UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
jou kennisvennoot • your knowledge partner

Residential Energy Storage Scheduling and System Sizing Cost Optimization

Sean Muller
23575786

Report submitted in partial fulfilment of the requirements of the module Project (E) 448 for the degree Baccalaureus in Engineering in the Department of Electrical and Electronic Engineering at Stellenbosch University.

Supervisor: Dr C. Y. Van Staden

November 2023

Acknowledgements

I like to express my deepest gratitude to my supervisor Dr Chantelle Van Staden for the exceptional guidance throughout the duration of this project. I would also like to thank Dr Herman Kamper for his LaTeXtemplate. A further thank you to all my friends and family who supported me throughout my journey through engineering, I wouldn't be here without you.

UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
jou kennisvennoot • your knowledge partner

Plagiaatverklaring / *Plagiarism Declaration*

1. Plagiaat is die oorneem en gebruik van die idees, materiaal en ander intellektuele eiendom van ander persone asof dit jou eie werk is.
   Plagiarism is the use of ideas, material and other intellectual property of another's work and to present is as my own.

2. Ek erken dat die pleeg van plagiaat 'n strafbare oortreding is aangesien dit 'n vorm van diefstal is.
   I agree that plagiarism is a punishable offence because it constitutes theft.

3. Ek verstaan ook dat direkte vertalings plagiaat is.
   I also understand that direct translations are plagiarism.

4. Dienooreenkomstig is alle aanhalings en bydraes vanuit enige bron (ingesluit die internet) volledig verwys (erken). Ek erken dat die woordelikse aanhaal van teks sonder aanhalingstekens (selfs al word die bron volledig erken) plagiaat is.
   Accordingly all quotations and contributions from any source whatsoever (including the internet) have been cited fully. I understand that the reproduction of text without quotation marks (even when the source is cited) is plagiarism

5. Ek verklaar dat die werk in hierdie skryfstuk vervat, behalwe waar anders aangedui, my eie oorspronklike werk is en dat ek dit nie vantevore in die geheel of gedeeltelik ingehandig het vir bepunting in hierdie module/werkstuk of 'n ander module/werkstuk nie.
   I declare that the work contained in this assignment, except where otherwise stated, is my original work and that I have not previously (in its entirety or in part) submitted it for grading in this module/assignment or another module/assignment.

Studentenommer / *Student number*	Handtekening / *Signature*
Voorletters en van / *Initials and surname*	Datum / *Date*

Abstract

Optimal energy storage scheduling is a decision problem where an agent must choose the optimal sequence of action to minimize some cost function. The report covers a model free approach to this problem in a residential setting with solar generation, examining four reinforcement learning algorithms in contrast with a rule-based approach. A secondary function of this report is to determine the affect of using these algorithms when finding the optimal size of the solar-battery system. The results show that the best reinforcement learning algorithm, proximal policy optimisation, shows equal performance to one of the rule-based approaches, though performs worse than the other rule-based approaches. The algorithm shows an ability to generalize to differently sized systems. The optimization of the system size using PSO gave similar results to online tools when done on the rule-based control method. However applying the same method to the PPO algorithm resulted in an undersize system.

Contents

List of Figures

List of Tables

Nomenclature

Variables and functions

S	The set of all possible states.
s	A specific state $s \in S$.
N	The total number of time steps in the environment.
T	The set of all time steps $t \in \{1, 2, 3, \ldots, N, N+1\}$.
t	The given time for $t \in T$.
S_t	The set of all possible states s at time t.
$A_{s,t}$	The set of all possible actions in state s at time t.
a	A given action a for $a \in A_{s,t}$
$r_t(s, a)$	The reward generated at time t in state s after action a
$p_t(s'\|s, a)$	The probability that at time t the next state will be s' given s and a such that $a \in A_{s,t}$, $s \in S$ and $s' \in S_{t+1}$.
R_t	The expected return of all future actions, usually exponentially discounted by γ
γ	The discount rate for future rewards, $0 \leq \gamma \leq 1$.
$\pi(s, a)$	The probability of taking action a in state s following policy π.
$V^{\pi}(s)$	The state-value function under policy π. The expected return from state s under the current policy
$Q^{\pi}(s, a)$	The action-value function under policy π. The expected return of taking action a under policy π in state s.
δ_t	The TD error which indicates how the actual outcome differs from the expected outcome.
$p^t_{best_i}$	The personal best position of all time of particle i.
g^t_{best}	The global best position of all time.
v_i^{t+1}	The updated velocity component of particle i for time $t+1$
x_i^t	The position of particle i at time t.
T_a	The air temperature around the PV module.
T_m	The temperature of the PV module.
G	The global irradiance on the inclined plane.
H_{sun}	The height of the sun.
W	The wind speed.
P	The power output of a PV module.

A	The area of a PV module.
eff_{nom}	The nominal efficiency of a PV module.
eff_{rel}	The relative efficiency of a PV module based on ambient conditions.
$C_{batt}^{Capital}$	The capital cost of purchasing a battery system.
E_{BATT}^{Nom}	The nominal battery storage capacity
N_{cycle}	The number of charge discharge cycles in a battery's lifespan.
$SOC(t)$	The state of charge of the battery at time t.
SOC_{min}	The minimum state of charge permissible for the battery.
$E_{batt}(t)$	The amount of energy entering the battery at time t in kWh, negative means the battery is discharging.
η_{batt}	The one way trip efficiency of the battery.
$C_{batt}^{Power}(t)$	The cost of battery power at time t.
$C_{batt}^{O\&M}$	The operation and maintenance cost of the battery in R/hour.
$C_{batt}(t)$	The cost of the battery at time t in R.
$C_{grid}(t)$	The cost of grid electricity at time t in R/kWh.
$C_{PV}^{O\&M}$	The operation and maintenance cost of the PV in R/hour.
$C_{PV}^{Capital}$	The capital cost of purchasing a PV system.
$C_{PV}(t)$	The cost of the PV system at time t.
$E_{load}(t)$	The energy demand from the residential load at time t
$E_{PV}(t)$	The energy supplied by the PV system at time t
$E_{grid}(t)$	The energy drawn from the grid at time t
$loadshedding(t)$	True or false value depending on if there is load shedding scheduled at time t.
$month(t)$	The current month at time t.
$next_{loadshedding}(t)$	The number of hours until the next scheduled load shedding at time t.
$battery_{clipped}(t)$	True or false depending on if the environment had to limit an agents battery action due to system constraints.
$load_{dropped}^{Total}$	The total amount of load not satisfied during loadshedding in an episode.

Acronyms

A3C Asynchronous Advantage Actor-Critic. 5

ADP Approximate Dynamic Programming. 4

BESS Battery Energy Storage System. 2, 5–7, 39

D3QN Double Dueling Deep Q-learning Network. 5

DDPG Deep Deterministic Policy Gradient. 5, 6, 15

DP Dynamic Programming. 4, 6, 8–10

DQN Deep Q-learning Network. 4–6, 15, 37

GA Genetic Algorithm. 6, 16, 18, 19

HVAC Heating Ventilation and Air Conditioning. 5

LPSP Loss of Power Supply Probability. 6

MDP Markov Decision Process. 4, 8–10

MIP Mixed Integer Programming. 5

MP Markov Process. 5

MS-DQN Multi-Step Deep Q-learning Network. 5

NAD Number of Autonomous Days. 6

NN-DQN Noisy Net Deep Q-learning Network. 5

OLA One-step Look-ahead Algorithm. 4

ORA One-step Roll-out Algorithm. 4, 5

PPO Proximal Policy Optimisation. 5, 6, 14, 15, 37–39

PSO Particle Swarm Optimization. 6, 16, 17, 38, 39, 48

PV Photovoltaic. 1–7, 16, 20–22, 37, 39

REFIT Renewable Energy Feed-In Tariffs. 24

RES Renewable Energy Sources. 1, 2

RL Reinforcement Learning. 4–6, 9, 10, 35, 37–39, 48, 49

SAC Soft Actor Critic. 15, 16, 37

TD Temporal Difference. 11, 12, 14

TD3 Twin Delayed Deep Deterministic Policy Gradient. 6, 16, 37

TOU Time of Use. 2, 7

WG Wind Generation. 5, 6

Chapter 1

Introduction

1.1. Background

Recently the global energy demand has increased drastically, primarily driven by the expansion of industrialization and urban development. This brings difficult challenges to mitigating climate change [4].

Promoting the use of Renewable Energy Sources (RES) is required to reduce carbon emissions and establish a sustainable future. Although RES can be some of the most financially viable methods of energy generation their intermittency necessitates the use of non-renewable energy sources such as coal and natural gas [5].

The South African Energy Sector report [6] states that coal makes up 83% of the country's nominal capacity and generates 90% of the total energy generated. With pumped storage and hydro making up 7% of the nominal capacity and wind and solar sources contributing a negligible amount. South Africa grapples with a challenging problem concerning both the cost and availability of electricity. Over the course of 14 years from 2007 to 2021, residential consumers in South Africa experienced an increase of 753% in the cost of electricity [7]. This increase is largely attributed to Eskom's large costs and accumulated debt [8]. South Africa is also currently dealing with the problem of load shedding, which was instated by Eskom to lower the strain placed on the national grid. This became necessary because of insufficient generation capacity brought about by failures at plants due to maintenance backlogs [9].

Since the start of load shedding the installation of small-scale residential Photovoltaic (PV) systems has accelerated. However, due to the absence of incentives and large initial cost only 7% of the installed capacity is in the residential sector [10]. It is estimated that 31% of the global energy demand comes from the residential sector, mainly used for appliances, lighting, heating, and cooling [11]. Typically, the shape of residential load profiles provides a poor match for the shape of PV generation profiles. With residential loads peaking in the early morning and late afternoon whereas PV generation peaks at midday. This results in a large proportion of the PV energy generation going unused [12]. This makes the use of energy storage solutions an attractive method to utilize this excess generation and improve the stability of RES [13]

Environmental factors such as weather conditions have a significant impact on renewable energy generation. A Battery Energy Storage System (BESS) is a useful approach for dealing with the intermittency and substantial uncertainty associated with RES. The BESS not only mitigates the instability from RES but can also be used to optimize the economy of the energy system based on Time of Use (TOU) tariffs.

1.2. Problem Statement

The primary aim of this report is to develop a machine-learning model that minimizes the operating cost of a PV battery system installed on a residential home through the optimal scheduling of the battery charge/discharge actions. The model should make decisions on when the battery should be charged or discharged based only on the current state of the system. The secondary goal of this report is to optimize the size of the installed PV battery system based on the behaviour of the machine-learning model to reduce the initial investment costs.

1.3. Objectives

The objectives for the primary goal of this report is to design a machine-learning algorithm that is capable of:

1. Controlling a PV battery system while reducing operating costs. However, this should be balanced with other goals such as meeting load during load shedding.
2. Performing similarly across the different battery and PV system sizes.
3. Outperforming simple logic/rule-based scheduling.

The secondary objective of this report is to:

1. Optimize the sizing of the PV and battery systems with different control methods to produce the system with the lowest invested capital needed while the system still meets specified requirements such as % load maintained during load shedding.

1.4. Scope

This report focuses on the optimal scheduling of a residential PV battery system. Thus, all the complexities of the electrical systems such as inverters and battery and solar panel architecture are not modelled. Only the necessary complexities needed to model the cost of energy from these systems are included such as the battery efficiency and number of charge/discharge cycles in its lifetime. As this report focuses on the control of the BESS the load is not controllable, i.e. it is fixed in time and cannot be shifted. However, the load is flexible during load shedding such that if the system can't meet the full load it instead satisfies a reduced load. The scope of this report was further limited by excluding the forecasting of load and PV generation profiles.

1.5. Report Overview

1. Chapter 1 covers the contextual information for the problem statement. The objectives are laid out along with the scope of this report.
2. Chapter 2 identifies the current techniques used in literature to solve optimal scheduling problems and system sizing optimization.
3. Chapter 3 provides detailed explanations for the different techniques examined in this report.
4. Chapter 4 details the preprocessing done on the provided datasets and discusses notable features found in them.
5. Chapter 5 explains the design process of the system as well as the modifications to the models and environment to enhance training.
6. Chapter 6 presents the results and analyses the system's performance and behaviour.
7. Chapter 7 summarizes the project and concludes whether the objectives were met or not.

Chapter 2

Literature Review

In this chapter, the existing literature for energy management scheduling and optimal system sizing is reviewed. The different techniques and applications used in each domain are explored, evaluating the performances and methodologies.

2.1. Optimal Energy Storage Scheduling

Optimal energy storage scheduling refers to the process of choosing the optimal sequence of charge/discharge actions that minimizes some cost function. The majority of approaches to solving this problem take a Reinforcement Learning (RL) approach. However, another common approach is to use Dynamic Programming (DP) as in [14]. Kwon et al. propose two Approximate Dynamic Programming (ADP) algorithms, One-step Look-ahead Algorithm (OLA) and One-step Roll-out Algorithm (ORA). These ADP algorithms are introduced as DP is computationally intensive and requires large volumes of information about the system model [15]. It is shown that ORA outperforms the other algorithms and finds solutions within 2% of the optimum.

Q-learning is one of the most simplest value-based RL algorithms. Thus, it is the earliest RL technique applied in energy systems because of its model-free approach. However, it is limited to problems with a low action or state space [16]. Q-learning algorithms outperform continuous RL algorithms when dealing with Markov Decision Processs (MDPs) with simple action spaces and only moderate amounts of data available [17].

Waldemar et al. [18] implement a Deep Q-learning Network (DQN) to optimize a storage-integrated photovoltaic PV system and show that the algorithm outperforms rule-based heuristics that require domain-specific knowledge.

Zhou et al. [19] evaluate the difference between ϵ - greedy and softmax action exploration policies for a Q-learning algorithm. The difference between 15-minute, 30-minute and 1-hour time action intervals are also explored. The ϵ - greedy exploration policy performs better than the softmax policy across all scheduling intervals. A comparison between the two behaviours shows that the ϵ - greedy exploration policy takes fewer battery actions.

This is attributed to it being more beneficial to choose actions with larger immediate rewards rather than exploring non-greedy options. However, they conclude that the softmax exploration policy may perform better with higher dimensional action and state spaces. For the duration of action intervals, the paper shows that the number of battery actions increases the shorter the interval resulting in higher operating costs.

A novel deep RL algorithm, Mixed Integer Programming (MIP)-DQN, that strictly enforces all operational constraints is proposed by Shengren et al. [20] The DQN that learns the action-value function is trained as normal with a penalty added to the reward function to loosely enforce the power balance constraint. The strict enforcement is only applied to online employment by formulating the DQN with fixed parameters as a mixed integer programming MIP problem and then solving it. Thus, the optimal action obtained from the MIP formulation strictly enforces the constraints in the action space.

A cyclic time-dependent Markov Process (MP) is used to model the fluctuating patterns in demand, electricity pricing and solar energy in an energy management system. A Q-learning algorithm is then trained on this cyclic time-dependent MP and compared to a benchmark using a ORA. This experiment shows that the proposed approach could reach close to optimal efficiency for BESSs. It is also noted that changes in the electricity price affected the Q-learning algorithm's ability more than other parameters [21].

Harrold et al. [22] use a Rainbow DQN to control a battery in a microgrid to perform energy arbitrage and use PV and Wind Generation (WG) sources more efficiently. The Rainbow DQN is a combination of the following variations of Q-learning: Double Dueling Deep Q-learning Network (D3QN), Multi-Step Deep Q-learning Network (MS-DQN) and, Noisy Net Deep Q-learning Network (NN-DQN). The goal of the combinations is to improve training stability, convergence rate and noise rejection. They evaluated the algorithm against an actor-critic method, a linear programming model and other variations on the DQN algorithm showing the Rainbow DQN outperforming all of them. They also found that as the action-space is increased from 5 to 9 the performance of DQN variants decreases as the agents can not effectively explore the state-action pairs. However, the Rainbow DQN increased its performance slightly. It is suggested this is due to the distributional approach to value calculations.

Nakabi and Toivanen [23] compare variations on DQNs, actor-critic and Proximal Policy Optimisation (PPO) methods aiming at enhancing the energy management of a microgrid. They find that the Asynchronous Advantage Actor-Critic (A3C) with added experience replay and semi-deterministic training has a higher rate of convergence and superior policies when compared with the other methods.

Liang et al. [24] use a Deep Deterministic Policy Gradient (DDPG) algorithm to minimize the energy cost of residential homes through scheduling of the Heating Ventilation and Air Conditioning (HVAC) systems. They demonstrate the robustness of the DDPG algorithm by introducing thermal disturbances and show that it saves energy costs compared to the optimal solution when these disturbances are introduced. When comparing the differences between DQN and DDPG algorithms on building energy systems both [25] and [26] find that the continuous action space of DDPG is superior to the discrete action space of DQN.

A model-based RL approach is taken by Xu et al. and applied to four advanced RL algorithms. PPO, DQN, DDPG and, Twin Delayed Deep Deterministic Policy Gradient (TD3). They find that all algorithms result in an improvement over the baseline control algorithm. However, TD3 performs the best. [16]

2.2. Optimal System Sizing

Genetic Algorithm (GA) are a common optimization method for the sizing of battery and renewable energy systems. Using a GA to find the optimal number of components for a hybrid PV WG system is contrasted with those of conventional methods, such as DP and gradient-based techniques. It is shown that the GA could find the global optimum with less computational complexity in comparison to the conventional methods [27].

GAs can be used to optimize other parameters of a system and not just the sizing. Yang et al. [28] use a GA to obtain the global optimum system configuration for a hybrid PV WG system that will achieve the desired Loss of Power Supply Probability (LPSP) for a telecommunication relay station. The decision variables are extended to include the slope angle of the PV model and the wind turbine installation height. It is shown that the GA finds a near-optimum solution in the early evolutionary generations. Their findings indicate that at an LPSP of less than 2% the optimal slope angle for the PV array is higher than suggested by the latitude. This is attributed to it being more favourable to install the PV array at a greater angle to maximize power production during the low solar radiation times of the year. However, the solar radiation and wind speed profiles could affect the optimization result.

Allwyn et al. [29] used a GA tool from MATLAB to optimize the cost by sizing a PV battery system for streetlights. The main focus is to compare both scenarios with and without consideration of the Number of Autonomous Days (NAD). It is discovered that while the number of batteries increases in the case considering NAD the number of panels decreased.

Selhatin and Saban [30] use Particle Swarm Optimization (PSO) on a custom energy management method to find the optimal PV and BESS sizes that minimizes total cost of the system. The effectiveness of the PSO algorithm is compared to the GA and is shown to perform better than the GA did.

Mulleriyawage and Shen [31] explore the optimal sizing of BESS under the benefits of operational optimization over the baseline of self-consumption maximization. They note that operational optimization of PV systems larger than 8 kW peak gives significant results over SCM. Comparing SCM and operational optimization on the BESS sizing with a PV system at an 8 kW peak shows that the optimal BESS capacity is larger for operation optimization. This results in higher investment costs but is shown to have lower annual costs.

Linear programming is used to study the optimal battery sizing for given PV system sizes under South African time of use TOU tariffs. Their results show that if the PV array size matches the peak load the optimal battery size will be 18% of the total daily load consumption which leads to 0kWh unused energy from the PV array. They also find that if the PV array size is increased beyond the peak load the optimal battery size also increases [32].

Chapter 3

Methodology

This chapter covers two sections, optimal energy storage scheduling and system sizing optimization. In each section, the problem will first be defined then the different techniques used in literature will be explained and explored.

3.1. Energy Storage Scheduling

The optimal scheduling of energy storage has had a multitude of different techniques applied to provide the required power at minimal cost. These techniques can be broken up into model-dependent and model-independent techniques. A brief overview of the techniques used in each category will be explored in this section.

3.1.1. Defining the Problem

Sequential decision problems, such as the control of energy storage, often arise in practice and are generally modelled as a MDP. The techniques developed to solve such problems come in two categories, either a model of the MDP is required and planning is done based on this model e.g. via DP. Or the model of the MDP is unknown and learning must be done through interactions with the unknown MDP e.g. temporal-difference techniques. [33]

MDPs are defined by the following characteristics $\langle S, A, T, r(s', s, a), p(s'|s, a)\rangle$. The set T contains points of time when the system is observed and can be affected by decisions. Energy storage scheduling will be classified as a discrete-time finite horizon problem, i.e. $T \in \{1, 2, \ldots, N, N+1\}$. When the decision maker makes a decision at time t they can choose an action a from the set of all possible actions at time t given state s, denoted as $A_{s,t}$ which is a subset of A.

The set S is the set of all possible states, where S_t is the set of all possible states at time t. For finite horizon problems S_t is valid for $t \in \{1, 2, \ldots, N, N+1\}$ as this includes the termination state. Decisions are only possible for $t \in \{1, 2, \ldots, N\}$. The reward function is defined as the expected reward that the decision-maker receives based on its action in a given state $r(s', s, a) = E\{r_{t+1}|a_t = a, s_t = s, s_{t+1} = s'\}$. $p(s'|s, a)$ describes the probability that the systems will be in state $s' \in S_{t+1}$ given that action $a \in A_{s,t}$ was chosen in-state $s \in S_t$

Generally $A_{s,t}$ and S_t are either, finite, countably infinite or bounded sections of the real number line. A key property of MDPs is the Markov property which states that the environments' response at s_{t+1}, r_{t+1} is only dependent on the current state and action s_t, a_t [34].

$$P(s_{t+1} = s', r_{t+1} = r | s_t, a_t, s_{t-1}, a_{t-1}, \dots s_0, a_0) = P(s_{t+1} = s', r_{t+1} = r | s_t, a_t,) \tag{3.1}$$

3.1.2. Reinforcement Learning

RL is not defined as a group of techniques used to solve a problem but rather the type of problem. The two broad categories of techniques used to solve RL problems are search techniques and utility estimation. The first set of techniques searches in the space of behaviours for the behaviour that produces the most reward in the environment, e.g. genetic algorithms. Utility estimation methods such as DP and statistical techniques such as Monte Carlo tree search are used to estimate the rewards of taking different actions in different states of the environment [1, 35].

A standard representation of the RL model displayed in Figure 3.1 shows an agent connected to some environment through the observed state i and reward signal r which are both based on the current state s of the environment. From these signals, the agent chooses an action a to then affect the state of the environment [1]. A big challenge that RL presents is finding the right balance between exploitation and exploration. In order for an RL agent to receive a large reward it must choose actions that have been tried already and shown to generate a lot of reward. However, to discover these actions the RL agent must explore new actions that have not been tried before. The challenge arises as a balance between exploitation and exploration is necessary for the agent to succeed at the task [35].

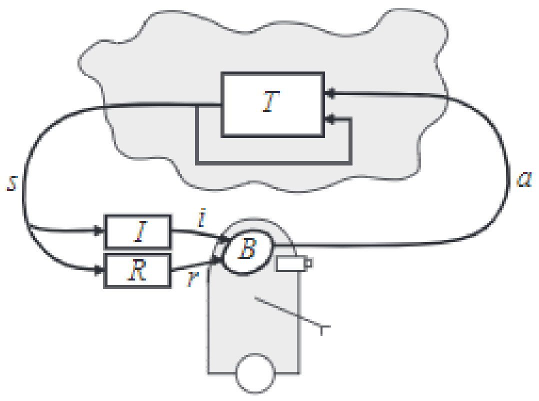


Figure 3.1: The standard model for Reinforcement Learning. [1]

The objective is to choose actions that maximize the long-term reward of the agent. In other words, the objective is to maximize the expected return R_t where $R_t = r_{t+1} + r_{t+2} + \cdots + r_T$. To deal with tasks where $T \to \infty$ and the above formulation results in the return being infinite the reward is discounted over the future. R_t is defined mathematically as:

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{3.2}$$

Where γ is that discount rate, $0 \leq \gamma \leq 1$.

A policy, $\pi(s, a)$, is a mapping from state s to action a and is defined as the probability of taking action a in state s. Policies can be both stochastic and deterministic. The value of a state under a given policy is denoted as $V^\pi(s)$ and defined as the expected return from following policy π starting in state s.

$$\begin{align} V^\pi(s) &= E_\pi \left\{R_t | s_t = s\right\} \tag{3.3} \\ &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,\middle|\, s_t = s \right\} \tag{3.4} \\ &= \sum_a \pi(s,a) \sum_{s'} p(s'|s,a) \left[r(s', s, a) + \gamma V^\pi(s') \right] \tag{3.5} \end{align}$$

Equation 3.5 is known as the Bellman equation and is the basis for most RL techniques.

The action-value function, Q^π is similar to the state-value function, V^π shown above and is defined as the expected return of taking action a in state s following policy π.

$$\begin{align} Q^\pi(s,a) &= E_\pi \left\{R_t | s_t = s, a_t = a\right\} \tag{3.6} \\ Q^\pi(s,a) &= r(s', s, a) + \gamma \sum_{s'} p(s'|s,a) \sum_a \pi(s', a') Q^\pi(s', a') \tag{3.7} \end{align}$$

Where s' and a' and the next state and action.

3.1.3. Dynamic Programming

Dynamic programming in general refers to the process of taking a recursive algorithm and first finding solutions to sub-problems. The solutions to the sub-problems are stored avoiding repeated calculations when calculating the overall solution. This drastically reduces the computational complexity of finding an optimal solution to the overall problem [36]. DP is a model-based technique, i.e. it requires that the dynamics of the environment are fully known. When applying DP to solve an MDP it is usually assumed that the set of states and actions are finite. DP algorithms follow an iterative process of policy evaluation and policy improvement. Policy evaluation is itself an iterative process where at each iteration the Bellman equation 3.5 is used to update the state-value function.

$$V_{k+1}(s) = \max_a \sum_{s'} p(s'|s,a)\left[r(s',s,a) + \gamma V_k(s')\right] \forall s \in S \tag{3.8}$$

Since the Bellman equation assures equality, $V_k = V^\pi$ is a final state for this update equation and thus $V_k \to V^\pi$ as $k \to \infty$. To calculate each successive approximation the update equation is applied across all states s. This step can be made more efficient by applying the update in place instead of storing old and new values. This results in new values sometimes being used to update other states in the same iteration. This in-place update usually results in faster convergence. Iterative policy evaluation only converges as $k \to \infty$ it is typically stopped when the update to the value function is below a specified threshold.

The value function is calculated for a given policy π to determine if a new policy is better than the current one. This is done by evaluating if it is better to choose an action not chosen by the current policy $a \neq \pi(s)$ or to follow the current policy. The value of following the current policy from s is $V^\pi(s)$, one way to determine if a new policy is better is to consider selecting a in s and then continuing with the existing policy π. The value of this action is calculated as the sum of the immediate reward and the expected value of following the existing policy

$$Q^\pi(s,a) = \sum_{s'} p(s'|s,a)\left[r(s',s,a) + \gamma V^\pi(s')\right] \tag{3.9}$$

If the action-value calculation $Q^\pi(s,a)$ is greater than $V^\pi(s)$ then it is reasonable to assume it would be better to select action a every time. Thus, the new policy would be better than the old policy. This can be generalized to apply to policies and not just actions. Given any pair of policies π and π' if $\forall s \in S$:

$$Q^\pi(s, \pi'(s)) \geq V^\pi(s) \tag{3.10}$$

Then π' must be equal to or better than π.

3.1.4. Temporal Difference Learning

Temporal Difference (TD) learning takes the ideas discussed in section 3.1.3 about updating estimations of value functions. However, instead of basing the updates on a model of the environment TD methods use the experience from interacting with the environment to update the value functions. Unlike Monte Carlo methods that have to wait until the end of an episode before updating their estimates of $V(s_t)$, TD methods only wait until the next time step. Instead of waiting for the total return to be calculated TD methods use the reward received and the estimate $V(s_{t+1})$ [35].

$$V(s_t) \leftarrow V(s_t) + \alpha \left[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\right] \tag{3.11}$$

This results in TD methods learning their estimates based on other estimates, i.e. they bootstrap.

3.1.5. Q-Learning

Q-learning is an off-policy TD learning algorithm that directly approximates the optimal action-value function Q^*. This is done independently of the policy being followed, though the policy still affects which state-action pairs are encountered and updated. Convergence occurs as long as all state-action pairs continue to be updated. To ensure sufficient visitations an ϵ-greedy policy is used where with a probability $1 - \epsilon$ the action chosen is the one corresponding to the highest Q value, $\pi(s_t) = \max_a Q(s_t, a)$. With a probability ϵ the action is chosen uniformly at random. This results in the following update rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)\right] \tag{3.12}$$

A big limitation of Q-learning is that it is limited to a discrete number of states. This can be circumvented by discretizing the state space however this can run into the curse of dimensionality for large state spaces [35]. A different approach is to use a Deep Q Network (DQN) to approximate the action-value function Q. This approximation is represented as $Q(s, a, \theta_i)$ where θ_i represents all the parameters for the neural network at iteration i. Reinforcement learning is known to be unstable and diverges when a nonlinear function approximator is used to approximate the action-value function. To mitigate this instability two ideas are introduced. First, a framework called experience replay where the agents experience $e_t = (s_t, a_t, r_t, s_{t+1})$ are stored in a data set D and then uniformly sampled from to perform minibatch $(s, a, r, s') \sim U(D)$ updates to the Q-learning function. This removes the correlation in observation sequences. The second method of dealing with instability involves fixing the target values $Q(s', a', \theta_i^-)$ and only periodically updating them [37]. Thus when updating the policy network it uses the following loss function:

$$L_i(\theta_i) = E_{(s,a,r,s')\sim U(D)}\left[\left(r + \gamma \max_{a'} Q(s', a', \theta_i^-) - Q(s, a, \theta_i)\right)^2\right] \tag{3.13}$$

Where θ_i^- is the parameters for the target network at iteration i. The target network is updated by cloning the policy network every C steps. The described framework is shown in Figure 3.2. A big drawback of Q-learning is that it has an overestimation bias of the Q function due to the noisy value estimate and the maximization step. The error is compounded further as the estimate is updated using the same estimate of a subsequent state [38].

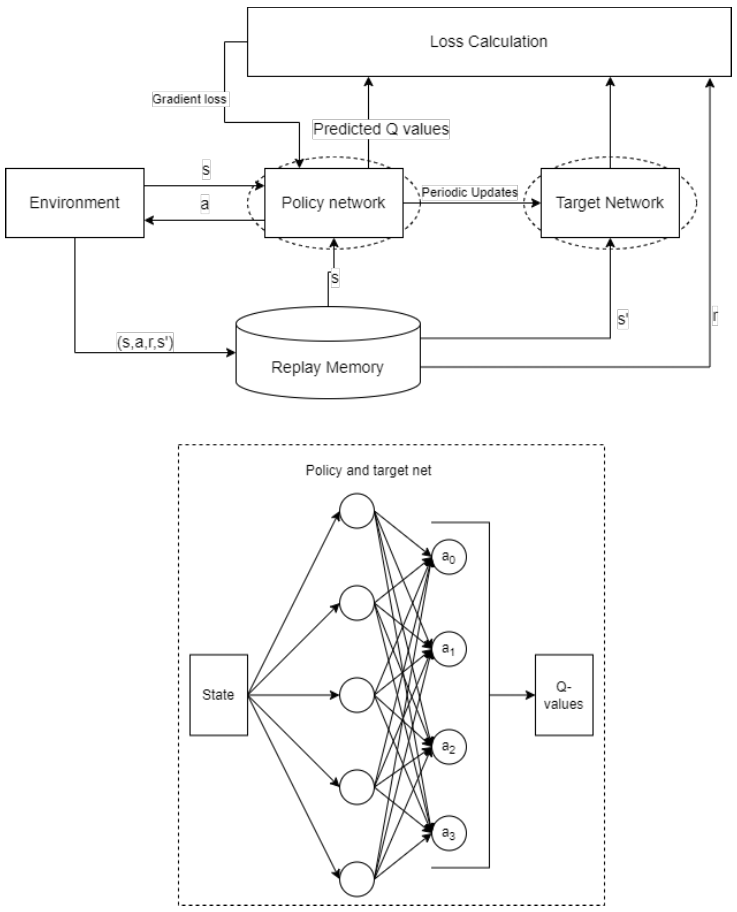


Figure 3.2: Architecture of a DQN learning algorithm.

3.1.6. Policy Gradient

A large limitation with greedy policy selection as described in section 3.1.5 is that small changes in the action-value function approximation can determine if the action is selected or not. Thus rather than using an approximate value function to determine a policy, the policy is approximated directly with its own parameters. A natural representation of this policy would be a neural network that takes the state as input and outputs action probabilities. Let θ represent the policy parameters and ρ the performance of following that policy. The policy parameters are then updated according to the gradient:

$$\nabla_\theta = \alpha \frac{\partial \rho}{\partial \theta} \tag{3.14}$$

Where α is the step size. Using this approach small changes in θ cause small changes in the policy π [39].

Silver et al. [40] show that the policy can be deterministic and that this is a limiting case of the stochastic policy gradient method.

3.1.7. Actor Critic

Actor-critic methods differ from Q-learning in that they have a separate structure to model the policy that is independent of the value function. This is labeled the actor while the value function is known as the critic. The critic typically uses the state value function $V(s)$. Following an action the critic evaluates the new state and outputs an evaluation known as the TD error δ to indicate how the outcome compares with expectations.

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{3.15}$$

When δ is positive following action a_t it shows that a_t should be selected more often in the future. Otherwise, it should be selected less often. The advantage of the actor-critic architecture see Figure 3.3 is that it allows an infinite number of actions, such as a continuous action space, this is due to the explicitly stored policy. Whereas methods that only learn the action-value function require a search through the action space to pick an action [35].

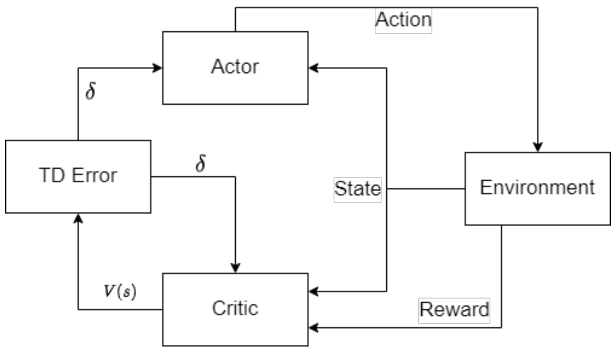


Figure 3.3: Actor critic architecture.

3.1.8. Proximal Policy Optimization

PPO is a policy gradient method that has the advantages of trust region policy optimization while being simpler to implement and more general [41]. A commonly used gradient estimator for policy gradient methods takes the form:

$$\hat{g} = E\left[\nabla_\theta \log \pi_\theta(a_t|s_t) R_t\right] \tag{3.16}$$

where R_t is the total accumulated return $\sum_{k=0}^{\infty} \gamma^k r_{t+k}$. The above formulation is an unbiased estimation of $\nabla_\theta E[R_t]$. However, the variance of this estimate can be reduced by subtracting a baseline $b_t(s_t)$. The approximate state-value function is normally used as the baseline $b_t(s_t) \approx V^\pi(s_t)$. This results in the advantage function $A(s_t, a_t) = Q(a_t, s_t) - V(s_t) = R_t - b_t$ as R_t is an estimation of Q^π. This is an actor-critic architecture as discussed in section 3.1.7 where the advantage function A_t fills the same role as δ in equation 3.15 [42]. What differentiates PPO from other policy gradient methods is the clipping of the surrogate objective function:

$$L^{CLIP}(\theta) = E\left[\min\left(r_t(\theta)\hat{A}_t, clipr_t(\theta), 1-\epsilon, 1+\epsilon\right)\right] \tag{3.17}$$

Where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ and ϵ is a hyperparameter. This clipping ensures that policy updates don't move the policy too far away from the old policy which makes the training process much more stable [41]. The final objective function is as follows:

$$L^{PPO}(\theta) = E\left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) + c_2 S[\pi_t heta](s_t)\right] \tag{3.18}$$

where S represents an entropy bonus to encourage exploration and L_t^{VF} is a square-error loss term $(V_\theta(s_t) - V_t^{targ})^2$ used to update the state-value function/baseline. c_1, c_2 are simply coefficients used to control training behaviour. PPO is an on-policy method which means it can not use the experience replay architecture discussed in section 3.1.5.

3.1.9. Soft Actor Critic

Typical actor-critic algorithms are based on on-policy policy gradient methods. On-policy training tends to improve stability but results in poor sample efficiency. DDPG uses an action value estimator to enable off-policy learning and can be seen as a combination of a DQN algorithm and a deterministic actor-critic algorithm. However, DQN is challenging to stabilize and sensitive to hyperparameters and on-policy methods such as PPO tend to produce better results albeit at lower sample efficiencies. Soft Actor Critic (SAC) combines off-policy training with a stochastic actor. Similarly to PPO, SAC includes an entropy term in the objective function:

$$L(\pi) = \sum_{t=0}^{T} E\left[r(s_t, a_t) + a\mathcal{H}(\pi(\cdot|s_t))\right] \tag{3.19}$$

Where the temperature parameter α determines the weighting of the entropy term. This adjusts the randomness of the policy. The parameterized value functions $V_\psi(s_t), Q_\theta(s_t, a_t)$ can be represented by neural networks while the policy $\pi_\phi(a_t|s_t)$ as a Gaussian with mean and covariance specified by neural nets. SAC utilizes 2 Q functions to reduce the overestimation of action values that degrade performance. The two functions are trained independently and the minimum of the two is used when calculating the gradient. Training involves alternating between collecting experiences to store in a replay buffer and applying gradient updates from batches taken from the replay buffer. Usually, multiple gradient steps are taken for each environment step.

3.1.10. Twin Delay Deep Deterministic Policy Gradient

TD3 is developed to counteract the overestimation bias discussed in section 3.1.5 and was developed at the same time as the SAC algorithm. It counteracts the overestimation bias through a clipped variant of Double Q-learning in an actor-critic architecture. This results in a pair of actors $(\pi_{\phi_1}, \pi_{\phi_2})$ and critics $(Q_{\theta_1}, Q_{\theta_2})$. Where the actors are optimized against their individual critics. The minimum of the two Q estimates is used in the update of the clipped double Q-learning:

$$y_1 = r + \gamma \min_{i=1,2} Q_{\theta_i}(s', \pi_{\phi_1}(s')) \tag{3.20}$$

A secondary benefit of this approach is that the minimum of a set of random variables decreases as the variance increases. Thus, the minimization in equation 3.20 leads to a preference for states with lower variance value estimates resulting in safer policy updates [38]. To address the concerns with deterministic policies overfitting to narrow peaks, noise is added to the target policy.

$$y = r + \gamma Q_\theta(s', \pi_\phi(s') + \epsilon) \tag{3.21}$$

$$\epsilon \; clip(\mathcal{N}(0, \sigma), -c, c) \tag{3.22}$$

The noise is clipped to keep the target close to the original action.

3.2. System Sizing Optimization

This section covers the two main techniques, PSO and GA, used when finding the optimal parameters for sizing a pv battery system.

3.2.1. Defining the Problem

The problem of optimally sizing a PV and battery system requires some form of search through the parameters space of battery and PV sizes. General optimization can be defined as follows:

$$\min_{x \in \mathcal{R}^n} f_i(x), (i = 1, 2, \ldots, M) \tag{3.23}$$

Subject to

$$h_j(x) = 0, (j = 1, 2, \ldots, J), \tag{3.24}$$

$$g_k(x) \leq 0, (k = 1, 2, \ldots, K) \tag{3.25}$$

where M is the number of objective functions $f_i(x)$ to be optimized in the space $\mathcal{R}^n$ spanned by the decision variables x subject to constraints $h_j(x), g_k(x)$ [43]. Depending on the nature of the parameter space and the properties of the objective function different search techniques are better suited than others. One of the more common search techniques, gradient descent, requires that the objective function is differentiable.

3.2.2. Particle Swarm Optimization

Swarm intelligence refers to the behaviour that emerges from multiple interacting unintelligent agents. The whole system of agents functions as some sort of collective intelligence due to the information shared among agents demonstrating self-organization features [43]. Each particle in the swarm moves towards its previous best position p_{best} and the global best position g_{best}. A standard PSO is described mathematically as:

$$p^t_{best_i} = x^*_i | f(x^*_i) = \min_{k=1,2,\ldots,t} f(x^k_i) \tag{3.26}$$

where

$$i \in \{1, 2, \ldots, N\} \tag{3.27}$$

and

(3.28)

$$g^t_{best} = x^t_* = \min_{k=1,2,\ldots,t;i=1,2,\ldots,N} f(x^k_i), \tag{3.29}$$

where i denotes a particle's index, t is the current iteration count and N is the total number of particles. The update rules for a particles position x and velocity v is defined as:

$$v^{t+1}_i = \omega v^t_i + c_1 r_1 (p^t_{best_i} - x^t_i) + c_2 r_2 (g^t_{best} - x^t_i) \tag{3.30}$$

and

$$x^{t+1}_i = x^t_i + v^{t+1}_i \tag{3.31}$$

Where r_1, r_2 are random vectors uniformly distributed in the range $[0, 1]^D$, D being the dimensionality of the search space. These random vectors encourage the exploration of the search space. c_1 and c_2 are positive constants called the acceleration coefficients and determine how strongly the particles move towards the global or personal bests. ω is the inertial constant and affects how strongly the previous iterations' velocity affects the next iteration. The velocity is clamped to force particles to have proper step sizes to search the space effectively.

3.2.3. Genetic Algorithms

Genetic algorithms are search algorithms inspired by the process of natural selection. They join together a randomized exchange of information with survival of the fittest. Each new generation is created using parts of the fittest from the older generation. Occasionally mutations occur, and a new part is tried in the population. The advantages of GAs over calculus-based search methods like gradient ascent is that the function does not need to be differentiable and gradient ascent algorithms get stuck at local optimums. GAs follow 4 key ideas, they work with a coding of the parameter set, they search using a population not a single point, they only use an objective function as guiding information and use stochastic transition rules and not deterministic rules. The coding of the parameter set must be a finite length string from some finite alphabet. This is considered the genetic material of a member.

A string A is constructed of a sequence of characters $a_1, a_2, a_3, \ldots, a_l$ where a denotes a character selected from some finite alphabet V. An example alphabet would be the binary alphabet $V = 0, 1$. It is possible to have unordered strings. A population of strings $A_1, A_2, \ldots, A_n$ contained in population $\mathbf{A(t)}$ where t denotes the generation. A simple GA would contain the three operations:

1. Reproduction
2. Crossover
3. Mutation

The reproduction process determines which members of the current population $\mathbf{A(t)}$ are copied to the next generation based on their objective function values f_i. Members with higher fitness values have a higher probability of getting selected. This process is the artificial equivalent of natural selection. The crossover operation randomly pairs members of the new generation. Then an integer position k along the string length is chosen uniformly at random from 1 to one less than the string length $l - 1$. Two new strings are created by swapping all characters from position $k + 1$ and l in one string with the other. For example given two paired members with strings $A_1 = [0, 1, 0, 0, 1]$ and $A_2 = [1, 0, 1, 0, 0]$ where $k = 2$ the resulting strings are:

$$A'_1 = [0, 1, 1, 0, 0]$$
$$A'_2 = [1, 0, 0, 0, 1]$$

Each complete string in a population of n strings is a complete idea for performing a particular task. Thus, substrings within each string contain different notions of critical aspects of the task. With this viewpoint, the population contains a multitude of notions that are ranked based on their performance for the task. Reproduction and crossover exploit this information by selecting high-performing notions and combining these notions with other high-performing strings. GAs get the majority of their search effectiveness from reproduction and crossover. However, mutation is necessary as crossover may result in lost notions and the selection process for a new generation removes genetic diversity. The mutation step is the occasional random alteration of a string position which is effectively a random walk through the string space.

Chapter 4

Data Analysis

This chapter examines the main datasets used in this study, namely PV power and the load profiles. Then the pre-processing steps used to clean up the datasets will be explained and justified. Finally, the profiles of the data will be examined.

4.1. Datasets

The dataset used for the PV supply is obtained from the Photovoltaic Geographical Information System (PVGIS) [44]. This system can provide hourly data on solar radiation and PV system operation for any location in Europe and Africa from 2005 to 2020. The location chosen for use is in Stellenbosch as this is the location where the data for the residential load was obtained.

This tool allows the user to optimize the slope and azimuth for fixed, vertical, inclined or two-axis mounting systems. A fixed-axis mounting system was chosen as this is the most common type of mounting for residential PV installation.

The data used contains the following information: The global irradiance on the inclined plane G_i, the sun height H_{sun}, the air temperature T_a, and the wind speed W. The power output of a PV system P is dependent on the global irradiance, the temperature of the module T_m, the area of the module A and the effective efficiency under these operating conditions [45].

$$\begin{aligned}
T_m &= T_a + \frac{G}{U_0 + U_1 \cdot W} \\
T'_m &= T_m - 25 \\
G' &= \frac{G}{1000} \\
eff_{rel}(G', T'_m) &= 1 + k_1 \ln(G') + k_1 \ln(G') + k_2 \ln(G')^2 \\
&\quad + k_3 T'_m + k_4 T'_m \ln(G') + k_5 T'_m \ln(G')^2 + k_6 T'^2_m \\
P &= G/1000 \cdot A \cdot eff_{nom} \cdot eff_{rel}(G, T_m)
\end{aligned}$$

Where the constants U and k are dependent on the PV module type.

The data used for the load profile was acquired from a residential house in Stellenbosch. The readings are instantaneous power readings every half an hour. The readings are separated into 4 elements: The geyser, the plugs, the stove, and the lights and pool. This data was recorded from 2021/05/01 to 2023/06/01.

4.2. Preprocessing

The load dataset contains many erroneous entries. Table 4.1 shows the number of null or negative readings for each laod profile. The negative readings are attributed mostly to faulty equipment on the stove. When a null reading occurs it is at the same date and time for all of the load profiles. A possible explanation for the null readings is that the recording system stopped recording.

Load Profile	Negative readings	Null readings
Stove	28406	419
Lights and Pool	2	297
Plugs	0	297
Geyser	0	297

Table 4.1: Occurrences of negative or null readings in each load profile

As there are only a few null readings and most occur simultaneously across all load profiles all the null entries and their associated date-time entries were removed from the datasets, this removes the null entries but leaves gaps in the date-time entries. All of the negative readings occur between -90 and 0. Since the majority of the readings for the stove are negative it was decided to set all negative readings to 0 rather than discard them.

The data contained in the PV dataset is in hourly increments while the load data is in half-hourly increments. Thus load data will be down-sampled from half-hourly data to hourly data. This was accomplished by averaging the measurements taken at both the beginning of each hour and at the half-hour mark and then logging the resulting value as the total energy consumption in Wh for that hour. If there was only 1 reading for that hour, due to null entries being removed, that reading would be taken as the total energy consumption.

The PV dataset was then synchronized with the load profiles by advancing the date-time by 4 years, resulting in a PV dataset ranging from 2021 to 2023. Subsequently, any PV data points occurring at times not present in the processed load data were excluded. This resulted in two datasets of identical length.

4.3. Data profiles

In this section, the hourly load usage and PV supply are examined and the total daily energy usage is compared throughout the year. The PV system was initially sized at 5 kW peak capacity based on an online sizing tool [46] matching the average monthly consumption and generation.

4.3.1. Hourly Energy Usage

Figure 4.1 shows the average consumption/generation at each time of day in kWh across all days. The following observations are made about the load profile timings. The geyser operates solely within two distinct time frames, from 03:00 to 06:00 and from 15:00 to 20:00. The most substantial energy demand during peak hours primarily originates from the usage of electrical plugs which also exhibit the highest average load. This is likely due to all appliances such as washing machines, dishwashers, kettles and toasters falling under this load profile. Furthermore, the stove displays energy consumption peaks, notably during lunchtime at 12:00 and dinner hours at 19:00.

Figure 4.1 shows that the system is only capable of meeting the energy demand with PV power between the hours of 06:00 and 15:00.

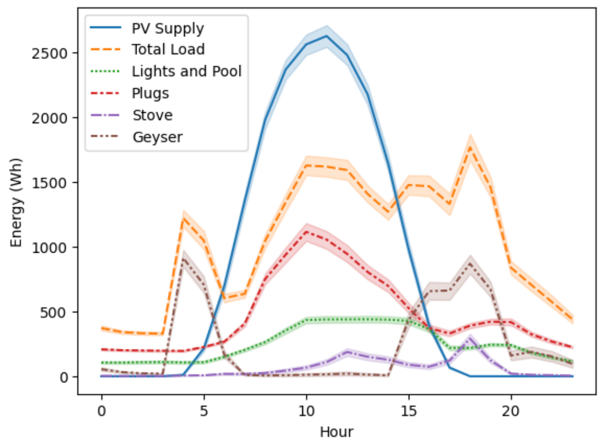


Figure 4.1: Average load vs average PV system power during a day.

4.3.2. Total daily energy usage

Figure 4.2 shows that the mean total daily load is higher in winter compared to summer. It does not vary much between summer and winter months in comparison to the PV supply. However, the peak load in winter is drastically higher than the peak load in summer. This contributes to the complexity of system sizing as sizing a system for winter months results in an oversized system during the summer. Table 4.2 further emphasizes the difference between the load and PV supply in the summer and winter months. On the day with the highest energy generation from the PV system, the load is less than half the mean of 24.7 kWh/day. Conversely, on the day with the highest load, the energy generated by the PV system is below the mean of 19.47 kWh/day.

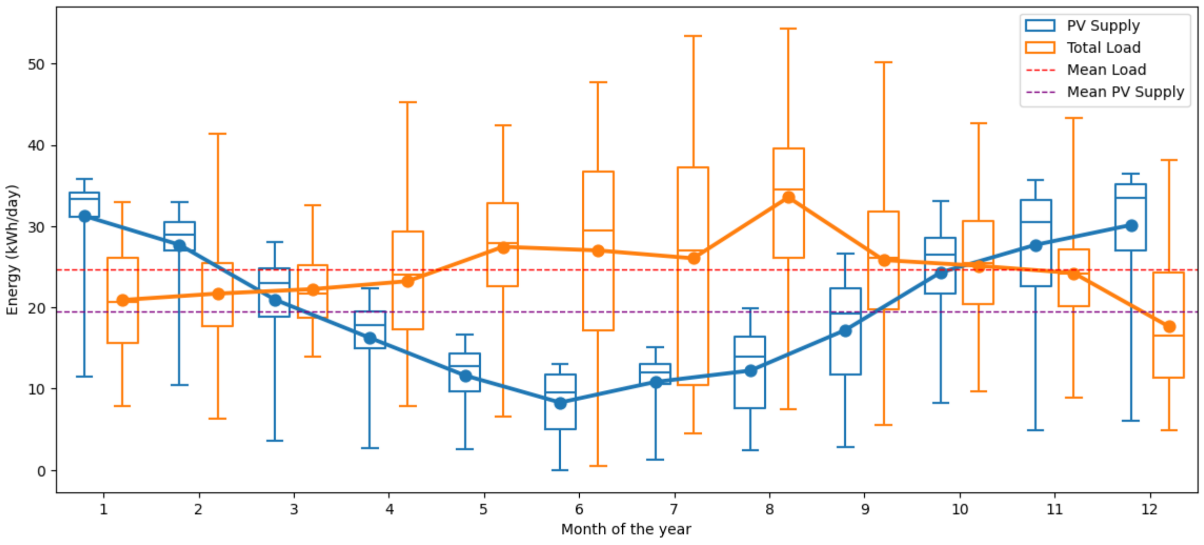


Figure 4.2: Box plot of total daily load and PV system supply during the year.

	Peak total daily load	Peak total daily PV generation
Date	2022/08/15	2021/12/18
Total daily load (kWh/day)	54.35	11.57
Total daily generation (kWh/day)	16.53	36.51

Table 4.2: Comparison of total daily load and PV energy generation on peak days.

Chapter 5

System Design

This chapter will cover the design and specifications of the environment. Then the construction of the basic agents used to benchmark the RL agents will be explained. Next, the experiments used to design the state features and reward function are laid out and the hyperparameters that will be tuned are shown. Finally, the sizing optimization criteria are discussed.

5.1. The Environment

The environment consists of models for the PV system, residential load, grid power supply and battery system. The interaction between these models is shown in Figure 5.1. Due to the low Renewable Energy Feed-In Tariffs (REFIT) it is typically not financially viable to push energy back to the grid [47]. Therefore, it was decided to not allow the model to send excess energy into the grid.

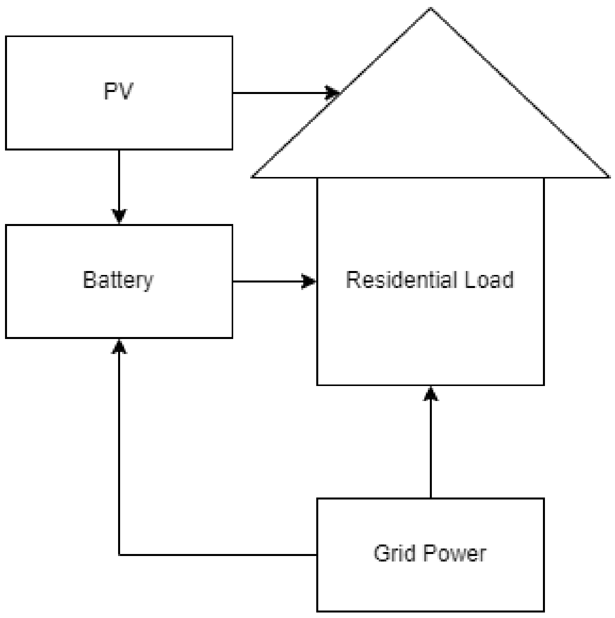


Figure 5.1: System structure of residential PV-battery system.

5.1.1. The Battery

A simple model is constructed for the battery that will approximate the key parameters of the battery relevant to the problem. The chosen type of battery is a lithium-ion battery as these make up the majority of residential PV battery systems [48]. The chosen parameters to model are battery capacity, max C-rate, efficiency, lifetime and State of Charge (SOC). Further battery complexities are not modelled as they are outside the scope of this report. A maximum C-rate of 0.5 was chosen as both show that lithium-ion batteries used in solar typically have C-rates of less than 1 [49, 50]. This results in a 2-hour battery, i.e. the battery can discharge its full capacity in 2 hours. A representative Round Trip Efficiency (RTE) of 85% was used in [51, 52]. However, for this report, the battery model will use the one-way trip efficiency as this will result in a better model of the current energy levels of the battery. If the RTE was used instead either charging or discharging the battery would have no loss of energy. Thus, a one-way efficiency of 92% is used as this results in equivalent RTE. The Eskom report [51] shows that lithium-ion batteries have a max Depth Of Discharge (DOD) of 95%, a lifetime of 12 years and can perform $\approx$ 4400 cycles.

The most critical part of the battery model is how the cost of energy is calculated. A common metric used to estimate this is the Levelized Cost of Energy (LCOE). The LCOE is defined as the average cost of a unit kWh generated and is calculated by dividing the annual cost of a system by the total energy generated [53]. This however does not take into account the effect different DODs have on the lifetime of the battery. The capital cost over time formula used in [54] will be the basis for the cost calculation used in this paper and is described mathematically as.

$$C_{batt}(t) = \frac{C_{batt}^{Capital}}{E_{BATT}^{Nom} N_{cycle} SOC(t)} \cdot \min(E_{batt}(t), 0). \tag{5.1}$$

where $C_{batt}^{Capital}$ is the initial capital cost of the battery system, E_{BATT}^{Nom} is the nominal energy of the battery, N_{cycle} is the number of battery cycles in its lifetime, $SOC(t)$ is the SOC of the battery at time t and $E_{batt}(t)$ is the energy entering the battery at time t. When $E_{batt}(t)$ is negative it means the battery is discharging and when it is positive the battery is charging. This results in a pricing model that has an exponential relationship with the SOC shown in Figure 5.2.

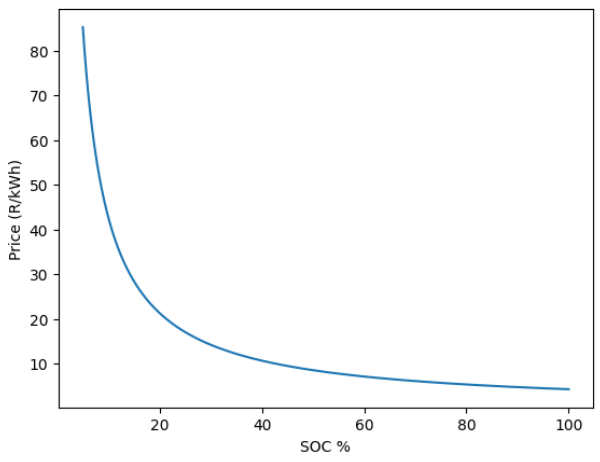


Figure 5.2: Price of energy in R/kWh from the battery at varying SOC levels.

This model however neglects the annual operations and maintenance (O&M) cost found to be 2% of the capital cost [51]. This extra cost was taken from an annual rate to an hourly rate and added to 5.1. The cost of capital of the battery is determined by the size of the battery system. Eskom and NREL [51, 52] both find the capital cost of a battery system to be ≈ R5000/kWh of installed capacity. However, NREL specifies additional inverter-based costs of R1700/kWh resulting in the final calculation for capital cost as

$$C_{batt}^{Capital} = R6700 \cdot E_{BATT}^{Nom} \tag{5.2}$$

Adding the above capital cost calculation and the addition of the operation and maintenance cost to 5.1 results in the final cost calculation

$$C_{batt}^{Power}(t) = \frac{R6700}{N_{cycle}SOC(t)} \tag{5.3}$$

$$C_{batt}^{O\&M} = \frac{0.02 \cdot R6700 \cdot E_{BATT}^{Nom}}{365 \cdot 24} \tag{5.4}$$

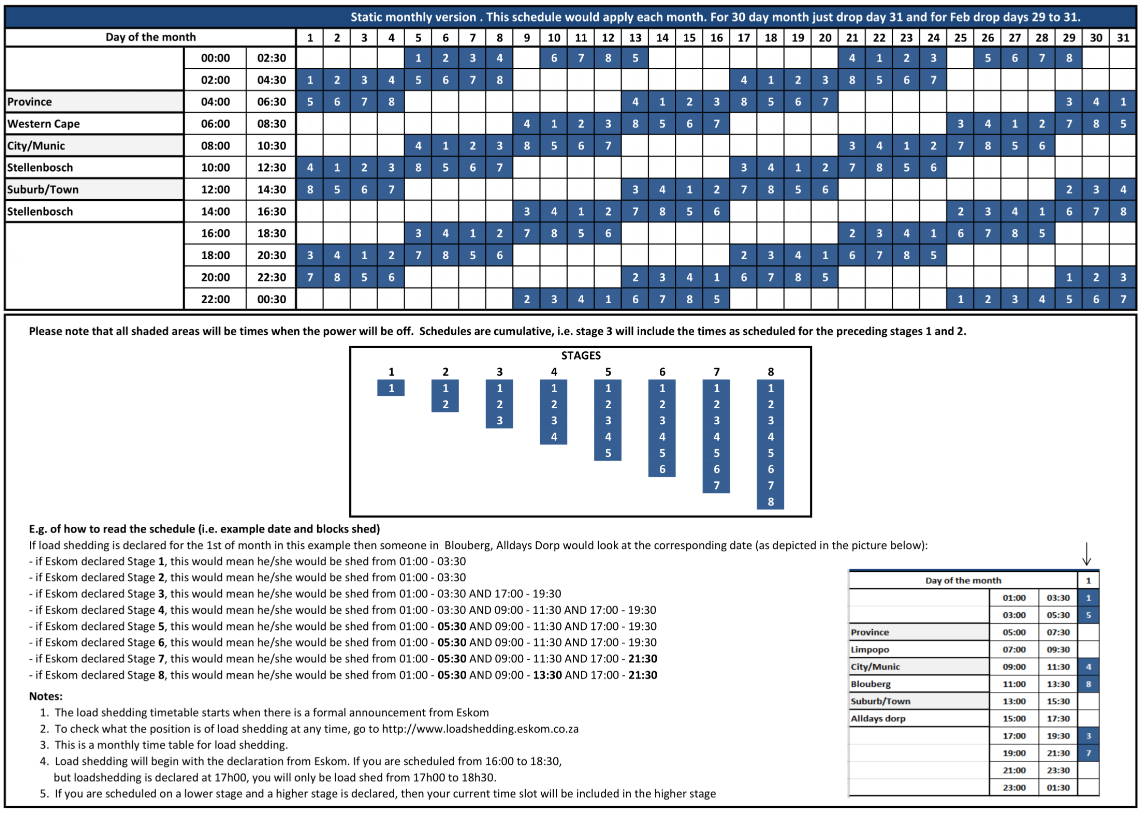
$$C_{batt}(t) = C_{batt}^{Power}(t) \cdot E_{batt}(t) + C_{batt}^{O\&M} \tag{5.5}$$

$$\tag{5.6}$$

where $C_{batt}^{Power}(t)$ is the cost of battery power draw in R/kWh, $C_{batt}^{O\&M}$ is the hourly cost of O&M in R.

5.1.2. Load and Grid Power

The load is modelled by the dataset discussed in 4.1 and is divided into 4 different profiles: the lights and pool, the geyser, the stove and the plugs. As load shedding is modelled, the load will be made flexible so that the system is not trying to satisfy the full load during load shedding. This is achieved by removing the geyser and stove load profiles during load shedding and only supplying the essential loads of the lights and the plugs. During load shedding the environment does not allow any energy to be provided by the grid. The load shedding schedule followed is stage 4 as this was the average load shedding stage so far in 2023 [55]. Figure 5.3 depicts the table used to determine the schedule .



Static monthly version . This schedule would apply each month. For 30 day month just drop day 31 and for Feb drop days 29 to 31.

Day of the month			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	00:00	02:30					1	2	3	4		6	7	8	5								4	1	2	3		5	6	7	8		
	02:00	04:30	1	2	3	4	5	6	7	8									4	1	2	3	8	5	6	7							
Province	04:00	06:30	5	6	7	8									4	1	2	3	8	5	6	7									3	4	1
Western Cape	06:00	08:30									4	1	2	3	8	5	6	7									3	4	1	2	7	8	5
City/Munic	08:00	10:30					4	1	2	3	8	5	6	7									3	4	1	2	7	8	5	6			
Stellenbosch	10:00	12:30	4	1	2	3	8	5	6	7									3	4	1	2	7	8	5	6							
Suburb/Town	12:00	14:30	8	5	6	7									3	4	1	2	7	8	5	6									2	3	4
Stellenbosch	14:00	16:30									3	4	1	2	7	8	5	6									2	3	4	1	6	7	8
	16:00	18:30					3	4	1	2	7	8	5	6									2	3	4	1	6	7	8	5			
	18:00	20:30	3	4	1	2	7	8	5	6									2	3	4	1	6	7	8	5							
	20:00	22:30	7	8	5	6									2	3	4	1	6	7	8	5									1	2	3
	22:00	00:30									2	3	4	1	6	7	8	5									1	2	3	4	5	6	7

Please note that all shaded areas will be times when the power will be off. Schedules are cumulative, i.e. stage 3 will include the times as scheduled for the preceding stages 1 and 2.

STAGES

1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	1
	2	2	2	2	2	2	2
		3	3	3	3	3	3
			4	4	4	4	4
				5	5	5	5
					6	6	6
						7	7
							8

E.g. of how to read the schedule (i.e. example date and blocks shed)

If load shedding is declared for the 1st of month in this example then someone in Blouberg, Alldays Dorp would look at the corresponding date (as depicted in the picture below):

- if Eskom declared Stage **1**, this would mean he/she would be shed from 01:00 - 03:30
- if Eskom declared Stage **2**, this would mean he/she would be shed from 01:00 - 03:30
- if Eskom declared Stage **3**, this would mean he/she would be shed from 01:00 - 03:30 AND 17:00 - 19:30
- if Eskom declared Stage **4**, this would mean he/she would be shed from 01:00 - 03:30 AND 09:00 - 11:30 AND 17:00 - 19:30
- if Eskom declared Stage **5**, this would mean he/she would be shed from 01:00 - **05:30** AND 09:00 - 11:30 AND 17:00 - 19:30
- if Eskom declared Stage **6**, this would mean he/she would be shed from 01:00 - **05:30** AND 09:00 - 11:30 AND 17:00 - 19:30
- if Eskom declared Stage **7**, this would mean he/she would be shed from 01:00 - **05:30** AND 09:00 - 11:30 AND 17:00 - **21:30**
- if Eskom declared Stage **8**, this would mean he/she would be shed from 01:00 - **05:30** AND 09:00 - **13:30** AND 17:00 - **21:30**

↓

Day of the month			1
	01:00	03:30	1
	03:00	05:30	5
Province	05:00	07:30	
Limpopo	07:00	09:30	
City/Munic	09:00	11:30	4
Blouberg	11:00	13:30	8
Suburb/Town	13:00	15:30	
Alldays dorp	15:00	17:30	
	17:00	19:30	3
	19:00	21:30	7
	21:00	23:30	
	23:00	01:30	

Notes:

1. The load shedding timetable starts when there is a formal announcement from Eskom
2. To check what the position is of load shedding at any time, go to http://www.loadshedding.eskom.co.za
3. This is a monthly time table for load shedding.
4. Load shedding will begin with the declaration from Eskom. If you are scheduled from 16:00 to 18:30, but loadshedding is declared at 17h00, you will only be load shed from 17h00 to 18h30.
5. If you are scheduled on a lower stage and a higher stage is declared, then your current time slot will be included in the higher stage

Figure 5.3: Monthly load shedding schedule for Stellenbosch. Taken from Eskom's website for municipal load shedding [2].

The price of grid electricity is based on the Eskoms HomeFlex tariff which is seasonally and Time-of-Use (TOU) differentiated. The high-demand season is from June to August and the low-demand season is from September to May. Figure 5.4 shows the weekly TOU breakdown for the high and low seasons into the peak, standard and off-peak times.

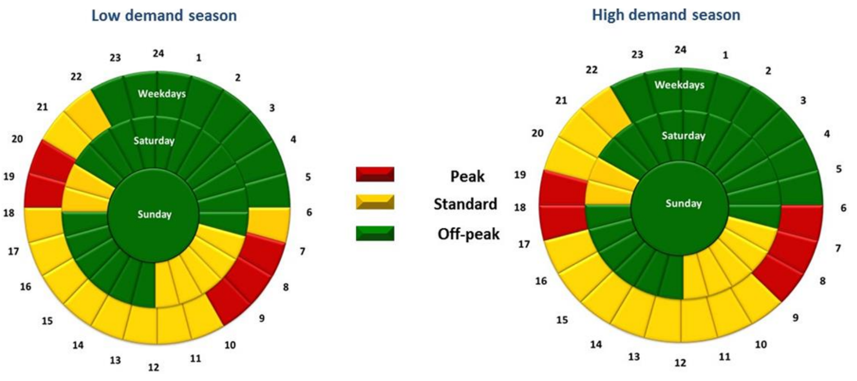


Figure 5.4: Time of use regions for Homeflex. Taken from Eskom's 2023/2024 tariff book [3].

The price of grid power $C_{grid}(t)$ is based on the season and the TOU and follows the pricing (VAT included) shown in Table 5.1.

Time of Use	Demand season	
	High	Low
Off-peak	R1.037/kWh	R0.8992/kWh
Standard	R1.8991/kWh	R1.4101/kWh
Peak	R6.2421/kWh	R2.044/kWh

Table 5.1: TOU tariffs for Homeflex. Adapted from Eskom's 2023/2024 tariff book [3].

5.1.3. Photovoltaics

The PV system is modelled by the dataset covered in section 4.1 which is normalized between 0 and 1 and then scaled by the specified PV system capacity. The total capital cost of the PV system $C_{PV}^{Capital}$ is calculated based on the system size and the rate of R14 410/kWh disclosed in [51]. The operations and maintenance cost for PV is 1.5% of capital cost per year and the system lifetime is 25 years. The cost, $C_{PV}^{Capital}$ is an hourly rate defined mathematically as

$$C_{PV}^{Capital} = \frac{R14410 \cdot E_{PV}^{Nom}}{25 \cdot 365 \cdot 24} \tag{5.7}$$

$$C_{PV}^{O\&M} = \frac{R14410 \cdot E_{PV}^{Nom} \cdot 0.015}{365 \cdot 24}, \tag{5.8}$$

$$C_{PV}(t) = C_{PV}^{Capital} + C_{PV}^{O\&M} \tag{5.9}$$

$$= R0.09/hour \tag{5.10}$$

Where E_{PV}^{Nom} is the nominal capacity of the PV system in kWh.

5.1.4. Environment Operation

The environment operates in discrete time steps where at each time step the environment receives a battery action and must satisfy some power balance and battery constraints. The battery constraints are that the SOC must be within the operating limits specified in section 5.1.1 and that $E_{batt}(t)$ must be within the maximum C-rate. These constraints are formalized as

$$SOC_{min} \leq SOC(t) \leq 1, \tag{5.11}$$

$$|E_{batt}(t)| \leq E_{batt}^{Nom} \cdot E_{c-rate}^{max} \tag{5.12}$$

where E_{c-rate}^{max} is the maximum C-rate. The SOC of the battery is also updated at each time step as

$$SOC(t) = SOC(t-1) + \eta_{batt}E_{batt}(t), E_{batt}(t) > 0 \cdot \tag{5.13}$$

$$SOC(t) = SOC(t-1) + \frac{E_{batt}(t)}{\eta_{batt}}, E_{batt}(t) < 0 \cdot \tag{5.14}$$

where η_{batt} is the battery's one-way efficiency. The reason $E_{batt}(t)$ is multiplied by the efficiency when positive and divided when negative is because $E_{batt}(t)$ is the power external to the battery. When charging the battery more energy must be sent to it then gets stored and when discharging the battery less energy comes out than it loses.

The power balance constraint is that the load $E_{load}(t)$ must be satisfied by some combination of grid power $E_{grid}(t)$, PV power $E_{PV}(t)$ and battery power $E_{batt}(t)$ as shown in equation 5.15.

$$E_{load}(t) = E_{PV}(t) - E_{batt}(t) + E_{grid}(t) \tag{5.15}$$

$E_{batt}(t)$ is subtracted as a negative $E_{batt}(t)$ means the battery is discharging. During load shedding $E_{grid}(t) = 0$ and if the load can't be satisfied the environment reduces the load by removing 1 profile at a time until it is either 0 or able to be satisfied. The load balancing process is detailed in Figure 5.5. Note that $E_{batt}(t)$ is clipped to ensure both battery operation constraints are satisfied before the system tries to satisfy the load.

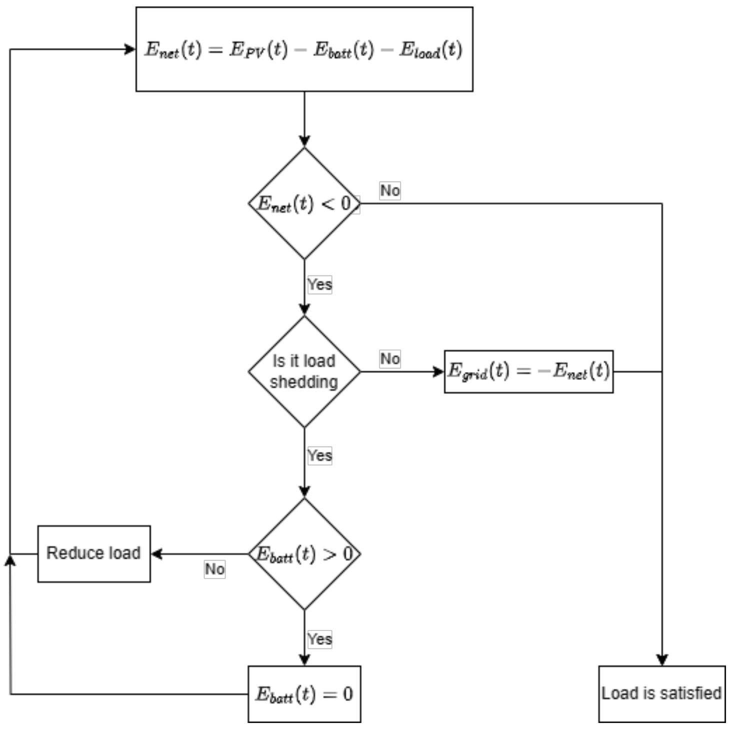


Figure 5.5: Flow diagram showing the environment logic used to satisfy the load.

The total cost for a given time t is defined

$$C(t) = C_{PV}(t) + C_{batt}^{Power}(t) \cdot E_{batt}(t) + C_{batt}^{O\&M} + C_{grid}(t) \cdot E_{grid}(t) \tag{5.16}$$

5.2. Rule Based Control Systems

A benchmark is needed to determine if the RL agents are performing well or not. As finding the actual optimal control sequence is computationally infeasible some simple rule-based control systems are created to benchmark the RL agents against. The control systems developed are for two different modes of operation, emergency mode where the battery is only used during load shedding and is primarily charged from the PV system and is only charged from the grid during off-peak hours. The second agent follows the same pattern but performs basic arbitrage by using the battery during the high-demand season in peak hours. The flow diagram shown in Figure 5.6 describes the rules the system follows.

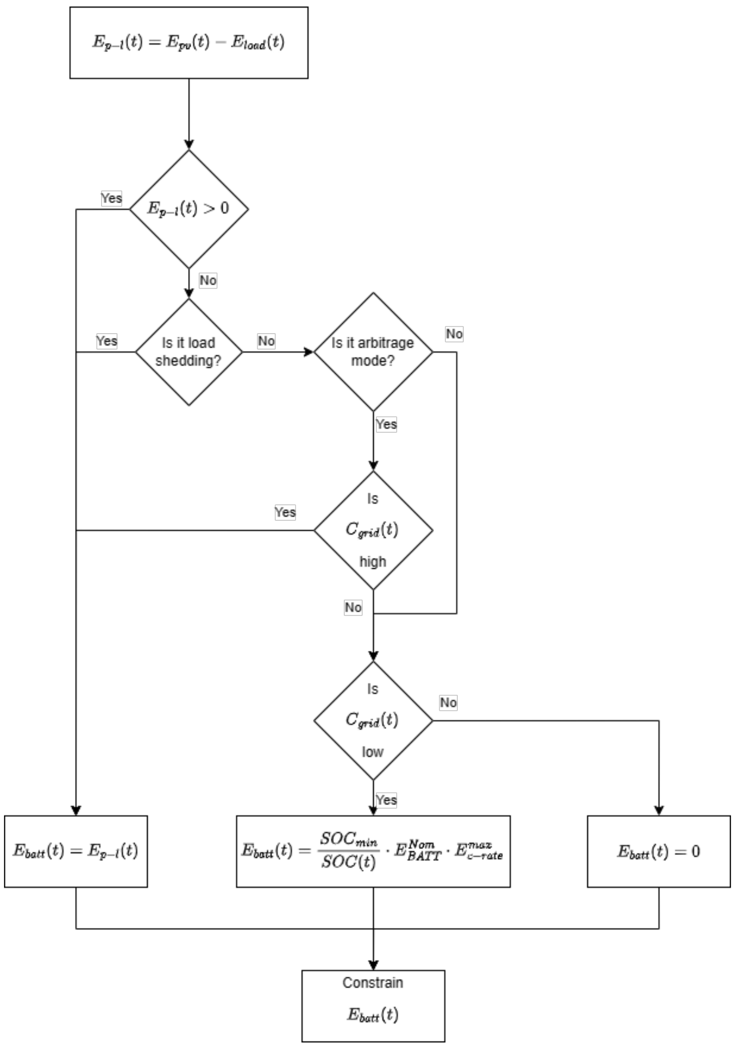


Figure 5.6: Logic for the rule-based control system.

5.3. Experimental Setup

In this section, the experiments used to guide the design of the RL agents and the environment are detailed as well as the experiments used to validate the agent's performance and the validity of the system sizing optimization.

5.3.1. Reinforcement Learning Agents

The different RL algorithms chosen to be evaluated in this report are DQN, PPO, TD3 and SAC. PPO, TD3 and SAC were chosen as these are some of the most recent techniques developed in the field of reinforcement learning. DQN was added as it is one of the simplest algorithms used in RL. This selection of algorithms also covers a wide array of different properties detailed in Table 5.2.

Learning Type Can either be off-policy or on-policy. An off-policy algorithm updates its value function using a policy different to the current policy used for exploration. On-policy algorithms use the current policy for both exploration and value function updates. For example, Q-learning updates the Q function based on greedy action selection while the policy it follows is an ϵ-greedy one. An off-policy algorithm allows for the use of experience replay which increases the sample efficiency.

Policy Type The policy that an algorithm follows can either be deterministic or stochastic. A stochastic policy is a policy that selects actions based on probabilities and implies $\pi(s,a) > 0 \forall a \in A$. While a deterministic policy does not include probabilities, e.g. $\pi(s) = \arg\max_a Q(s,a)$. Stochastic policies encourage exploration as the agent won't always take the same action in the same state.

Action Space The action space refers to the type of actions the agents can take and can be continuous $a \in (-1, 1)$ or discrete $a \in -1, -0.5, 0, 0.5, 1$.

Architecture The architecture of an algorithm describes whether the policy is directly modelled or not, i.e. actor-critic or if the policy is based on the value function, i.e. the critic.

Algorithm	Learning Type	Policy Type	Action Space	Architecture
DQN	Off-policy	Deterministic	Discrete	Critic
PPO	On-Policy	Stochastic	Discrete and Continuous	Actor-critic
TD3	Off-policy	Deterministic	Continuous	Actor-critic
SAC	Off-policy	Stochastic	Continuous	Actor-critic

Table 5.2: The different properties of the chosen algorithms to investigate.

The main parameters used to benchmark the RL agents against are the total cost for an episode and the total load that needed to be dropped due to system limitations. The cost benchmark is chosen as this is what the agents are trying to optimize for. Tracking the total load dropped ensures that the agents are not choosing to ignore the load to reduce cost. A metric that is useful for monitoring the agents' behaviour but isn't useful as a benchmark would be how often the environment had to clip the action chosen by the agent due to system constraints. This can indicate if the agent is choosing to charge and discharge the battery at the maximum amount and not learning a useful policy.

State Feature Selection

The first experiments to be run will be to determine the optimal state features to pass to the RL algorithms. The following initial state feature configurations will be tested:

1. $E_{load}(t), E_{PV}(t), SOC(t)$
2. $E_{load}(t), E_{PV}(t), SOC(t), C_{grid}(t)$
3. $E_{load}(t), E_{PV}(t), SOC(t), C_{grid}(t), loadshedding(t)$
4. $E_{load}(t), E_{PV}(t), SOC(t), C_{grid}(t), loadshedding(t), month(t)$
5. $E_{load}(t), E_{PV}(t), SOC(t), C_{grid}(t), loadshedding(t), next_{loadshedding}(t)$
6. $E_{PV}(t) - E_{load}(t), SOC(t), C_{grid}(t), loadshedding(t)$

Where $loadshedding(t)$ is either true or false depending on if there is load shedding and $next_{loadshedding}(t)$ is the number of time steps until the next load shedding period.

The performance of the model under different state features will be based mainly on the mean episode reward as this gives a good indication of how the model is learning. Once the optimal state features are found experiments will be run to determine the best range for each, i.e. should the values be normalized between 0 and 1 or between -1 and 1. In the case of the SOC, the ranges to be tested are $[SOC_{min}, 1]; [0, 1]; [-1, 1]$.

Reward Function Engineering

The reward function plays the largest role in determining an agent's behaviour. Thus, it is critical to design a reward function that produces the desired behaviour while facilitating easy training of the agent. The elements of the reward function are based on Zhou et al. [19] where penalty terms are utilized in their reward function for load imbalances and invalid battery actions. The following reward functions will be tested:

1. $r_t = -c_1 \cdot C(t)$

2. $r_t = -c_1 \cdot C(t) - c_2 \cdot load_{dropped}$
3. $r_t = -c_1 \cdot C(t)^2 - c_3 battery_{clipped}$
4. $r_t = -c_1 \cdot C(t) - c_2 \cdot load_{dropped} - c_3 battery_{clipped}$

Where $load_{dropped}(t)$ is the amount of load not satisfied and $battery_{clipped}(t)$ can either be 1 or 0 depending on if the environment was required to clip the agent's action to stay within operating limits. The reason for testing a $C(t)^2$ term in the reward function is that this will heavily penalize a large cost relative to a low cost. This may encourage the agent to smooth out the cost over time and reduce overall cost. The coefficients c_1, c_2, c_3 are there to determine the relative weighting of the different terms in the reward function. Different values for these coefficients will also be explored to get the desired agent behaviour.

Hyperparameter Tuning

Finally, once the state features and reward function are determined the hyperparameters for the models will be tuned. This will be done by testing hyperparameters on either side of the defaults for the model and then extending the search in the direction that shows better agent performance until performance starts to degrade. Table 5.3 details all the hyperparameters that will be searched over:

Algorithm	Hyperparameters
All	γ, learning rate, neural network architecture , training frequency
TD3, SAC and DQN	buffer size
TD3 and SAC	action noise type, action noise σ
DQN	ϵ_{end}, $epsilon_{fraction}$

Table 5.3: Table of all the hyperparameters that will be tuned

Model Evaluation

After the hyperparameter tuning is complete the different models will be compared against each other and the baselines describe in section 5.2. The metrics used to determine a models' performance will be the total cost over an episode and the amount of load dropped. Then the models will be applied to systems with different battery and PV sizings to what they were trained on. This will test their ability to generalized and will be evaluated against the rule based models.

5.3.2. Particle Swarm Optimization

The decision to only use PSO to find the optimal sizing was based on two key factors. Firstly the crossover technique discussed in section 3.2.3 does not work well in the given problem environment as there are only two characters in the string encoding, the battery size and the PV size. Thus, crossover will result in two members of the population just swapping system parameters. Secondly PSO requires many fewer members in comparison to GAs [56] and at each iteration for each member of the population an agent will have to be run on the whole environment. This makes PSO less computationally expensive.

In comparison to the RL models, PSO has very few parameters to optimize. The hyperparameters are c_1, c_2, ω that determine the swarm behaviour and N the number of particles in a swarm. Experiments will be run to determine the optimal hyperparameter that result in the fastest convergence. The final parameter that requires experimental investigation is the objective function. The experiments will explore the following iterations of objective functions:

1. $J = c_1 \sum_{t \in T} C(t)$
2. $J = c_1 \cdot \sum_{t \in T} C(t) + c_2 \cdot (C_{PV}^{Capital} + C_{batt}^{Capital})$
3. $J = c_1 \cdot \sum_{t \in T} C(t) + c_3 \cdot load_{dropped}^{Total}$
4. $J = c_1 \cdot \sum_{t \in T} C(t) + c_2 \cdot (C_{PV}^{Capital} + C_{batt}^{Capital}) + c_3 \cdot load_{dropped}^{Total}$

The results from these experiments will be compared against online tools used for sizing PV and battery system.

Chapter 6

Results

This chapter details the results for the experiments outlined in section 5.3.

6.1. Training the Reinforcement Learning Models

6.1.1. State Feature Selection

This section details the results of the experiments used to determine the optimal configuration of state features that will allow the RL models to learn the best policy. Figure 6.1 compares the model's training performance with and without information about load shedding encoded in the state features. The figure shows that including load shedding provides a significant boost to the model's training performance. The experiments used to evaluate other state features returned no significant results, including $C_{grid}(t)$ had no effect while including the month had a slight negative impact. Overall it was found that normalizing the state features between -1 and 1 was slightly beneficial and improved training speed by a small amount.

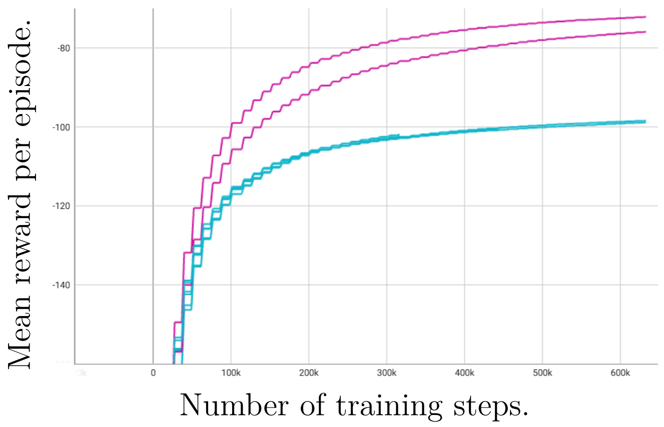


Figure 6.1: Training performance of load shedding as a state feature and no load shedding.

6.1.2. Reward Function Engineering

The results of the reward function experimentation show that the choice of reward terms greatly affects the behaviour the models learn. Figure 6.2 shows the episode cost for differing reward functions. As shown reward functions 1 and 3 defined in section 5.3 not including penalty terms for dropping load during load shedding and choosing battery actions outside the limitations of the system causes the model to exploit these to drastically reduce cost.

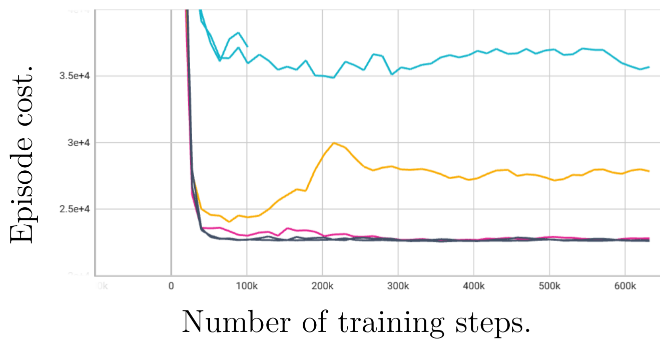


Figure 6.2: Episode cost with reward function 1, 2, 3, 4 as defined in 5.3.1

The exact coefficients for the penalty terms were determined to be $c_1 = 30$ for the dropped load penalty and $c_2 = 10$ for the system limitations violation term.

6.1.3. Hyperparameter Tuning

The experiments used to determine the optimal hyperparameters required the training of over 200 models, of which the final results are presented in table 6.1. The performance was comparable across linear, sigmoid and exponential learning rate decay functions. An interesting result from the experimentation shows that changes in γ changed the model behaviour. A higher γ results in the model reducing the load dropped while a lower γ has the model reducing the episode cost. This is likely due to a lower γ reducing the time horizon of the model, i.e. future rewards have a lesser impact on current behaviour and thus the model will select more immediately beneficial actions.

Hyperparameter	Value
Learning Rate	Exponential decay: from 1×10^{-3} to 1×10^{-4}
γ	0.7
buffer size	1×10^6
training frequency	3072
Neural network architecture	n_{in} x 64 x 64 x 1
Action noise type	Gaussian
Action noise σ	0.01
ϵ_{frac}	0.1
ϵ_{final}	0.01

Table 6.1: The final hyperparameters used to train the reinforcement learning models.

6.2. Model Evaluation

In this section first the performance of the different RL models are evaluated to determine the best model. Then the best RL model is compared against the rule-based control system across multiple system sizes to assess its ability to generalize.

Figure 6.3 shows the mean episode reward for the best-performing model of each type. Initially, the TD3 and SAC models outperform both the PPO and DQN models. This is to be expected as they are both off-policy models and are thus more sample efficient than PPO. However, after 600k -800k training steps the TD3 and SAC models plateau while the PPO model keeps learning and still doesn't reach the asymptote after 1.8M training steps. While the PPO model may not be as sample efficient as TD3 and SAC it requires way less wall clock time to perform the same number of training steps.

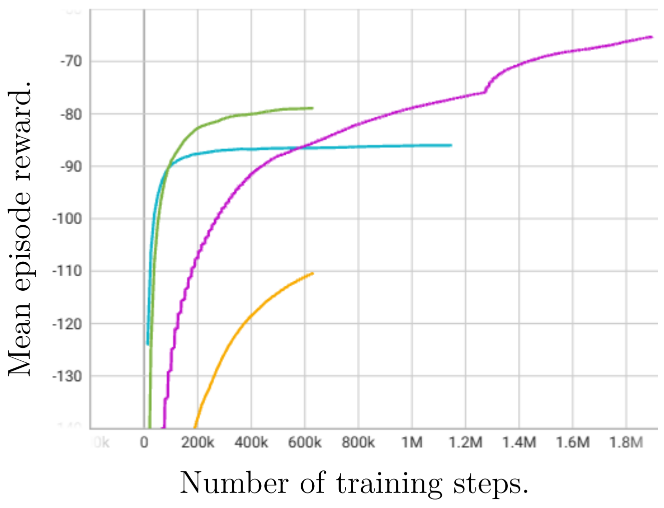


Figure 6.3: Training performance of PPO, SAC, TD3 and, DQN models

The PPO model is compared to the two rule base control methods in a test environment with a battery size of 10kWh and a PV system size of 5kWh which differs from the training environment with a battery size of 5kWh. The results shown in Table 6.2 indicate that while the PPO model can outperform the control-based method in emergency mode in terms of operating cost it does so at the cost of not being able to meet the load demand during load shedding. The control-based method in arbitrage mode outperforms both other methods in terms of operating cost and only fails to meet a small amount of load.

Control method	Episode cost	Dropped load
Proximal policy optimization	R8036.98	363.6 kWh
Emergency mode	R9998.88	0kWh
Arbitrage mode	R7611.09	5.7kWh

Table 6.2: Comparison between the Proximal Policy Optimization model and the two baseline control methods.

The RL approach to optimal scheduling fails to outperform basic rule-based control methods, however, the model can generalize and work on systems different to the training environment.

6.3. Optimal System Sizing

After the experiments to determine the optimum hyperparameters for PSO were conducted the following values were found to be optimal: $c_1 = 0.4$, $c_2 = 0.5$, $w = 0.9$. While changes in the reward function had minimal affect on the output it was decided that the following objective function would be used as it includes all the relevant terms.

$$J = \sum_{t \in T} C(t) + load^{total}_{dropped} 50 \tag{6.1}$$

Comparing the optimisation using the arbitrage rule-based control method and the PPO agent results in Table 6.3

Control method	Arbitrage	Proximal policy optimization
Episode cost	R7474	R7731
Dropped load	0kWh	322kWh
Capital cost	R127507	R84409
Battery size	10.60 kWh	5.1kWh
PV size	3.92kWh	3.486kWh

Table 6.3: Comparison between the Proximal Policy Optimization model and the arbitrage rule-based control method.

The system size that suggest optimal performance for the arbitrage rule-based method is very similar to the size suggested by an online tool [57] which suggest for a residential home with 26-30kWh average daily consumption that a solar system size of 5kW and a battery size of 10kWh is ideal for maximizing returns. The suggested system for the PPO model is undersized in comparison and results in load being dropped. However, this system has a much lower capital cost and could be a viable solution for a residential owner that is severely cash limited. The success of PSO on the rule-based methods shows that this approach may be successful and a better performing RL algorithm.

Chapter 7

Summary and Conclusion

In this project, various RL techniques were investigated to create a model capable of approximating the optimal control of a BESS without a model of the system dynamics or a method of predicting load and PV supply. The aim of this is to reduce the operating cost of a residential home's electricity. A secondary investigation was carried out to determine the impact of various control methods on the optimal sizing of a residential battery system.

First, related work in the current literature was explored. Then the background and mathematical basis for these techniques were explained and the final 4 RL techniques to be explored were selected. Following this, the design of the environment model to be used in training is covered. Once the environment was designed a series of experiments were conducted to determine the optimum parameters for the models and then the different models were compared to determine the best model to take to the final stage of experiments. These experiments showed that while the model could not outperform the basic rule-based control methods used as benchmarks it was comparable to one of them and demonstrated an ability to still perform to equivalent levels on a system with different specifications to the training environment. The system optimization works really well on the arbitrage rule-based control method, however applying the PSO to the PPO model results in an undersized system. This shows there may be promise for this method of system sizing optimization if a better RL model could be trained.

Various avenues of further research can be conducted based on this initial research. The scope can be expanded to include either a model of the system dynamics or some form of forecasting method to provide the RL model with more information to better control the BESS. This will most likely result in a reduction in generalizability as the control method becomes dependent on the system dynamics. This project was conducted in a low resource setting with only one residential load. Increasing the number of loads to train on could improve performance and/or the ability of the model to generalize to different settings.

Bibliography

[1] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996. [Online]. Available: https://www.jair.org/index.php/jair/article/view/10166/24110

[2] Eskom, "Municipal loadshedding schedules." [Online]. Available: https://www.eskom.co.za/distribution/wp-content/uploads/2021/06/Stellenbosch.pdf

[3] "Tariff book 2023/2024." [Online]. Available: https://www.eskom.co.za/distribution/wp-content/uploads/2023/04/2023_24-Tariff-Book_01042023.pdf

[4] Z. Niu, J. Wu, X. Liu, L. Huang, and P. S. Nielsen, "Understanding energy demand behaviors through spatio-temporal smart meter data analysis," *Energy*, vol. 226, p. 120493, 2021.

[5] M. Ram, M. Child, A. Aghahosseini, D. Bogdanov, A. Lohrmann, and C. Breyer, "A comparative analysis of electricity generation costs from renewable, fossil fuel and nuclear sources in g20 countries for the period 2015-2030," *Journal of cleaner production*, vol. 199, pp. 687–704, 2018.

[6] D. of Mineral Resources and Energy, "2022 south african energy sector report." [Online]. Available: https://www.energy.gov.za/files/media/explained/2022-South-African-Energy-Sector-Report.pdf

[7] ESKOM. (2021) Customer care. tariffs and charges. [Online]. Available: https://www.eskom.co.za/distribution/tariffs-and-charges/

[8] R. David, "R 230 billion eskom funding shortfall threatens job creation. press releases (p. 2). democratic alliance," 2015.

[9] R. Inglesi, "Aggregate electricity demand in south africa: Conditional forecasts to 2030," *Applied energy*, vol. 87, no. 1, pp. 197–204, 2010.

[10] A. Pandarum, G. A. Lekoloane, and D. B. Milazi, "Trends and statistics of solar pv distributed generation in south africa," 2019.

[11] C. O. Okoye and O. Solyali, "Optimal sizing of stand-alone photovoltaic systems in residential buildings," *Energy*, vol. 126, pp. 573–584, 2017.

[12] R. Luthander, A. M. Nilsson, J. Widén, and M. Åberg, "Graphical analysis of photovoltaic generation and load matching in buildings: A novel way of studying self-consumption and self-sufficiency," *Applied Energy*, vol. 250, pp. 748–759, 2019.

[13] B. Hong, W. Zhang, Y. Zhou, J. Chen, Y. Xiang, and Y. Mu, "Energy-internet-oriented microgrid energy management system architecture and its application in china," *Applied Energy*, vol. 228, pp. 2153–2164, 2018.

[14] S. Kwon, Y. Xu, and N. Gautam, "Meeting inelastic demand in systems with storage and renewable sources," *IEEE Transactions on Smart Grid*, vol. 8, no. 4, pp. 1619–1629, 2017.

[15] L. Huang, J. Walrand, and K. Ramchandran, "Optimal demand response with energy storage management," in *2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, 2012, pp. 61–66.

[16] Y. Xu, W. Gao, Y. Li, and F. Xiao, "Operational optimization for the grid-connected residential photovoltaic-battery system using model-based reinforcement learning," *Journal of Building Engineering*, vol. 73, p. 106774, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352710223009531

[17] Z. Wang, Y. Liu, Z. Ma, X. Liu, and J. Ma, "Lipsg: Lightweight privacy-preserving q-learning-based energy management for the iot-enabled smart grid," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 3935–3947, 2020.

[18] W. Kolodziejczyk, I. Zoltowska, and P. Cichosz, "Real-time energy purchase optimization for a storage-integrated photovoltaic system by deep reinforcement learning," *Control Engineering Practice*, vol. 106, p. 104598, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066120301763

[19] K. Zhou, K. Zhou, and S. Yang, "Reinforcement learning-based scheduling strategy for energy storage in microgrid," *Journal of Energy Storage*, vol. 51, p. 104379, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X22004030

[20] H. Shengren, P. P. Vergara, E. M. Salazar Duque, and P. Palensky, "Optimal energy system scheduling using a constraint-aware reinforcement learning algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 152, p. 109230, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142061523002879

[21] S. Abedi, S. W. Yoon, and S. Kwon, "Battery energy storage control using a reinforcement learning approach with cyclic time-dependent markov process,"

International Journal of Electrical Power and Energy Systems, vol. 134, p. 107368, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142061521006074

[22] D. J. Harrold, J. Cao, and Z. Fan, "Data-driven battery operation for energy arbitrage using rainbow deep reinforcement learning," *Energy*, vol. 238, p. 121958, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360544221022064

[23] T. A. Nakabi and P. Toivanen, "Deep reinforcement learning for energy management in a microgrid with flexible demand," *Sustainable Energy, Grids and Networks*, vol. 25, p. 100413, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352467720303441

[24] L. Yu, W. Xie, D. Xie, Y. Zou, D. Zhang, Z. Sun, L. Zhang, Y. Zhang, and T. Jiang, "Deep reinforcement learning for smart home energy management," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2751–2762, 2020.

[25] E. Mocanu, D. C. Mocanu, P. H. Nguyen, A. Liotta, M. E. Webber, M. Gibescu, and J. G. Slootweg, "On-line building energy optimization using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 3698–3708, 2019.

[26] Y. Du, H. Zandi, O. Kotevska, K. Kurte, J. Munk, K. Amasyali, E. Mckee, and F. Li, "Intelligent multi-zone residential hvac control strategy based on deep reinforcement learning," *Applied Energy*, vol. 281, p. 116117, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030626192031535X

[27] I. Tégani, A. Aboubou, M. Ayad, M. Becherif, R. Saadi, and O. Kraa, "Optimal sizing design and energy management of stand-alone photovoltaic/wind generator systems," *Energy Procedia*, vol. 50, pp. 163–170, 2014, technologies and Materials for Renewable Energy, Environment and Sustainability (TMREES14 – EUMISD). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1876610214007565

[28] H. Yang, W. Zhou, L. Lu, and Z. Fang, "Optimal sizing method for stand-alone hybrid solar–wind system with lpsp technology by using genetic algorithm," *Solar Energy*, vol. 82, no. 4, pp. 354–367, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0038092X07001831

[29] R. G. Allwyn, A. Al-Hinai, R. Al-Abri, and A. Malik, "Optimization and techno-economic analysis of pv/battery system for street lighting using genetic algorithm – a case study in oman," *Cleaner Engineering and Technology*, vol. 8, p. 100475, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666790822000805

[30] S. Garip and S. Ozdemir, “Optimization of pv and battery energy storage size in grid-connected microgrid,” *Applied Sciences*, vol. 12, no. 16, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/16/8247

[31] U. Mulleriyawage and W. Shen, “Optimally sizing of battery energy storage capacity by operational optimization of residential pv-battery systems: An australian household case study,” *Renewable Energy*, vol. 160, pp. 852–864, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960148120310983

[32] S. P. Koko, “Optimal battery sizing for a grid-tied solar photovoltaic system supplying a residential load: A case study under south african solar irradiance,” *Energy Reports*, vol. 8, pp. 410–418, 2022, iCPE 2021 - The 2nd International Conference on Power Engineering. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352484722004309

[33] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, “A survey of multi-objective sequential decision-making,” *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013. [Online]. Available: https://www.jair.org/index.php/jair/article/view/10836/25862

[34] M. L. Puterman, “Chapter 8 markov decision processes,” in *Stochastic Models*, ser. Handbooks in Operations Research and Management Science. Elsevier, 1990, vol. 2, pp. 331–434. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0927050705801720

[35] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html

[36] Y. Xiao, W. Sun, and L. Sun, “Dynamic programming based economic day-ahead scheduling of integrated tri-generation energy system with hybrid energy storage,” *Journal of Energy Storage*, vol. 44, p. 103395, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X21010835

[37] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015. [Online]. Available: https://doi.org/10.1038/nature14236

[38] S. Fujimoto, H. van Hoof, and D. Meger, “Addressing function approximation error in actor-critic methods,” 2018. [Online]. Available: https://arxiv.org/abs/1802.09477

[39] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in Neural Information Processing Systems*, S. Solla, T. Leen, and K. Müller, Eds., vol. 12. MIT Press, 1999. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf

[40] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 1. Bejing, China: PMLR, 22–24 Jun 2014, pp. 387–395. [Online]. Available: https://proceedings.mlr.press/v32/silver14.html

[41] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017. [Online]. Available: https://arxiv.org/abs/1707.06347

[42] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," 2016.

[43] A. E. Hassanien and E. Emary, *Swarm Intelligence.* CRC Press, Sep. 2018. [Online]. Available: https://doi.org/10.1201/9781315222455

[44] E. Commission, "PHOTOVOLTAIC GEOGRAPHICAL INFORMATION SYSTEM." [Online]. Available: https://joint-research-centre.ec.europa.eu/photovoltaic-geographical-information-system-pvgis_en

[45] T. Huld and A. M. G. Amillo, "Estimating pv module performance over large geographical regions: The role of irradiance, air temperature, wind speed and solar spectrum," *Energies*, vol. 8, no. 6, pp. 5159–5181, 2015. [Online]. Available: https://www.mdpi.com/1996-1073/8/6/5159

[46] "Energy solar battery system sizing calculator." [Online]. Available: https://alumo.co.za/solar-calculator-south-africa

[47] T. McKay and D. Hendricks, "Pitiful rooftop solar uptake in sunny south africa: A policy, funding and service delivery perspective," *Frontiers in Sustainable Cities*, vol. 4, 2022. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frsc.2022.969040

[48] N. Orth, N. Munzke, J. Weniger, C. Messner, R. Schreier, M. Mast, L. Meissner, and V. Quaschning, "Efficiency characterization of 26 residential photovoltaic battery storage systems," *Journal of Energy Storage*, vol. 65, p. 107299, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X23006965

[49] J.-O. Lee and Y.-S. Kim, "Novel battery degradation cost formulation for optimal scheduling of battery energy storage systems," *International Journal of Electrical Power and Energy Systems*, vol. 137, p. 107795, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0142061521010140

[50] E. Arksand, "Parametrization of a lithium-ion battery," Master's thesis, KTH, Chemical Engineering, 2021.

[51] A. Pandarum, "Price parity of solar pv with storage." [Online]. Available: https://www.ameu.co.za/price_parity_of_solar_pv_with_storage_-_a_pandarum.pdf

[52] [Online]. Available: https://atb.nrel.gov/electricity/2023/residential_battery_storage

[53] M. Kamran, "Chapter 7 - microgrid and hybrid energy systems," in *Fundamentals of Smart Grid Systems*, M. Kamran, Ed. Academic Press, 2023, pp. 299–363. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780323995603000065

[54] A. Bouakkaz, A. J. G. Mena, S. Haddad, and M. L. Ferrari, "Efficient energy scheduling considering cost reduction and energy saving in hybrid energy system with energy storage," *Journal of Energy Storage*, vol. 33, p. 101887, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352152X20317242

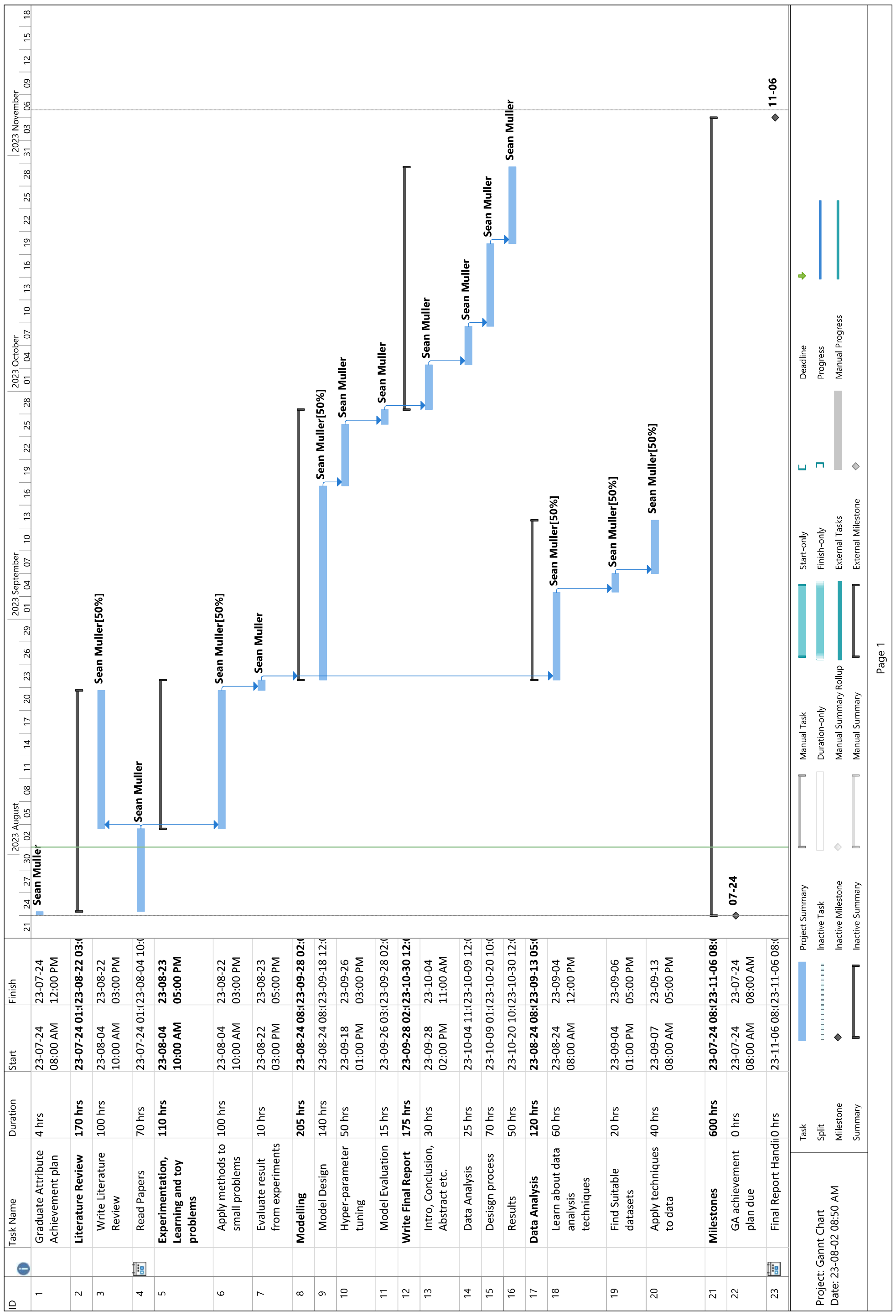
[55] H. Energy, "Its now official: South africa has now had more load shedding in 2023 to date than in 2022 combined," 2023. [Online]. Available: https://www.dailymaverick.co.za/article/2023-05-12-its-now-official-south-africa-has-now-had-more-load-shedding-in-2023-to-date-than-in-2

[56] A. G. Gad, "Particle swarm optimization algorithm and its applications: A systematic review," *Archives of Computational Methods in Engineering*, vol. 29, no. 5, pp. 2531–2561, Apr. 2022. [Online]. Available: https://doi.org/10.1007/s11831-021-09694-4

[57] "Residential solar battery system size calculator." [Online]. Available: https://www.solarchoice.net.au/learn/calculators/residential-solar-battery-system-size-calculator/

Appendix A

Project Planning Schedule



ID		Task Name	Duration	Start	Finish
1		Graduate Attribute Achievement plan	4 hrs	23-07-24 08:00 AM	23-07-24 12:00 PM
2		**Literature Review**	**170 hrs**	**23-07-24 01:(**	**23-08-22 03:(**
3		Write Literature Review	100 hrs	23-08-04 10:00 AM	23-08-22 03:00 PM
4		Read Papers	70 hrs	23-07-24 01:(	23-08-04 10:(
5		**Experimentation, Learning and toy problems**	**110 hrs**	**23-08-04 10:00 AM**	**23-08-23 05:00 PM**
6		Apply methods to small problems	100 hrs	23-08-04 10:00 AM	23-08-22 03:00 PM
7		Evaluate result from experiments	10 hrs	23-08-22 03:00 PM	23-08-23 05:00 PM
8		**Modelling**	**205 hrs**	**23-08-24 08:(**	**23-09-28 02:(**
9		Model Design	140 hrs	23-08-24 08:(	23-09-18 12:(
10		Hyper-parameter tuning	50 hrs	23-09-18 01:00 PM	23-09-26 03:00 PM
11		Model Evaluation	15 hrs	23-09-26 03:(	23-09-28 02:(
12		**Write Final Report**	**175 hrs**	**23-09-28 02:(**	**23-10-30 12:(**
13		Intro, Conclusion, Abstract etc.	30 hrs	23-09-28 02:00 PM	23-10-04 11:00 AM
14		Data Analysis	25 hrs	23-10-04 11:(	23-10-09 12:(
15		Desisgn process	70 hrs	23-10-09 01:(	23-10-20 10:(
16		Results	50 hrs	23-10-20 10:(	23-10-30 12:(
17		**Data Analysis**	**120 hrs**	**23-08-24 08:(**	**23-09-13 05:(**
18		Learn about data analysis techniques	60 hrs	23-08-24 08:00 AM	23-09-04 12:00 PM
19		Find Suitable datasets	20 hrs	23-09-04 01:00 PM	23-09-06 05:00 PM
20		Apply techniques to data	40 hrs	23-09-07 08:00 AM	23-09-13 05:00 PM
21		**Milestones**	**600 hrs**	**23-07-24 08:(**	**23-11-06 08:(**
22		GA achievement plan due	0 hrs	23-07-24 08:00 AM	23-07-24 08:00 AM
23		Final Report Handi	0 hrs	23-11-06 08:(	23-11-06 08:(

Project: Gannt Chart
Date: 23-08-02 08:50 AM

Page 1

Appendix B

Outcomes Compliance

This appendix details the contents of the report which demonstrate compliance with the ECSA learning outcomes.

ELO 1. Problem solving: The objectives and scope of this project are first specified in chapter 1. Existing techniques used to solve similar problems are identified and explored in chapters 2 and 3. The problem was then modelled in section 5.1. Achieving the specified objectives required the use of techniques and optimization methods well outside the scope of engineering standards and codes.

ELO 2. Application of scientific and engineering knowledge: The application of knowledge across various fields such as data analysis and machine learning played a key role in designing suitable algorithms and experimental scenarios shown in chapter 5. This knowledge was outlined in chapters 2 and 3.

ELO 3. Engineering design: Engineering design is used throughout this project. Iterative development of the system model was conducted in section 5.1. Various solutions to the defined problem were tested in section 6.1, they were then compared in section 6.2 to determine the best solution.

ELO 4. Investigations, experiments and data analysis: An investigation into the existing techniques in literature was conducted in chapter 2. Data analysis was conducted on the data to be used to train the RL models in chapter 4. Various experiments were defined in section 5.3 to determine the best parameters of the RL models, environment and the PSO algorithm. The results of which are discussed and analyzed in chapter 6.

ELO 5. Engineering methods skills and tools, including Information Technology: The key engineering tool that was utilized throughout this project was Python and connected packages used for reinforcement learning shown in chapter 5 and 6. Further Python tools were used to analyze the data as shown in chapter 4.

ELO 6. Professional and technical communication: This outcome is shown to be met by this report which communicates the entirety of the work conducted and the video presentation which contains only the most pertinent information in a simple and communicable manner.

ELO 8. Individual work: All work conducted in the course of this project, from the research to the design and experimentation and ultimately the documentation is done individually with weekly advice from my supervisor.

ELO 9. Independent learning ability: Various unfamiliar concepts and practices were learned over the course of this project through reading research articles and text-books on RL and related fields. These concepts were adapted and applied during the implementation and design of this project. This is most obviously demonstrated in the literature review of chapter 2 and the explanation of methodology in chapter 3.

Appendix C

Experimental Results

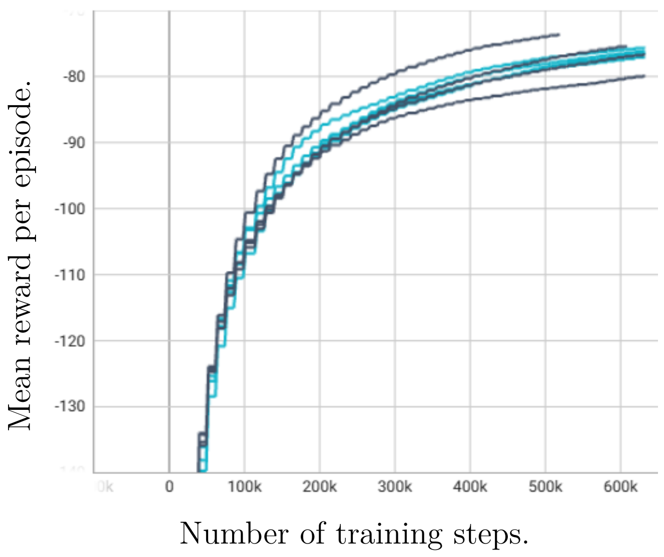


Figure C.1: Training performance of month as a state feature and no month.

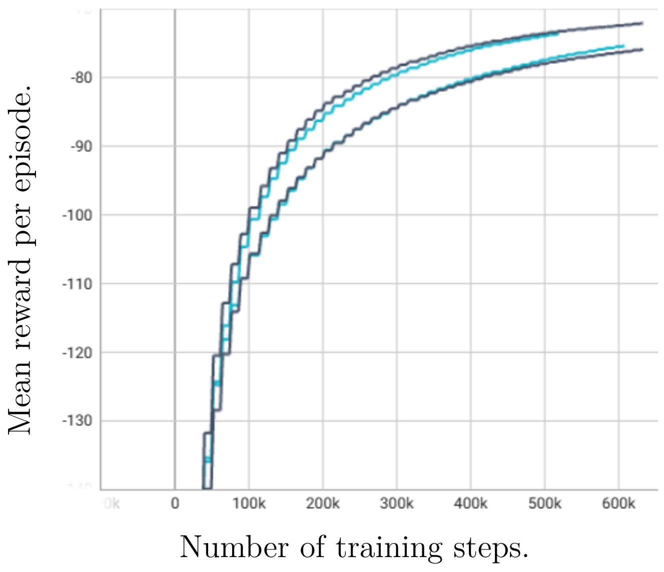


Figure C.2: Training performance of $C_{grid}(t)$ as a state feature and no $C_{grid}(t)$.

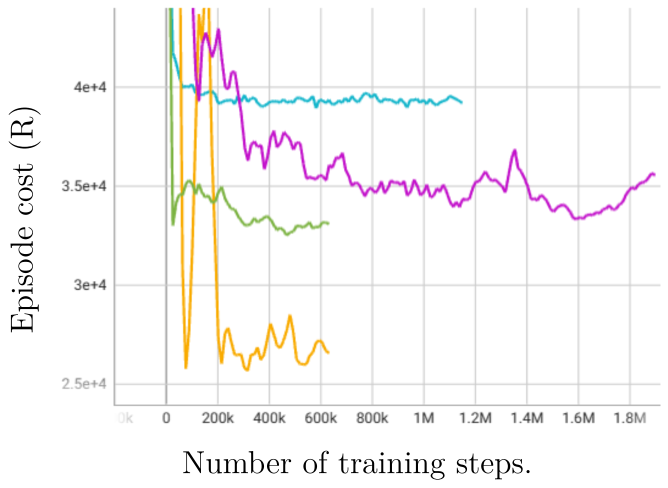


Figure C.3: Episode cost of PPO, SAC, TD3 and, DQN models.

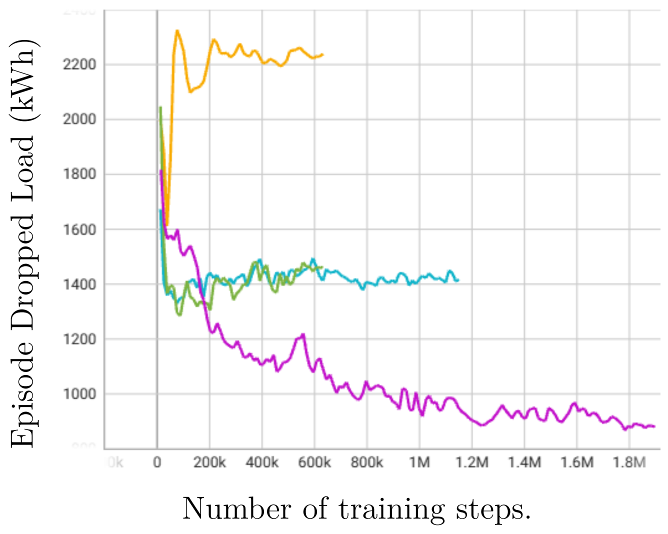


Figure C.4: Dropped load of PPO, SAC, TD3 and, DQN models.