertiary control are able to decentralize the decision making, albeit still relying on some level of information centralization. This represents a significant change in the centralized entity from command-control to information aggregator. Nevertheless, it still cannot be considered full blown decentralization.

Overall, in the matter of structure, it can be said that it is possible to leverage MOMARL techniques to that end. However, limitations of scale still persist. These should be addressed on future work further inform the viability of employing such techniques in a way that conforms to the requirements of the industry. Section 6.2, below, does an exercise in devising possible future works.

6.1.2 Code Quality and Further Experimentation

As far as objective ii is concerned, it can be considered as been fully satisfied. The expressive power of the codebase developed enables the declaration and execution of a vast number of experiments far beyond the subset visited in this study. The learning params can be easily changed with a simple literal declaration. These include, but not limited to, number and types of generators and loads; exploration/exploitation constants; experiment duration; and learning update constants. Moreover, the code is structured in a way to provide low coupling between the simulation; learning algorithm; actor and critic entities; and abstraction from environment state to agent inputs. This allows for extensions in multiple directions, some of which are mentioned in section 6.2. The ability to declare new models with different inputs for state and reward functions can be highlighted one such means of extension and a display of the advantages of the structure chosen for the codebase.

As discussed above, amalgamating multiple objectives into a single reward function is no easy task. I am pleased with the guidelines defined in this study to perform this task, which were a product of my growing understanding of the problem at hand. However, even upon relying on these guidelines, the search for a parameter setup and ratio between the reward components which performs as desired was quite time consuming, even more so than previously expected. Initially framing one of the models of the reward composition approach meant that the same difficulties were observed to some extent on both approaches. This, in turn, consumed more time than expected as running the simulations take considerable time. In retrospect, if I were to perform the same study from the start, I would dedicate even more time to studying more guidelines and would perhaps choose a set of objectives that is easier to decompose into single objective models. Granted, I understand that the difficulties encountered in finding the best configurations for the reward compositions reinforce the reasons by which I believe the action composition approach to be superior, whenever it can be employed.

6.2 Future Work

This work relies on a number of assumptions and simplifications to restrict the scope of study and make it viable to perform the analysis within the timeframe available. In light of this, this work should not be regarded as an attempt at creating a fully operational system, but rather as a proof of concept used to inform future work and a platform which facilitates such further analyses.

As such, this subsection is dedicated at identifying some of these simplifications, their unfolding consequences and an exercise at envisioning possible solutions for remedying said consequences.

6.2.1 Decentralized Control

It is important to consider the decentralization of control tackled in this analysis. Traditional control solutions for economic dispatch are fully centralized, with a single entity centralizing information and decision making. In essence this entity acts as a sole omniscient being which issues commands to all generating nodes in the system. The system designed in this analysis albeit decentralized from the decision making standpoint, still relies on centralized information regarding the current state of the system. That is, during execution, the generators do not share between themselves information regarding their policy (i.e. which actions they will take next), they do, however, share information regarding their current state. This information comes in the form of the total secondary action of the system, which is used as input for the actors in the cost-optimizing models. Although not completely fulfilling the decentralization requirement, this marks an important step towards full decentralization, as it changes the nature of the centralized entity from a fully fledged decision maker to an information broker. Future work towards increased decentralization could involve the use of accessory metadata such as timestamps associated with the total secondary action. Intuitively, this could help relax the real time constraint of the information centralization by enabling agents to rely on stale information for approximating the desired behaviour.

6.2.2 Environment Complexity

Another topic which should be addressed in future work is the increase in complexity associated with increasing the number of agents interacting with the system. Systems with more agents naturally take longer to train by simple consequence of the increase in the state action-space associated with having more agents. Additionally, training is also elongated by the increase in the number of networks being trained. Together, these realities produce a twofold effect of both increasing the processing time spent for running each episode, and requiring more episodes to be ran in order to achieve a working model. If we are to consider the ultimate goal of having a large number of agents interacting in the electrical grid, the system as designed in this model is not up to the task as the computing resources required for that would become prohibitively expensive.

Furthermore, the offline training methodology applied in this study would be poorly suited for deploying in a real world scenario. In an offline trained system, introducing new agents in the system would require for the entire system to be retrained, which is undesirable. Large distributed systems should enable the seamless introduction of new agents by automatically adapting to the changes caused by said introduction.

For tackling the problems above mentioned, an interesting strategy could be to rely on agent templates or presets. In that sense, instead of each agent being individually tailored, one could have a finite number of presets trained and have a number of instances of each preset present and interacting in the power grid. Global information regarding the number of instances per template present in the system would still be required to be centralized, although hopefully this could be dealt with by relaxing the real time constraint as hypothesised in section 5.2.

6.2.3 Electrical Simulation

One of the most evident simplifications made over the course of the study lies in the simulation of the electrical system. On this matter, one can highlight two of the most impactful simplifications: i) the entirety of the power consuming nodes are reduced into a single load, moreover, ii) this node is kept stable over the course of each episode.

On this topic, future work could focus on improving the details of the electrical system simulation. A more granular and faithful simulation can be accomplished by leveraging open source solutions such as SciGrid(Power.scigrid.de, 2017) or PyPSA(Brown, Hörsch and Schlachtberger, 2018). Moreover, it is known that real life power systems do not observe a constant load, but rather the load is continuously changing. In this regard, further experiments could be run in which the load changes over time.

6.2.4 More Objectives

In the settings observed in this project, there were only two objectives to optimize the system for. However, in the context of real world applications, other objectives there are bound to arise. One example would be diminishing the ecological impact of powering the grid, giving priority to less pollutant generators and reserving the use of those higher pollutant to times of greater need. Both multi-objective frameworks used allow for expansion into even objectives. For the reward composition approach, introducing a new objective would amount to multiplying the global reward function by a new component, and adjust the constants according to the order of precedence of all objectives. The action composition approach, in turn, requires a new single objective model to be trained, and the action composition function to be updated to include the actions issued by this new model, along with updating the weights assigned to each models' actions.

6.2.5 Cooperation by Omission

As was seen in section 4.1, the models trained to perform frequency control converged into an interesting strategy that had two out of the three models stabilize their outputs at the lowest possible value, while a sole generator effectively performed frequency control. As discussed in said section, there are a number of measures that can be taken to disincentivize this behaviour. I would single out performing longer training sessions which cover a wider range of the total output spectrum.

6.2.6 Downward Bias in Reward Composition

Section 4.2 showed the presence of a consistent bias toward lowering the observed frequency when both frequency balance and cost optimization objectives are combined in a single reward function. Despite adjustments being made to neutralize this effect, it persisted manifesting itself. In light of that, further work could be carried out to uncover the root nature of this bias and possible courses of action that can be taken to neutralize it.

6.3 Reflection

This project posed a unique opportunity to perform an in depth study in a topic of my choice. It is true that multiple classes in the MSc course had projects which left the topic to be studied quite open, however there is a significant difference in terms of time allocated to perform the work. During regular terms the high workload had an effect of shallowing the studies performed by sheer fact of the competing priorities of multiple concurring classes. In the case of the individual thesis, there was ample time to dedicate to the study, as such the difference in depth and quality of the work performed is palpable.

As far as the chosen topic for study, even though it was based on a suggestion made by my supervisor, Dr. Eduardo Alonso, to build on top of a previous study, I feel it was a good match for my practical skill set as well as my intentions. Despite my previous work experience being in the software engineering realm, my original academic formation is that of electrical engineering with emphasis in computing. As such the depth in which the study was performed was within my reach, although the electrical engineering facet of the work was beyond my expertise. Having said that, it is clear that my interests and future ambitions lie more on the reinforcement learning realm rather than electrical engineering. A fact which is reflected in this study, as the reinforcement learning aspect was more robustly developed than the electrical engineering.

Overall I am more than pleased with the final result of this study, granted that there are improvements to be made, as mentioned in section 6.2. With respect to reinforcement learning, the implementation of a relatively new algorithm (MADDPG) was definitely one of the most interesting aspects of the project. Even though I had, to some degree, previously studied the matter of multiple agents interacting in an environment for the Software Agents class project, the difference in complexity of the methods employed is quite large. Said project relied on simple tabular TD learning methods, while this one included deep neural networks, actor critic, and information sharing between critics.

Being a software engineer by trade, I was quite interested in how to perform data science experimentations while building a robust code base. This had a deep impact in shaping the project as a whole and was one of the main reasons why objective ii was chosen. Even though data science projects in general rely on code to perform experimentations, it is often the case that the code produced can be classified as “spaghetti code” which makes for difficult reuse and understanding of the internal abstractions. In that sense, as much as an experiment on how to use RL techniques to perform frequency control, this project as a whole was an experiment on how to leverage software engineering techniques to perform an RL experiment. One interesting byproduct of this objective is that in order to structure the project code in the desired way, it was necessary to revisit the original implementations instead of relying on the original code of the MADDPG algorithm and electrical system simulations as standalone libraries. In doing so, I increased my understanding of the algorithm employed. So the choice of dedicating time to structure the code had a positive byproduct of deepening my comprehension of the techniques used.

6.4 Conclusions

This study proposes two strategies for performing semi-decentralized tertiary control, each with its own benefits and disadvantages. Overall both were able to perform the task in general terms, although further work would be required for tuning the solutions, demonstrating its generalizable capabilities and applying it to industrial scenarios. Simultaneously, the codebase implemented facilitates further experimentation in that regard and is open for future use.

On a final note, the difficulties encountered in designing proper reward functions serves as an example that reinforcement learning algorithms and techniques, as with any tool, depend on the expertise of those who wield it. Having a profound understanding of the domain in which the algorithms are being applied is tantamount to achieving positive results.

Bibliography

Ambrose, J. (2019a). Electric cars could form battery hubs to store renewable energy. [online] the Guardian. Available at: <https://www.theguardian.com/environment/2019/jul/11/electric-cars-could-form-battery-hubs-to-store-renewable-energy> [Accessed 19 Sep. 2019].

Ambrose, J. (2019b). New rules give households right to sell solar power back to energy firms. [online] the Guardian. Available at: <https://www.theguardian.com/environment/2019/jun/09/energy-firms-buy-electricity-from-household-rooftop-solar-panels> [Accessed 19 Sep. 2019].

Andrade, J., Baldick, R. (2017) How Do We Estimate Transmission Costs for New Generation?, IEEE Spectrum, Accessed: 23 March 2019, <https://spectrum.ieee.org/energywise/energy/policy/how-do-we-estimate-transmission-costs-for-new-generation>.

Apostolopoulou, D., Sauer, P. W. and Dominguez-Garcia, A. D. (2015a) Balancing authority area coordination with limited exchange of information, IEEE Power and Energy Society General Meeting, 2015–September. doi: 10.1109/PESGM.2015.7286133.

Apostolopoulou, D., Sauer, P. W. and Dominguez-Garcia, A. D. (2015b) Distributed optimal load frequency control and balancing authority area coordination, 2015 North American Power Symposium, NAPS 2015. doi: 10.1109/NAPS.2015.7335113.

Brown, T., Hörsch, J. and Schlachtberger, D. (2018). PyPSA: Python for Power System Analysis. Journal of Open Research Software, 6.

Chen, X. et al. (2018) Meta-Learning for Multi-objective Reinforcement Learning. <https://arxiv.org/pdf/1811.03376.pdf>

Congress, U. S. (2005). Energy Policy Act of 2005, Subtitle C, Section 322. Industry Applications, 1–551. <https://www.epa.gov/laws-regulations/summary-energy-policy-act>

Emami, P. (2016). Deep Deterministic Policy Gradients in TensorFlow. [online] Github. Available at: <https://pemami4911.github.io/blog/2016/08/21/ddpg-rl.html>[Accessed 19 Sep. 2019].

Kapoor, S. (2018) Multi-Agent Reinforcement Learning: A Report on Challenges and Approaches. Available at: <https://arxiv.org/pdf/1807.09427.pdf> (Accessed: 10 April 2019).

Kirby, B. J. (2003). Frequency Control Concerns in the North American Electric Power System. Available at: https://www.osti.gov/servlets/purl/885842.

Leggett, T. (2017). How your electric car could be 'a virtual power station'. [online] BBC News. Available at: <https://www.bbc.co.uk/news/business-42013625> [Accessed 19 Sep. 2019].

Liang, E., Liaw, R. (2018) Scaling Multi-Agent Reinforcement Learning, The Berkeley Artificial Intelligence Research Blog, Accessed: 10 April 2019, <ttps://bair.berkeley.edu/blog/2018/12/12/rllib>.

Liu, C., Xu, X. and Hu, D. (2015) Multiobjective reinforcement learning: A comprehensive overview, IEEE Transactions on Systems, Man, and Cybernetics: Systems. IEEE, 45(3), pp. 385–398. doi: 10.1109/TSMC.2014.2358639.

Lowe, R. et al. (2018) Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. Available at: <https://arxiv.org/pdf/1706.02275.pdf> (Accessed: 2 August 2019).

Lowe, R., et al. (2017). Learning to Cooperate, Compete, and Communicate. [online] OpenAI. Available at: <https://openai.com/blog/learning-to-cooperate-compete-and-communicate/> [Accessed 19 Sep. 2019].

Miller, R. H., Malinowski, J. H., (1994) Power system operation, McGraw-Hill Professional, ISBN 0-07-041977-9

Moffaert, K. Van and Nowé, A. (2014) Multi-Objective Reinforcement Learning using Sets of Pareto Dominating Policies, Journal of Machine Learning Research. <http://www.jmlr.org/papers/volume15/vanmoffaert14a/vanmoffaert14a.pdf>.

Morgan, M. G., Barkovich, B. R. and Meier, A. K. (1973) The social costs of producing electric power from coal: A first-order calculation, Proceedings of the IEEE, 61(10), pp. 1431–1442. doi: 10.1109/PROC.1973.9295.

Nguyen, T. (2018). A multi-objective deep reinforcement learning framework. arXiv preprint arXiv:1803.02965. <https://arxiv.org/abs/1803.02965>

Pipattanasomporn, M., Feroze, H. and Rahman, S. (2009) 'Multi-agent systems in a distributed smart grid: Design and implementation', 2009 IEEE/PES Power Systems Conference and Exposition, PSCE 2009. IEEE, pp. 1–8. doi: 10.1109/PSCE.2009.4840087.

Power.scigrid.de. (2017). SciGRID General information. [online] Available at: <https://www.power.scigrid.de/> [Accessed 20 Sep. 2019].

Rhodes, J. (2017) How Does Geography Figure Into the Full Cost of Electricity?, IEEE Spectrum, Accessed: 23 March 2019, <https://spectrum.ieee.org/energywise/energy/policy/how-does-geography-figure-into-the-full-cost-of-electricity>.

Rozada, S. (2018) Frequency Control in Unbalanced Distribution Systems, City, University of London MSc Data Science Thesis

Sensfuß, F., Ragwitz, M., Genoese, M. (2007). The Merit-order effect: A detailed analysis of the price effect of renewable electricity generation on spot market prices in Germany. Working Paper Sustainability and Innovation No. S 7/2007 (PDF). Karlsruhe: Fraunhofer Institute for Systems and Innovation Research (Fraunhofer ISI).

Steitz, C. (2019). Nissan Leaf gets approval for vehicle-to-grid use in Germany. [online] reuters. Available at: <https://www.reuters.com/article/us-autos-electricity-germany/nissan-leaf-approved-for-vehicle-to-grid-use-in-germany-idUSKCN1MX1AH> [Accessed 19 Sep. 2019].

Sutton, R. and Barto, A. (2018). Reinforcement learning. 2nd ed. Cambridge: MIT Press.

Tielens, P. and Van Hertem, D. (2012) Grid Inertia and Frequency Control in Power Systems with High Penetration of Renewables.

Tielens, P. and Van Hertem, D. (2015) ‘The relevance of inertia in power systems’. doi: 10.1016/j.rser.2015.11.016.

’T Hoen, P. J. et al. (2006) ‘An overview of cooperative and competitive multiagent learning’, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 3898 LNAI(January 2005), pp. 1–46.

U.S. Energy Information Administration. (2019). Levelized Cost and Levelized Avoided Cost of New Generation Resources in the Annual Energy Outlook 2019. Independent Statistics & Analysis, Retrieved from <https://www.eia.gov/outlooks/aeo/pdf/electricity_generation.pdf>

Wirfs-Brock, J., Paterson, L. (2015) IE Questions: What Is Inertia? And What’s Its Role In Grid Reliability?, Inside Energy, Accessed: 10 April 2019, <http://insideenergy.org/2015/06/15/ie-questions-what-is-inertia-and-whats-its-role-in-reliability>

Wood, A. and Wollenberg, B. (1996). Power Generation, Operation, and Control. New York: John Wiley & Sons., pp. 29–72

Yoon, C. (2019). Deep Deterministic Policy Gradients Explained. [online] Medium. Available at: <https://towardsdatascience.com/deep-deterministic-policy-gradients-explained-2d94655a9b7b> [Accessed 19 Sep. 2019].

Appendix A - Original Project Proposal

Cost Optimization In Frequency Control Of Unbalanced Distribution System

Flavio R. de A. F. Mello

1. Introduction

This study aims to enact decentralized control of the power output of multiple power sources to balance a dynamic load in a synchronous grid while minimizing both error and financial cost incurred in the power generation process. To achieve such objective, multi objective objective and multi agent reinforcement learning techniques will be employed.

Over the last decades, increase in adoption of residential power generators such as solar panels, along with shifts towards increased use of methods based on renewable resources represent initial signs of a large shift in the power generation and distribution paradigm. Overall these changes lead to a significant increase in the number of electrical power generating nodes in the shared network. Such increase poses a difficult challenge to currently employed power systems control techniques which often rely on some portion of centralized control. Naturally, an exponential increase in nodes to be controlled fatally leads to a proportional increase in the complexity and processing power required to enact such control. To meet such challenges, decentralized control techniques are needed.

Furthermore, this project is based on the work of Rozada, S. (Rozada, 2018), whose study of the feasibility of enacting such control with Multi Agent Reinforcement Learning (MARL) techniques shown promising results. Thus, this study sets out to built on top of previous results by introducing Multi Objective Reinforcement Learning (MORL) techniques into the model devised by Rozada. Ultimately, this study strives to answer the following question:

"Can we leverage Multi Objective, Multi Agent Reinforcement (MOMARL) techniques to control the power output of multiple power generators to efficiently balance a dynamic load while accounting for the secondary objective of minimizing the cost of energy production?"

2. Critical Context

In order to appropriately devise sound simplification models and correctly assess the ensuing results, one must first appreciate the original problem being dealt with. In the task at hand, the original problem is one of electrical engineering nature. Contemporary electrical power distribution is done by way of a synchronous grid connecting multiple consumers (e.g. households, industry, public infrastructure, etc) and power generation plants - which may rely on a multitude of different technologies: nuclear, fossil fuels and hydro based solutions being the most widely known representatives. The standard used throughout the world in terms of infrastructure is to supply energy to the system using Alternating Current (AC), operating at frequencies of either 50 or 60Hz. Working in tandem, all power plants connected to the same grid operate to keep the system balanced (i.e. one in which the amount of power injected into the system is equal to amount of power being consumed). This can be achieved by focusing at the operational frequency of the system at any given point. If the load is greater than the generation, there is a decrease in frequency. Conversely, a system with a load which is smaller than the generation would observe an increase in the frequency. The control task is then to continuously adjust the power output of all the suppliers in a system to counterbalance changes in the overall load.

2.1 Power Generation Control

Primary Control: Primary control aims to accomodate for changes in system load by proportionately adjusting the output of all generators within the synchronous grid. This is traditionally implemented using Droop Control(Miller and Malinowski, 1994) which can be deployed in distributed manner. However, this technique has its limitations. Namely, it is known for introducing steady state errors and does not take into account any economic considerations as input.

Secondary Control: Secondary control also aims to balance the power in the grid by minimizing the frequency deviation from the nominal value. It does so by dealing with the steady state error introduced by primary control. This layer relies on Automatic Generation Control (AGC) algorithms(Miller and Malinowski, 1994), which often depend on some sort of centralization, relying on a single node having holistic knowledge of the complete network and issuing commands to each power generator to adjust its output accordingly.

Tertiary Control: Also named as Economic Dispatch, the tertiary control aims to control the power generation to balance the system-wide load while minimizing the total production cost and accounting for operational limits of generation and transmission facilities (Congress, U. S., 2005). This is also done in centralized fashion.

The rise of microgrids and increase in the distribution of power supply into the grid it entails poses a difficult challenge for any centralized control technique. The action control space grows exponentially with respect to the number of nodes supplying power to the system. Additionally, physical distance and increase in data processing requirements will invariantly lead to delays in the selection, issuing and execution of control strategies. For surmount these difficulties, decentralized control techniques are required to more efficiently and reliably deal with electrical system load balancing.

There is existing work in the bibliography regarding such decentralized techniques, from a traditional control standpoint, Apostolopoulou, Sauer and Dominguez-Garcia propose methods for approximating the ACG algorithm while solving the economic dispatch in semi-decentralized fashion by restricting the Balancing Authority (BA) areas' communication and, thus, avoiding congestion associated with the exponential increase of connections in the network and having a single node process the economic dispatch for the whole network(Apostolopoulou et al., 2015a) (Apostolopoulou et al., 2015b). From a software agents standpoint, Rozada proposes the use of multi agent reinforcement learning algorithms, modeling each power plant control system as an individual agent and running collaborative MARL algorithms to have the agents converge into optimal policies for dealing with changes in system-wide load (Rozada, 2018).

Although the initial results are promising for primary and secondary control, Rozada's model falls short when enacting tertiary control in the system. This is because the technique relies on a single objective model, that is a model which aims to aims to maximize a single quantity, (or, equivalently, to minimize a single error function). The reason why it was successful for both primary and secondary controls is that both layers have the same ultimate objective — to minimize the error between the system frequency and a nominal set frequency. In this sense, primary and secondary refer to layers in the engineering solutions employed in combination to reach a same goal: to enact such control and minimize frequency deviation. Because of this shared objective, Rozada was able to reach a successful model for dealing with Primary and Secondary controls using a Multi Agent Actor Critic Reinforcement Learning model by condensing both control layers into a single objective(Rozada, 2018).

This study aims to add a new layer in the solution, by incorporating multi objective learning techniques to deal with the tertiary control problem as a separate, subordinate objective.

2.2 Cost Estimation

Estimating the cost of generating energy is in itself a rather complicated problem. Traditionally, the cost is measured in monetary value per energy unit produced (e.g. $/MWh) and broken down into 3 larger factors:

Capital Costs: One time costs associated with the project creation/construction in general.

Fuel Costs: Recurrent costs associated with obtaining/transporting the fuel necessary to run the power generation plant.

Additional Costs: Typically recurring costs not associated with fuel, examples such as insurance, parasitic load, etc.

These factors vary different between different ways of producing energy (e.g. Wind, Solar have considerably high capital cost, but no fuel associated costs, Nuclear has an extremely high capital cost, as costs associated with decommissioning and nuclear waste management are considered capital costs, conversely, fossil fuel-based power stations have comparatively low capital costs, but high fuel costs).

Considering the many moving parts associated with running a power plant, and the sheer longevity of the enterprises — many plants being designed to operate for multiple decades — makes a precisa, fine grained calculation almost impossible. In order to enable some comparison between the cost effectiveness of different kinds of power stations, the Levelized Cost of Energy (LCOE) is often used(U.S. Energy Information Administration, 2019). LCOE is calculated based on estimated capital and fuel costs along with projected average operational load over the complete life of the enterprise. Being such a coarse estimation of the cost estimated over the complete lifetime, LCOE is expressed as being constant per energy unit produced (e.g. $/MWh) and does not provide insight into differences in producing cost based on the load relative to the capacity.

2.3 Inertia

If a system is modeled to only account for LCOE, the problem of optimizing for cost becomes trivial, and a simple Merit Order approach can be taken to activate plants in order of increasing LCOE (Sensfuß et al., 2007). Evidently there are multiple other factors at play when controlling a fully operational power grid. For the purpose of this study, an additional component will be taken into account in the model — inertia. In electrical grids, inertia can be described as its resistance to change operational frequency due to changes in the generator/load balance, and it is associated with the Newtonian concept of inertia(Tielens and Van Hertem, 2012). In the most commonly employed electricity generation models, electricity is generated using a rotating dynamo, changes in the system load will lead to changes in the rotating speed (frequency) of such dynamos, the extent of such changes is then proportional to the inertia of these rotating masses. Furthermore, different power plants are able to respond with different speeds to demands in change of power output.

The introduction of inertia in the model has the effect of better approximating the physical characteristics of the problem, while also de-trivializing the economic dispatch problem — while a cheaper option might be available, its higher inertia would prevent it from being selected to counterbalance fast changes in the system load.

In conclusion, in order to effectively study the chaotic problem of controlling power generation for balancing load in a complex grid, multiple simplifications and assumptions are made, the degree of which depend on the depth and granularity needed for the study in question. The extent of the simplifications used in this study are further discussed in section 3.

2.4 Multi Objective and Multi Agent Reinforcement Learning

Reinforcement Learning (RL) has its origins as a field of study stemming from research regarding the learning process of animals. Originally, techniques were developed for a single agent interacting with a state, while striving to achieve a single objective. Q-Learning and SARSA are prime examples of such algorithms. Over time, further work in the field developed new algorithms and methods to account for multiple independent agents interacting in a same state space while acting in either collaborative, competitive or neutral fashion. These are considered Multi Agent Reinforcement Learning (MARL) scenarios (Kapoor, 2018). The field also evolved to include Multi Objective Reinforcement Learning (MORL) scenarios in which more than one objective is desired — the agent(s) aim(s) to solve two or more tasks simultaneously (Liu, Xu and Hu, 2015). The proposed task for this study is classified as both Multi Agent — multiple power stations (agents) operate independently, but within the same grid (environment) while aiming to achieve the same objective (cooperative) — and Multi Objective — agents are required to fulfill two objectives: minimising frequency deviation from nominal value, and minimising cost of production. For this purpose, both multi agent and multi objective methodologies will be united in a single model.

3. Approaches

3.1 Model Implementation

This study intends to follow the same general model established by Rozada (Rozada, 2018), with some added changes to better study the economic dispatch problem. Ultimately, the complex power grid control problem will be distilled into a model with the following components and assumptions:

- Two distinct power generators with different values for LCOE and inertia. Multiple configurations of LCOE and inertia will be tested.
- A single, dynamic load representing the total load in the grid. Changes in the total load will happen randomly throughout the experiments following predetermined probability distributions.
- All nodes are fully operational for the entire duration of the simulations (i.e. no downtime at all, faults and any sort of scheduled maintenance are modeled).
- The system load is smaller than the capacity of either generator. (i.e. the generators have unbounded capacity as its role in economic dispatch is not the subject of this study).
- Lossless power transmission. The entire power provided by the generators reach the load without any sort of loss in the transmission lines, transformers, substations, etc.
- Time is discretized for ease of implementation of the selected Reinforcement Learning techniques. Continuous time is then divided into same homogeneous slices of arbitrary duration. Consequently, actions in the power output of the generators are also discretized, becoming fixed-amplitude increments/decrements which can occur in each time slice.

3.2 Software Implementation

The ecosystem selected to develop this study is the Python programming language. Python is a modern programming language, with focus on code readability and support for Object Oriented and some Functional Programming paradigms (Kuhlman, 2012). Its ease of use led to its adoption by multiple development communities, including data science and with availability of multiple open source libraries focusing in machine learning, statistics and neural network capabilities. Although Python v2 is still widely supported and used by the community, I intend to implement my solution based on Python v3, as the former is set to be deprecated by 2020. Additional tools which will be used in the study are:

PIP **-** Popular package management system for the Python language. Used to install and manage the specific versions of each package used in the solution.

Pandas / **NumPy** / **Scikit-Learn** / **Pyplot** **-** Standard python packages used by the data science community. Pandas focuses on manipulation of tabular based data. NumPy is a scientific computing focused package containing mathematical functions including linear algebra and multidimensional matrices. Scikit-Learn provides machine learning algorithm implementations and data partitioning functions. Finally, pyplot is a standard visual plotting library which will be used to present the results of the experiments.

PyBrain - Machine learning library focused on flexibility and simplicity of use. Provides implementations for classic reinforcement learning algorithms.

Keras / **TensorFlow** **-** Keras is a neural network library for python. Its ability to run on top of TensorFlow and relatively ease of use makes it a popular tool.

Git / **Github** **-** Git is an open source distributed versioning control system widely adopted in academia and industry. Github is a code-hosting service based on Git.

Jupyter - Interactive programming environment built on top of python. Frequently used to share results as it offers the capability of building a narrative intercalating code and visual representations.

This study focuses on the suitability of using Reinforcement Learning techniques to solve the tertiary control, as such the implementation will focus on high level components to enable faster prototyping, rather than relying in low level components which enable better performance via fine tuning.

The wide range of supporting material and package ecosystem available for Python makes it a prime candidate for the task at hand. Additionally, its full support for object orientation makes for an easier transition than Matlab from my past experience in the software development industry.

Architecture-wise, the project does not require overly complex structures. I propose a shallow folder structure which should suffice for the task at hand. Apart from project configuration files residing at the root folder, the files are to be separated into 6 larger groups/folders:

Electricity - Contains the modelling of the electrical grid to be studied.

Models/DTOs **-** Models and data transfer objects used to pass along structured data between methods and classes.

Learning **-** Implementation of reinforcement learning algorithms.

Plotting **-** Visual outputs from data obtained in the experiments.

Experiments - The experiments themselves, the topmost level code in the project. Leverages all the other components to prepare, run and represent the results of experiments. This is the only folder where Jupyter Notebooks should be used to leverage its narrative capabilities.

Data - Storage for past results and intermediate data to ensure reproducibility of the findings of this study.

Additionally, figure 1 represents the dependency graph in the project structure. It is important to note that the proposed architecture dependency graph is acyclical, thus avoiding circular dependency issues.

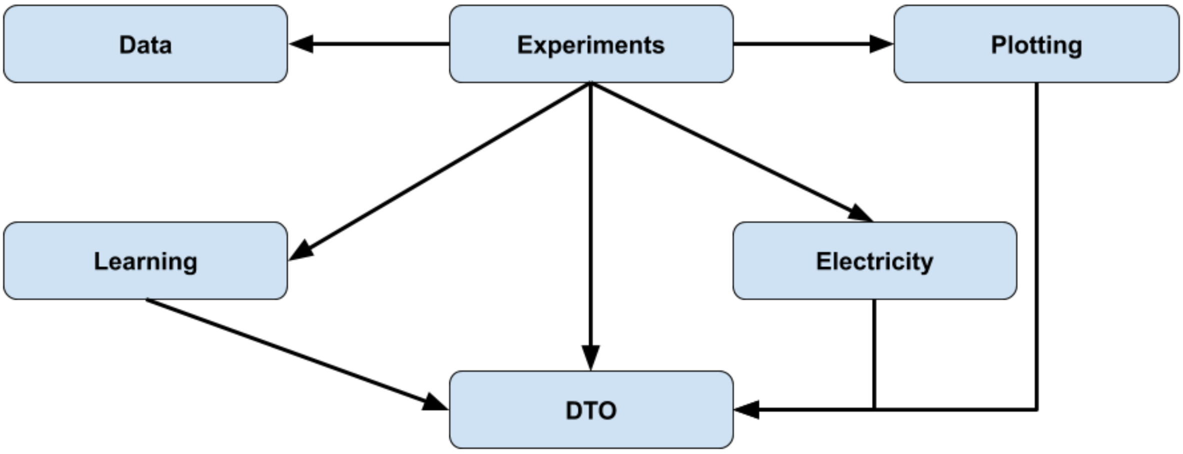


Figure 1: Dependency graph of the proposed folder structure.

3.3 Evaluation Methodology

This study aims to study the feasibility of using Multi Objective, Multi Agent Reinforcement Learning techniques to perform the control of multiple power stations to maintain a grid with a dynamic load within a maximum error of a determined nominal value by modeling a system accounting for different inertia and cost factors.

Naturally, the evaluation of the results can only be done after they are obtained. However, the evaluation methodology should be carefully considered before the implementation takes place. The evaluation of the results attained in the study will be broken down into three main sections: quantitative analysis, qualitative analysis and discussion.

Quantitative Analysis

- Compare the obtained results with currently deployed semi-centralized solutions.
- Compare the results with the ones reached by Rozada, particularly, how the introduction of an additional objective impacted the agents' ability to minimize error in nominal frequency.

Qualitative Analysis

- Compare the obtained results with currently deployed semi-centralized solutions. What are the benefits and costs of replacing currently used solutions with the proposed model?
- Compare the results with the ones reached by Rozada, particularly, to which extent the new objective hindered the agents' ability to minimize frequency error? Do the primary objective results still remain satisfactory?

Discussion

- Are the reached results satisfactory and in line with what the study proposed to achieve?
- Evaluate the tradeoffs associated with multiple objectives.
- Do the results corroborate the expectations? If negative, what are the possible causes of such deviation?

4. Work Plan

4.1 Methodology

The strict time constraints posed by this projects immutable deadline would indicate a waterfall-like approach to planning to be suitable. Conversely, although I do have prior experience studying electricity transmission in my undergraduate degree in Electrical and Computer Engineering, having also studied Reinforcement Learning techniques during the second term of the MSc programme, neither area could be considered within my expertise — therefore an agile-based approach would be best suited to deal with such uncertainty. In order to cater to both needs, I plan to use a mixture of waterfall and agile methodologies which proved to be successful in my prior experience in the software development industry. Initially, waterfall principles are used to establish a general timeline, breaking down the project into coarse blocks, to ensure the scope and planned activities fall within the determined timeframe. Additionally, agile techniques are used to gradually break down these larger work blocks into atomic tasks, schedule them in development sprints, and groom future tasks to decrease uncertainty in the backlog. Throughout the entire duration of the project, both methodologies alternate to guide the development according to macro (waterfall) and micro (agile) needs. As an additional measure to deal with uncertainty, time is explicitly allocated at the end of the project to account for unforeseen issues that may arise. Should it not be necessary, any time leftover is used to further polish the resulting project.

4.2 Supervisory Feedback

During the entire duration of the study, progress will be periodically reported to both supervisors and meetings are to be scheduled fortnightly. These meetings will be used as a valuable opportunity to gather feedback regarding the ongoing progress on both scopes: Reinforcement Learning with Prof. Eduardo Alonso, and Electrical Systems modelling with Prof. Dimitra Apostolopoulou. Any resulting feedback will be reintroduced in the study using the agile methodologies described in section 4.1

4.3 Work Plan Schedule

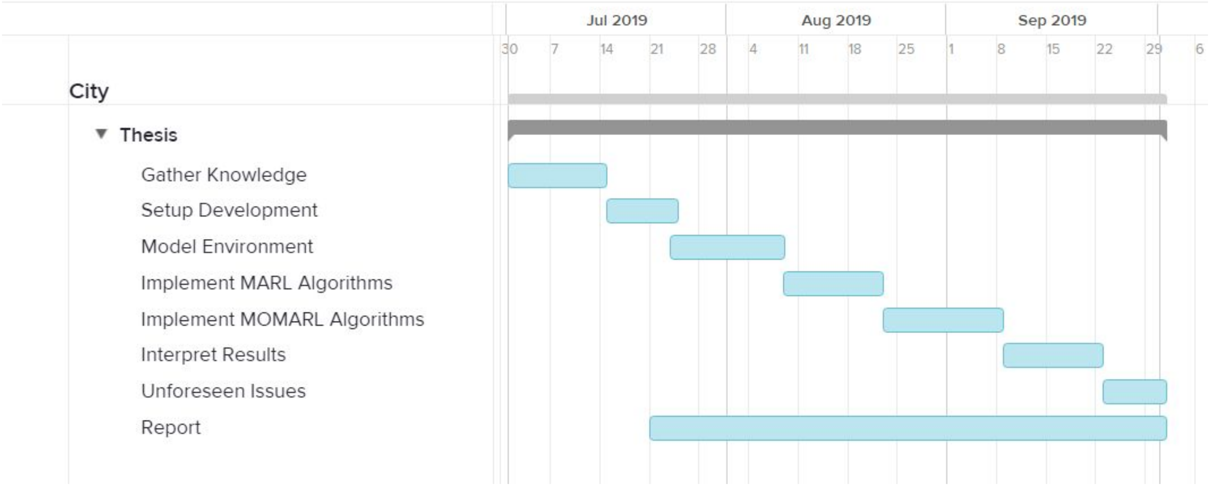


Figure 2: Work plan schedule spanning the complete duration of summer term.

4.4 Task Descriptions

Study Knowledge: Gather the necessary knowledge to proceed with a successful study. Topics covering both electrical systems (power grid modelling, inertia and cost estimation) and reinforcement learning (actor critic models, deep q learning, multi agent actor critic, and multi objective reinforcement learning techniques).

Setup Development: Setup and configuration of all the infrastructure needed for the project. Includes studying the code used by Rozada(Rozada, 2018), searching for python libraries which could ease the development, provisioning a git repository, and configuring the development environment.

Model Environment: Modelling of the electrical components of the environment. Including models for inertia and cost.
Implement MARL Algorithm: Implement multi agent RL algorithm. Particularly, the Multi Agent Actor Critic RL algorithm that Rozada proved to be successful in orchestrating primary and secondary controls(Rozada, 2018). Attempt to replicate Rozada's results as a starting point for tackling tertiary control.
Implement MOMARL Algorithms: Implement one or more instances of Multi Objective, Multi Agent RL algorithms, depending on the quantity of MORL techniques found. Run experiments with multiple different configurations.
Interpret Results: Compare and evaluate results according to methodology established in section 3.3. Fine tune visualizations to include in final report.
Unforeseen Issues: Time pre-allocated to deal with unforeseen issues that may arise during the course of the study.
Report: Writing of the report. It is a long running, continuous task that spans most of the project duration. Report should be gradually written over the course of the study to ensure details are not left out and decrease the risk of not having a full report by the end of the study.

5. Risks

Risk	Likelihood (1-3)	Consequence (1-5)	Impact L x C (1-15)	Mitigation Strategy
Loss of project code due to hard drive failure	1	5	5	Use GIT versioning control system(VCS) along with service provider (Github) to store a copy of the code in the cloud
Loss of the report due to hard drive failure	1	5	5	Keep report along with the project code safely stored in the cloud via the versioning control system
Repeated changes in the development environment results in a non replicable software environment	2	3	6	Leverage environment provisioning tools such as Vagrant, Pyenv, Pipenv and Docker
Results reached unable to be reproduced	1	4	4	Keep records of randomly generated experimental data along with rest of the project and submitted into VCS
No Multi Objective Multi Agent RL model found in knowledge gathering	2	5	10	Also search for Multi Objective and Multi Agent techniques independently and merge the concepts found to form a Multi Objective/Multi Agent model
Implementation takes too long, leaving not enough time to write a fully	2	5	10	Incrementally write report throughout the duration of the study. Keep track of current progress relative to planned

polished report before the deadline				with Waterfall methodology, replan the scope of implementation accordingly using Agile methods
Resulting body of work deviates from expected or is not suited for submission	1	5	5	Leverage meetings and continuous communication with supervisors, to receive feedback on ongoing project effort
No suitable models for inertia and estimating the cost of energy production found	1	3	3	Use simplified linear models for both parameters and run multiple experiments with different configurations of parameters for both generators
Underestimate the body of work required to fulfill proposed study	2	4	8	Plan a general roadmap with milestones using Waterfall techniques. Use agile methodologies to continuously track progress and replan scope accordingly. Allocate extra time to accomodate for unforeseen difficulties

6. References

Andrade, J., Baldick, R. (2017) How Do We Estimate Transmission Costs for New Generation?, IEEE Spectrum, Accessed: 23 March 2019, <https://spectrum.ieee.org/energywise/energy/policy/how-do-we-estimate-transmission-costs-for-new-generation>.

Apostolopoulou, D., Sauer, P. W. and Dominguez-Garcia, A. D. (2015a) Balancing authority area coordination with limited exchange of information, IEEE Power and Energy Society General Meeting,
2015–September. doi: 10.1109/PESGM.2015.7286133.

Apostolopoulou, D., Sauer, P. W. and Dominguez-Garcia, A. D. (2015b) Distributed optimal load frequency control and balancing authority area coordination, 2015 North American Power Symposium, NAPS 2015. doi: 10.1109/NAPS.2015.7335113.

Chen, X. et al. (2018) Meta-Learning for Multi-objective Reinforcement Learning. <https://arxiv.org/pdf/1811.03376.pdf>

Congress, U. S. (2005). Energy Policy Act of 2005, Subtitle C, Section 322. Industry Applications, 1–551. <https://www.epa.gov/laws-regulations/summary-energy-policy-act>

Kapoor, S. (2018) Multi-Agent Reinforcement Learning: A Report on Challenges and Approaches, <https://arxiv.org/pdf/1807.09427.pdf>.

Kuhlman, D. (2012) A Python Book: Beginning Python, Advanced Python, and Python Exercises.

Liang, E., Liaw, R. (2018) Scaling Multi-Agent Reinforcement Learning, The Berkeley Artificial Intelligence Research Blog, Accessed: 10 April 2019, <ttps://bair.berkeley.edu/blog/2018/12/12/rllib>.

Liu, C., Xu, X. and Hu, D. (2015) Multiobjective reinforcement learning: A comprehensive overview, IEEE Transactions on Systems, Man, and Cybernetics: Systems. IEEE, 45(3), pp. 385–398. doi: 10.1109/TSMC.2014.2358639.

Miller, R. H., Malinowski, J. H., (1994) Power system operation, McGraw-Hill Professional, ISBN 0-07-041977-9

Moffaert, K. Van and Nowé, A. (2014) Multi-Objective Reinforcement Learning using Sets of Pareto Dominating Policies, Journal of Machine Learning Research. <http://www.jmlr.org/papers/volume15/vanmoffaert14a/vanmoffaert14a.pdf>.

Morgan, M. G., Barkovich, B. R. and Meier, A. K. (1973) The social costs of producing electric power from coal: A first-order calculation, Proceedings of the IEEE, 61(10), pp. 1431–1442. doi: 10.1109/PROC.1973.9295.

Nguyen, T. (2018). A multi-objective deep reinforcement learning framework. arXiv preprint arXiv:1803.02965. https://arxiv.org/abs/1803.02965

Rhodes, J. (2017) How Does Geography Figure Into the Full Cost of Electricity?, IEEE Spectrum, Accessed: 23 March 2019, <https://spectrum.ieee.org/energywise/energy/policy/how-does-geography-figure-into-the-full-cost-of-electricity>.

Rozada, S. (2018) Frequency Control in Unbalanced Distribution Systems, City, University of London MSc Data Science Thesis

Sensfuß, F., Ragwitz, M., Genoese, M. (2007). The Merit-order effect: A detailed analysis of the price effect of renewable electricity generation on spot market prices in Germany. Working Paper Sustainability and Innovation No. S 7/2007 (PDF). Karlsruhe: Fraunhofer Institute for Systems and Innovation Research (Fraunhofer ISI).

Tielens, P. and Van Hertem, D. (2012) Grid Inertia and Frequency Control in Power Systems with High Penetration of Renewables.

U.S. Energy Information Administration. (2019). Levelized Cost and Levelized Avoided Cost of New Generation Resources in the Annual Energy Outlook 2019. Independent Statistics & Analysis, Retrieved from https://www.eia.gov/outlooks/aeo/pdf/electricity_generation.pdf

Wirfs-Brock, J., Paterson, L. (2015) IE Questions: What Is Inertia? And What's Its Role In Grid Reliability?, Inside Energy, Accessed: 10 April 2019, <http://insideenergy.org/2015/06/15/ie-questions-what-is-inertia-and-whats-its-role-in-reliability?.

Research Ethics Review Form: BSc, MSc and MA Projects

Computer Science Research Ethics Committee (CSREC)

http://www.city.ac.uk/department-computer-science/research-ethics

Undergraduate and postgraduate students undertaking their final project in the Department of Computer Science are required to consider the ethics of their project work and to ensure that it complies with research ethics guidelines. In some cases, a project will need approval from an ethics committee before it can proceed. Usually, but not always, this will be because the student is involving other people (“participants”) in the project.

In order to ensure that appropriate consideration is given to ethical issues, all students must complete this form and attach it to their project proposal document. There are two parts:

PART A: Ethics Checklist. All students must complete this part. The checklist identifies whether the project requires ethical approval and, if so, where to apply for approval.

PART B: Ethics Proportionate Review Form. Students who have answered “no” to questions 1 – 18 and “yes” to question 19 in the ethics checklist must complete this part. The project supervisor has delegated authority to provide approval in such cases that are considered to involve MINIMAL risk.
The approval may be provisional: the student may need to seek additional approval from the supervisor as the project progresses and details are established.

A.1 If you answer YES to any of the questions in this block, you must apply to an appropriate external ethics committee for approval and log this approval as an External Application through Research Ethics Online - https://ethics.city.ac.uk/		*Delete as appropriate*
1.1	Does your research require approval from the National Research Ethics Service (NRES)? *e.g. because you are recruiting current NHS patients or staff?* *If you are unsure try - https://www.hra.nhs.uk/approvals-amendments/what-approvals-do-i-need/*	**NO**
1.2	Will you recruit participants who fall under the auspices of the Mental Capacity Act? *Such research needs to be approved by an external ethics committee such as NRES or the Social Care Research Ethics Committee - http://www.scie.org.uk/research/ethics-committee/*	**NO**
1.3	Will you recruit any participants who are currently under the auspices of the Criminal Justice System, for example, but not limited to, people on remand, prisoners and those on probation? *Such research needs to be authorised by the ethics approval system of the National Offender Management Service.*	**NO**
A.2 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee, you must apply for approval from the Senate Research Ethics Committee (SREC) through Research Ethics Online - https://ethics.city.ac.uk/		*Delete as appropriate*
2.1	Does your research involve participants who are unable to give informed consent? *For example, but not limited to, people who may have a degree of learning disability or mental health problem, that means they are unable to make an informed decision on their own behalf.*	**NO**
2.2	Is there a risk that your research might lead to disclosures from participants concerning their involvement in illegal activities?	**NO**
2.3	Is there a risk that obscene and or illegal material may need to be accessed for your	**NO**

	research study (including online content and other material)?	
2.4	Does your project involve participants disclosing information about special category or sensitive subjects? *For example, but not limited to: racial or ethnic origin; political opinions; religious beliefs; trade union membership; physical or mental health; sexual life; criminal offences and proceedings*	**NO**
2.5	Does your research involve you travelling to another country outside of the UK, where the Foreign & Commonwealth Office has issued a travel warning that affects the area in which you will study? *Please check the latest guidance from the FCO - http://www.fco.gov.uk/en/*	**NO**
2.6	Does your research involve invasive or intrusive procedures? *These may include, but are not limited to, electrical stimulation, heat, cold or bruising.*	**NO**
2.7	Does your research involve animals?	**NO**
2.8	Does your research involve the administration of drugs, placebos or other substances to study participants?	**NO**
A.3 If you answer YES to any of the questions in this block, then unless you are applying to an external ethics committee or the SREC, you must apply for approval from the Computer Science Research Ethics Committee (CSREC) through Research Ethics Online - https://ethics.city.ac.uk/ **Depending on the level of risk associated with your application, it may be referred to the Senate Research Ethics Committee.**		*Delete as appropriate*
3.1	Does your research involve participants who are under the age of 18?	**NO**
3.2	Does your research involve adults who are vulnerable because of their social, psychological or medical circumstances (vulnerable adults)? *This includes adults with cognitive and / or learning disabilities, adults with physical disabilities and older people.*	**NO**
3.3	Are participants recruited because they are staff or students of City, University of London? *For example, students studying on a particular course or module.* *If yes, then approval is also required from the Head of Department or Programme Director.*	**NO**
3.4	Does your research involve intentional deception of participants?	**NO**
3.5	Does your research involve participants taking part without their informed consent?	**NO**
3.5	Is the risk posed to participants greater than that in normal working life?	**NO**
3.7	Is the risk posed to you, the researcher(s), greater than that in normal working life?	**NO**
A.4 If you answer YES to the following question and your answers to all other questions in sections A1, A2 and A3 are NO, then your project is deemed to be of MINIMAL RISK. **If this is the case, then you can apply for approval through your supervisor under PROPORTIONATE REVIEW. You do so by completing PART B of this form.** **If you have answered NO to all questions on this form, then your project does not require ethical approval. You should submit and retain this form as evidence of this.**		*Delete as appropriate*

4	Does your project involve human participants or their identifiable personal data? *For example, as interviewees, respondents to a survey or participants in testing.*	**NO**

Appendix B - Source Code

The original source code is hosted at Github in the form of a publicly accessible repository (https://github.com/melloflavio/2019-MSc_Thesis). It is available for use in future studies, granted it is is properly cited and its contributions to future work indicated.

There are no data appendices attached to the submission as the study is based on reinforcement learning which relies on synthetic data generated through the agents interaction with the simulated environment.

B.1 Instructions for Running Code

1. Starting with a system with python v3.7.3 and the corresponding pip version installed

    Note: *Pyenv* and *pipenv* are optional, but facilitate the process of ensuring the correct versions of the libraries

2. Install all libraries specified in section B.2. Ensure the correct versions are being installed.
    a. If using *pipenv* this can be done running the command `pipenv install`, which automatically installs all the libraries as specified in the pipfile
    b. If relying on *pip* only, each library can be installed individually
3. Start the jupyter notebook server
4. Run the experiments found in the form of notebooks in the `./app/experiments` folder
5. A template notebook can also be found which provide the blueprint for running future experiments.

B.2 Libraries

Table B.1 contains a list of the software libraries used in the project.

Library Name	Version	Purpose
Python	3.7.3	General purpose programming language used to write the complete software
pylint	2.3.1	Enforce code style consistency across the entire project
matplotlib	3.1.1	Library for generating plots. Used to produce all the plots based on the experimental results
numpy	1.17.0	Mathematical function extensions

pandas	0.25.0	Data manipulation
jupyter	1.0.0	Framework for performing and sharing experiments. Widely adopted in the data science community for its ease of use based on interactive programming as well as the ability to save the code execution results alongside the code.
ipykernel	5.1.1	Kernel associated with the jupyter framework
singleton-decorator	1.0.0	Declare configuration parameters as singleton by use of a decorator
pydash	4.7.5	Function programming extensions for python
scipy	1.3.0	Science, mathematics and engineering toolkit. Used to perform dynamic programming to find the optimal combination for solving economic dispatch
tensorflow	1.14.0	Neural Network framework used to declare and run the NNs
dataclasses-json	0.2.14	Export dataclasses to json file
Desmos	online	2D graphic calculator. Used to produce plots for single objective reward functions and reward function components
Geogebra	online	3D graphic calculator. Used to produce 3d plots for multi-objective reward functions
2018 MSc Data Science thesis - Sergio Rozada	https://github.com/sergiorozada12/frequency-control-rl	Used as starting point for the MADDPG algorithm and electric system simulation implementation.

Table B.1: List of libraries and supporting software used in the development of the project

B.3 Folder Structure

All the files contained in the submission, along with the folder structure of the project can be found in the list below.

```
root
├── app
│   ├── dto
│   │   ├── cost_profile.py
│   │   ├── electrical_constants.py
│   │   ├── electrical_state.py
│   │   ├── electrical_system_specs.py
│   │   ├── epsilon_specs.py
│   │   ├── __init__.py
│   │   └── system_history.py
│   ├── electricity
│   │   ├── area_dynamics.py
│   │   ├── cost_calculator.py
│   │   ├── electrical_system_factory.py
│   │   ├── electrical_system.py
│   │   ├── generator.py
│   │   ├── __init__.py
│   │   └── load.py
│   ├── experiments
│   │   ├── Experiment_I-Frequency-15k.ipynb
│   │   ├── Experiment_I-Frequency-9k.ipynb
│   │   ├── Experiment_II_a-CostFrequency.ipynb
│   │   ├── Experiment_II_b-CostFrequency.ipynb
│   │   ├── Experiment_III-8k.ipynb
│   │   ├── Experiment_III-Cost-15k.ipynb
│   │   ├── Experiment_III-Test-DualModel-15k.ipynb
│   │   ├── Experiment_III-Test-DualModel-8k.ipynb
│   │   └── Template-Experiment.ipynb
│   ├── learning
│   │   ├── actor_dto.py
│   │   ├── actor.py
│   │   ├── cost
│   │   │   ├── __init__.py
│   │   │   ├── model_adapter_cost.py
│   │   │   └── nn_extensions_cost.py
│   │   ├── cost_diff_frequency
│   │   │   ├── __init__.py
│   │   │   ├── model_adapter_cost_diff_frequency.py
│   │   │   └── nn_extensions_cost_diff_frequency.py
│   │   ├── cost_frequency
│   │   │   ├── __init__.py
│   │   │   ├── model_adapter_cost_frequency.py
│   │   │   └── nn_extensions_cost_frequency.py
│   │   ├── cost_single
│   │   │   ├── __init__.py
│   │   │   ├── model_adapter_cost_single.py
│   │   │   └── nn_extensions_cost_single.py
│   │   ├── critic_dto.py
│   │   ├── critic.py
│   │   ├── epsilon.py
│   │   ├── experience_buffer_dto.py
│   │   ├── experience_buffer.py
│   │   ├── frequency
│   │   │   ├── __init__.py
│   │   │   ├── model_adapter_frequency.py
│   │   │   └── nn_extensions_frequency.py
│   │   ├── __init__.py
│   │   ├── learning_agent.py
│   │   ├── learning_params.py
│   │   ├── learning_state.py
│   │   ├── model_adapter.py
│   │   ├── model_tester_action_composition.py
│   │   ├── model_tester.py
│   │   └── model_trainer.py
│   ├── models
│   │   ├── Experiment_I-Frequency-15k
│   │   │   ├── checkpoint
│   │   │   ├── learning_params.json
│   │   │   ├── model.data-00000-of-00001
│   │   │   ├── model.index
│   │   │   └── model.meta
│   │   ├── Experiment_I-Frequency-9k
│   │   │   ├── checkpoint
│   │   │   ├── learning_params.json
│   │   │   ├── model.data-00000-of-00001
│   │   │   ├── model.index
│   │   │   └── model.meta
│   │   ├── Experiment_II_a-CostFrequency
│   │   │   ├── checkpoint
│   │   │   ├── learning_params.json
│   │   │   ├── model.data-00000-of-00001
│   │   │   ├── model.index
│   │   │   └── model.meta
│   │   ├── Experiment_II_b-CostFrequency
│   │   │   ├── checkpoint
│   │   │   ├── learning_params.json
│   │   │   ├── model.data-00000-of-00001
│   │   │   ├── model.index
│   │   │   └── model.meta
│   │   ├── Experiment_III-8k
│   │   │   ├── checkpoint
│   │   │   ├── learning_params.json
│   │   │   ├── model.data-00000-of-00001
│   │   │   ├── model.index
│   │   │   └── model.meta
│   │   ├── Experiment_III-Cost-15k
│   │   │   ├── checkpoint
│   │   │   ├── learning_params.json
│   │   │   ├── model.data-00000-of-00001
│   │   │   ├── model.index
│   │   │   └── model.meta
│   │   ├── __init__.py
│   │   └── model_paths.py
│   └── plots
│       ├── costs_plot.py
│       ├── frequency_plot.py
│       ├── __init__.py
│       ├── observed_power_plot.py
│       ├── plot_all.py
│       ├── plot_constants.py
│       ├── plot_training_progress.py
│   └── rewards_plot.py
├── Pipfile
├── Pipfile.lock
└── README.md
```