University of BRISTOL

Wireless Power Control Optimisation

Wenbo Duan

April 2022

Final year project thesis submitted in support of the degree of

Bachelor of Engineering in Electrical and Electronic Engineering

Department of Electrical & Electronic Engineering

University of Bristol

DECLARATION AND DISCLAIMER

Unless otherwise acknowledged, the content of this thesis is the original work of the author. None of the work in this thesis has been submitted by the author in support of an application for another degree or qualification at this or any other university or institute of learning.

The views in this document are those of the author and do not in any way represent those of the University.

The author confirms that the printed copy and electronic version of this thesis are identical.

Signed:

Dated:

29/04/2022

Abstract

Septrum resources are critical in today's information age. It is foreseeable that the limited spectrum will become increasingly congested and rare in the future as the amount of wireless equipment increases. In this thesis, we examined an optimization problem centred on this issue, namely the sum-rate maximisation problem, which seeks to maximise channel capacity at a given frequency. The sum-rate maximisation is referred to as an NP-hard problem due to the non-convex nature of the optimization objective. While given the advancement of computational hardware and machine learning in the recent decay, we proposed two new ideas towards the sum-rate maximisation problem.

We first proposed the algorithm: Gradient Descent with Random Restarts (GDR). By introducing the concept of searching boundary constraints and searching result ranking, we customised the conventional gradient descent ideas for our problem setting. We examined the GDR algorithm in the presence of a Single-in-Single-Out (SISO) interference channel, examining its effectiveness, robustness, and scalability. By analysing the output performance during the experiment and comparing it to existing algorithms, we conclude that for simple channel configurations where the SISO channel of the transceiver pairs is less than 4, the GDR algorithm achieved the same optimal result as the baseline algorithm WMMES and that the output performance is stable in the presence of external Sigal-to-Noise-Ratio (SNR) variation. While the result searched by GDR degrades significantly as the complexity of the solution space increases, this indicates that GDR's scalability is currently limited.

According to the characteristics of Markov Decision Process (MDP) and our problem settings, we also established a new Reinforcement Learning (RL) training environment based on the open-source tool OpenAI GYM. Based on the customized environment, we applied the Vanilla Policy Gradient (VPG), a classic on-policy algorithm, in our optimisation task. The convergence of our framework is observed during the training stage, while the final performance is limited in the validation stage, about 40% less than WMMES. Through the subsequent experiment on different channel configuration, we concluded that structure design of neural networks in VPG and the MDP design in our training environment are both the important factors that affects the final performance.

TABLE OF CONTENTS

List of Tables

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

Transmit power control has a substantial impact on the network's traffic carrying capacity, which is becoming increasingly critical given today's scarcity of spectrum resources. Despite much research over the last decade, this optimization problem remains difficult due to the lack of concavity. On the other hand, Given today's advancement of computational hardware the gradient-based optimisations has attracted more attention. In this chapter, we will begin by introducing the context and importance of limited power control optimization problem, evaluating the existing difficulties, explain the motivation targetting the problem and end with the introduction of thesis objective, contribution and layout.

1.1 Background

1.1.1 Spectrum Scarcity

Radio communication has become the cornerstone in the information age. One problem that consistently exists in the wireless communication domain is the contradiction between the rapidly increasing demand for wireless network usage and the scarcity of radio frequency spectrum. From the social prospective, as UK spectrum strategy white-paper point out [5], the cumulative effect of spectrum is significant for increased economic and social benefit ; from the technical prospective, each band of frequency from about 8.3 KHz to 275 GHz are important that holds unique usage purposes in wireless communication. For example, the low-frequency radio transmission can retain signal strength over longer distance which is ideal for ship radio, and the high frequency radiation can transmit more data which has been widely licensed to the satellite communication.

As a scarce resource constrained by Radio Regulation (RR) and local governments, the radio frequency spectrum is extremely congested and spectrum licencing is expensive. Therefore,

radio resource management (RRM) of scarce resources such as transmit power, frequency bands, antenna patterns, and even base stations is critical to maximising the assigned frequency's utilisation efficiency. RRM is a critical but broad topic that varies according to the intended use. Among schemes for various optimization objectives such as revenue, stability, and power consumption [40] [29] [13], this thesis will focus on designing judicious schemes that maximise channel performance in terms of throughput and capacity, which is one of the most critical goals in light of spectrum scarcity.

1.1.2 Interference Management

Multiple Access [28], which operates collections of transmitter-receiver pairs concurrently in a shared medium, is one way to increase channel capacity. It has already become a fundamental design principle in today's wireless communication systems, including cellular networks and ad hoc wireless networks [20]. However, given the broadcast medium's inherent limitations, one bottleneck that should be highlighted in these radio systems is the *interference* that occurs when multiple users are served concurrently. Due to the fact that the radio channel is fundamentally a broadcast communication medium, signals transmitted by one user may be received by all other users within the transmitter's range.

In the presence of interference, communication is analysed using the concept interference channel (IC). Fig.1.1 shows a schematic of a basic IC model - scalar interference model. It shows K independent transmitters sending separate information to K receivers using the same channel, resulting K(K-1) interference links. For any receiver R_{Xk}, the received signal is given by:

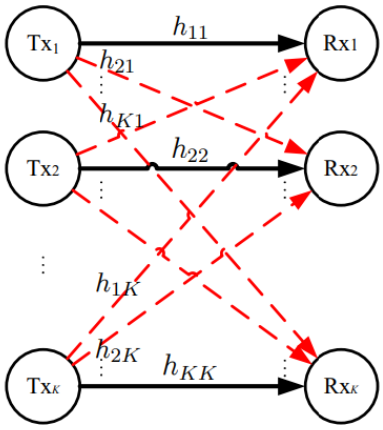


Figure 1.1: Scalar interference channel mode. The solid lines represent the direct channels, while the dotted lines represent the interfering channels.

$$y_k = h_{kk}x_k + \sum_{j \neq k} h_{kj}x_j + n_k \tag{1.1}$$

Where h_{kk} is the channel of the desired link, h_{kj} is the channel of the interference link,x_k and x_j are the desired and interference transmitted message vectors respectively and n_k is the Additive Gaussian White Noise (AWGN). As illustrated in Equation 1.1, a particular receiver is likely to encounter significant interference from neighbouring transmitters in the absence of coordination. Thus, the challenge is to devise a method for allocating resources at each transmitter in such a way that interference is avoided wherever possible.

Indeed, this resource is called power. It is also a fundamental concept in wireless communication systems; because received power is signal strength to the desired receiver and interference to all other receivers, it is ultimately power, in the form of interference, that limits system capacity. Thus, power control's objective is not only to maintain the desired link quality, but also to minimise interference with others.

1.1.2.1 Treating Interference as Noise

As a significant factor affecting channel capacity, interference is typically dealt with in one of two ways: avoidance or ignore [8]. Avoidance of interference is typically accomplished by orthogonalizing the transmit resource (time / frequency), which enables multiple users to share the wireless medium. The disadvantage of these strategies is that they waste resources and thus do not maximise capacity.

On the other hand, Treating Interference as Noise (TIN) is an ignore strategy that is optimal when the interfering signal's power is relatively low in comparison to the desired signal link's power. By treating multi-user interference as additive noise, it has been rigorously demonstrated that a certain level of interference can be tolerated without impairing transmission reliability [9]. More importantly, the authors of [30] demonstrate that, under certain conditions, treating interference as noise in a Gaussian IC actually results in the optimal sum-rate channel capacity. These conclusions provide a theoretical justification for considering channel capacity when applying this simplification (TIN).

As a result of the simplification of TIN, the analysis toward our research objective - channel capacity - can be expressed using the Shannon formula, where, for example, a single channel capacity in the aforementioned scalar interference channel model can be expressed as:

$$R_k = \log_2(1 + SINR_k) \tag{1.2}$$

Where R_k represents the informatics-theory capacity of the Kth channel, and $SINR$ refers to signal-to-noise-plus-interference ratio.

1.1.3 Non-convex Optimisation

On the basis of the solution to equation 1.2, a complete relationship between total channel capacity and transmitter power can be deduced. The network's capacity can be quantified as

the sum of the rates of each channel Rk, with the transmitter power being the primary factor affecting the results. Where our initial objective of increasing channel capacity can be specified as allocating various power values across the network in order to maximise the sum capacity rate, namely:

$$\arg \max_{\gamma_1,\cdots,\gamma_K} \sum_{i=1}^{K} R_k \tag{1.3}$$

Despite the fact that TIN simplifies the analysis of channel capacity, the introduction of the $SINR$ increases the complexity of our objective function. Fig.1.2 may be interpreted intuitively. We drew the solution space for a random 2x2 IC channel, where the x and y axis represent the power allocation ratios for the various transmitters, and the z axis represents the sum of rates. Thus, our objective of determining the optimal power allocation scheme in order to maximise total channel capacity can be rephrased as: determining the pair of (x, y) that maps to the highest point on the curved surface.

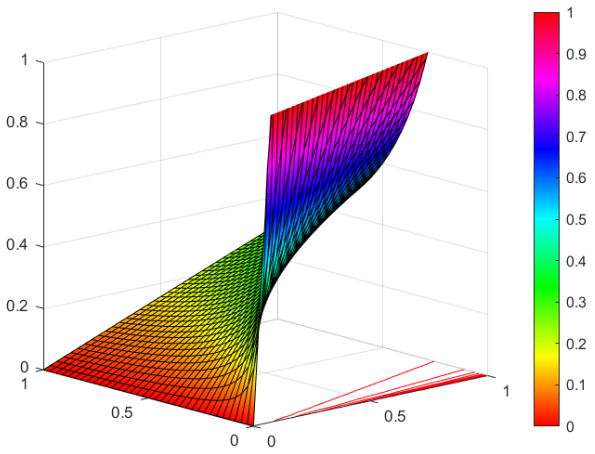


Figure 1.2: The solution space of a 2x2 scalar IC channel. Different color represents different sum-rate that can be achieved at current position. The non-convexity of solution space shows that finding the convergence point at this range is hard.

However due to lack of convergence on the curved surface, it is hard to find an optima point. In fact, this optimisation goal has been proofed as NP-hard, [19] [11] [18] which is one of the most challenging problem classes in the computational complexity theory that doesn't exist a critical solution [15]. On the other hand, it is the non-convexity and uncertainty inherent in this optimization that has resulted in a variety of algorithms being proposed today, ranging from the best performance to the fastest iteration speed. And this is also the start of our research.

1.2 Motivation

The significant practical benefits of our objective and the complexity of the problem have resulted in a diversified of research over the last few decades. While some previous algorithms, such as WMMSE [32], have demonstrated excellent performance in terms of maximising system utility, their implementation in real systems faces numerous serious obstacles. One of the most difficult is the high computational cost in real time. According to a performance investigation, the solution time for interior-point algorithms can be prohibitively long in comparison to the channel coherence time [2], which requires a massive amount of computational resources for online power allocation.

Interestingly, computational hardware has advanced dramatically over the last decade. According to a study conducted by MIT [35] on 1,058 computer science research papers from the arXiv pre-print repository, the cutting-edge computational power used in those papers increased by approximately 10% per year from 2012 to 2019. Additionally, an increasing amount of computation hardware is being customised and enhanced for parallel computing, such as NIVIDIA CUDA and Google's TPU, which are extremely efficient for matrix, gradient, and large scale variable computation [1]. These advancement in the recent 10 years greatly accelerate the gradient-based algorithm and inspired us to customize an gradient based optimisation with the help of hardware advantages.

Additionally, aside from numerical optimization, deep learning has become a focal point of research in recent years. However, within the community of deep learners, Supervised Learning remains the dominant framework for optimising the sum-rate maximisation. Typically, in our task settings, the supervised learning model will consider a given resource optimization algorithm to be a "black box" and will attempt to learn its input/output relationship using a deep neural network (DNN) and the input data. Thus, due to the nature of supervised learning, the performance of the training model is heavily dependent on the ground-truth labelling data, implying that the aforementioned numerical algorithm is still used in these methods, reintroducing the computationally expensive drawbacks.

This disadvantage motivated us to explore deep reinforcement learning (RL), a subset of machine learning that eliminates the need for external data. Rather than that, reinforcement learning discovers the optimal strategy through the interaction of a decision bot with a simulation environment. As a result of our previous experience with reinforcement learning, we were inspired to create a training environment for implementing reinforcement learning algorithms for this problem.

1.3 Thesis Objective

The purpose of this research is to investigate new solutions around the topic of Sum-rate Maximum optimisation. Given the complexity of the non-convexity , the demands for outperforming

the existing baselines is hard, while we value the process of algorithm design. Apart from finishing the final design of the two original ideas towards the sum-rate maximization problem, this research is expected to achieved the following objects in the end:

1. Successfully implementing the designed algorithm in a simulation environment.

   We hope to develop a gradient-based algorithm that is optimised for the sum-rate maximisation objective. The entire process, from algorithm design to experiment execution and result selection need expected to be meticulously documented.

2. Well designing a reinforcement learning training environment.

   We hope to thoroughly understand the Markov Decision Process (MDP) and then design the formulation of each MDP element. All of the design elements can be well defined and coded to create a simulation environment for reinforcement learning training. The final environment is executable and compatible with existing reinforcement learning training algorithms.

3. Clearly indicating the advantage and disadvantage of the proposed solutions

   The distinction between different baseline algorithms is foreseeable. We hope to draw informative comparisons from the experiment data and subjectively assess our design's performance.

1.4 Contribution

We carried two new algorithms to solve the interference channel's sum-rate maximisation problem, taking into account the universality of gradient descent and the computational efficiency of deep learning. Both algorithms are tested in a single-in-single-out (SISO):

- **Gradient Descent with Random Restarts**

  We devised a new gradient descent algorithm. We introduced result ranking methods with randomly initialised starting points multiple over a fixed number of iterations to overcome the local optima power control ratio. Additionally, we introduced the sigmoid function in order to obtain boundary constraints.

- **Reinforcement Learning Training Environment under SISO channel**

  We designed a simulation environment for the training of reinforcement learning algorithm under SISO channel. Among the Markov Decision Process, we proposed the idea to simply the action space, flattening the action space from $[0,1]^k$ to $[0,1]$ through the full utilization of computation resources. We established the environment by using OpenAI Gym and open sourced it in GitHub.

1.5 Structure Review

We introduced spectrum scarcity and discussed why the relative sum-rate maximisation problem is difficult to solve in this chapter. The motivation and thesis objective were discussed in light of the analysis of existing methods' bottlenecks. This section summarises the work contribution as well.

Chapter 2 examined the relative solutions to the resource allocation problem. We classified previous research into two broad categories and organised the existing literature into well-defined sub sections.

Chapter 3 began with a formal definition of our research model and objective, followed by the presentation of our first proposed algorithm. In terms of the algorithm, we begin by introducing the design concept, presenting the pseudocode, and conducting a series of experiments to determine the algorithm's performance. Finally, the experiment's and analysis' outputs were presented, followed by reflections on the first algorithm's design.

In Chapter 4, we discussed the fundamentals of Reinforcement Learning and then proposed our Markov Decision Process design. Following that, we introduced the reinforcement learning algorithm Vanilla Policy Gradient to our environment and used it to conduct a series of experiments to validate the proposed reinforcement learning framework's performance. The following sections contain the results and analysis, while the final section contains the conclusion and reflection on our RL environment design.

In Chapter 5, we summarized the experiment results and key conclusions from both Chapter 2 and Chapter 3.

CHAPTER 2

LITERATURE REVIEW

Starting from the wireless indoor networks optimisation [23], power control in wireless networks has been systematically studied since the 1980s. Over the tremendous growth of wireless networks like cellular network and ad hoc network in the past 40 years, the research in this area can be divided into two main threads. The first target of wireless power controlling is to achieve fixed signal to interference-plus-noise ratio (SINR) targets with minimum transmission power, in order to improve user-perceived Quality of Service (QoS) [39]; and the other target is concerned with joint SINR allocation and power control [27].

The biggest difficulty of the second thread is non-convexity due to complicated interference coupling between links [3]. Thus, practical issues in this thread like Weighted Sum-rate Maximisation (WSM) has been focused on finding a high quality sub-optimal solution efficiently with a variety of algorithms. Here, we reviewed the previous researches and categorized the mainstream algorithms for this problem as follows:

2.1 Water-filling Method

Water-filling is a classical algorithm in wireless power allocation. The process of water filling is similar to pouring the water in the vessel. By setting a horizontal lines to all sub-channels, the water-filling algorithms is to allocate more power to channel with better SNR ratio and vice versa.

In 2002, the author in [38] proposed the method which lets this Nash equilibrium can also be reached by an iterative water-filling procedure under the sufficient conditions in [38], where each transmitter successively optimizes its power spectrum while regarding other users' interference as noise. In [24], the author presented the water-filling power allocation policy under a total power constraint and highlighted some ideas for low-complexity solutions. In a wider scenario of

transmission configuration, the water-filling typed algorithm was also proposed in OFDM-based cognitive radio in [26] where the author shows that water-filling algorithms are always performed iteratively to solve the power allocation problem.

2.2 Successive Convex Approximation

Another group of ideas to approximate the non-convex sum-rate maximisation problem is called successive convex approximation (SCA). As illustrated in the Fig. 2.1 [12], each semicircle represents an convex approximation each time. The idea is to construct and maximize a series of (concave) lower bounds of the original WSRM problem, so that a high quality solution can be obtained asymptotically. In [25], an algorithm called Successive Convex Approximation for Low complExity (SCALE) is proposed to improve the spectral efficiency of the DSL network. This algorithm transforms the noncocave sum rate maximization problem into a series of convex problems by utilizing the following lower bound.

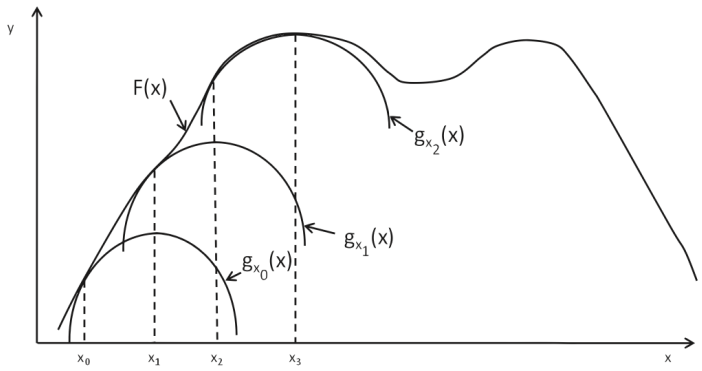


Figure 2.1: Graphical illustration of successive convex approximation. Different semicircle represents a single convex approximation

In [37], a different lower bound is proposed for the WSRM problem. Specifically, the authors decompose the objective function as the difference of two concave functions. Similar to the steps of SCA introduced earlier, in each iteration of the algorithm, the second sum is replaced with its linear lower bound, and the resulting concave maximization problem is solved.

2.3 Weighted Minimum Mean Square Error Method

Another group of ideas that target the weighted sum-ratio maximization objective is called weighted sum MSE minimization, where the weighted sum-rate maximization problem is transformed to an equivalent weighted sum MSE minimization (WMMSE) problem with some specially chosen weight matrices that depend on the optimal beamforming matrices. Since the weight

matrices are generally unknown, the authors of [3] proposed an iterative algorithm that adaptively chooses the weight matrices and updates the linear transmit–receive beamformers at each iteration. And thanks to the extension work of [32], the WMMES algorithm has low per-iteration complexity and is guaranteed to converge to a stationary point of the original sum-utility maximization problem, and it now becomes the most popular optimisation algorithm for the sum-rate maximisation objective.

2.4 Deep Learning Method

Apart from the traditional methods in non-convex optimisation, the advancement of deep neural network brings out a vast of new approximation algorithms in the recent decade.

Deep neural network is an artificial neural work with multiple hidden layers between the input and the output layers [10] Therefore, a DNN models high-level abstractions in data through multiple nonlinear transformations to learn multiple levels of representation and abstraction.

In the early stage in 2017, the author in [33] proposed the idea to treat the given resource optimisation algorithm as a "black box", and use a non-linear feed-forward multi-stage network to approximate the solutions for sparse optimization. According to it's compassion with WMMES algorithm, the author believed that Deep Nuerual Networks have great potential for real-time wireless resource allocation problems, but also admit that his current work only represents a very preliminary step towards understanding the capability of DNN. To the best of knowledge, this is the first time that DNN is applied on the sum-rate maximisation problem.

For the early work of deep learning to the sum-rate maximization problem, the author in [21] summarized as in the early work of DNN based methods, ground-truth labels for power allocation were created using baseline algorithms that had been shown to work well across a variety of scenarios, but exhibited lower complexity than solving the original problems. What's more, a comprehensive explanation for this performance distance was given in [31], where the author explained as this is because MLP and CNN fail to exploit the underlying topology of wireless networks. To enable more efficient learning, spatial convolution and graph embedding have been proposed to exploit the Euclidean geometry of the users' geolocations.

In the next few years, more fresh deep learning frameworks come out successively, where the algorithm without need of labelled data attracts more and more attention, like unsupervised learning [31] [6], [7]. To the latest of 2021, the author in [21] illustrated a way of injecting Contrastive Self-supervised Learning in the optimisation problem, where not only manually labelling data sets were not required, but also the framework can generate self-training labels from the channel matrix. This work presented a robust approximation to the problem.

CHAPTER 3

GRADIENT DESCENT WITH RANDOM RESTARTS

In this chapter, we will propose and investigate the performance of our first method, Gradient Descent with Random Restarts. The first section will define the problem, followed by a formal specification of the mathematical issue. Sections 2 and 3 will discuss the concept of algorithm design, including the implementation pseudocode. Sections 3 and 4 will demonstrate the algorithm's performance against four metrics, analyse its benefits and drawbacks, and give ideas for further refinement.

3.1 Problem Modeling

In the following experiment we consider an N users Single-In-Single-Out (SISO) scalar interference channel, where each transmitter / receiver is equipped with one antenna.

As seen in Fig.3.1, the network consists of N nodes, each of which comprises of a transmitter (T_x) and a receiver (R_x). A node's T_x transmits information to its associated R_x. We suppose that all nodes share the same frequency band, and hence that the information of one node (T_{xi}) interferes with the information of other pairs' R_{xj},$(i \neq j)$. Because the channel has no memory, an independent realisation of the channel matrix is drawn for each use. Thus, the input channel matrix $H^{N\times N}$ can be denoted as

$$H := \begin{bmatrix} |h_{11}|^2 & \cdots & |h_{1N}|^2 \\ \vdots & \ddots & \vdots \\ |h_{N1}|^2 & \cdots & |h_{NN}|^2 \end{bmatrix} \tag{3.1}$$

Regarding each channel link, h_{ij} is an independent and identically distributed (i.i.d) random variable. In a rich scattering environment the i.i.d variables follow the Complex Circular Gaussian Distribution: $h_{ij} \sim \mathbf{CN}(0, \sigma_h^2)$ and thus the channel gain $|h_{ij}|$ follows Rayleigh Distribution. Let

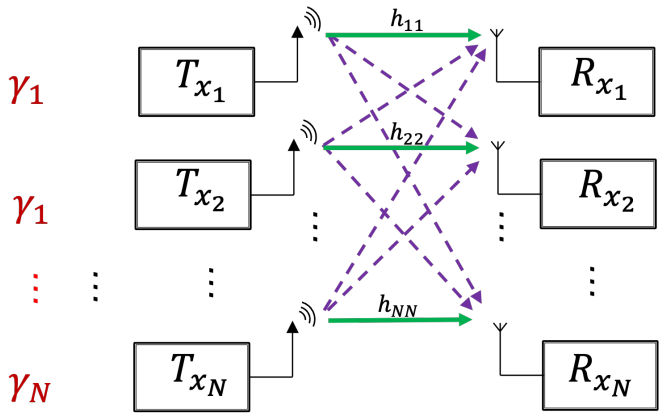


Figure 3.1: The schematic diagram of an SISO interference channel with N×N nodes. All transmitter signals are transmitted concurrently. The green lines indicate the targeted channels, whereas the purple lines indicate interference channels. The transmission power ratio γ determines the strength of each transmitter's signal.

x_i denote the signal transmitted by transmitter T_{x_i}, subject to a transmit power constraint $P_{\max}$. In general, the received signal at receiver R_{x_i} is composed of following components 1): the desired signal 2): the interference from other transmitters and 3): background noise, where:

$$y_i = h_{ii}x_i + \sum_{j \neq i} h_{ij}x_j + n_i \tag{3.2}$$

In which n_i is the Gaussian additive white noise generated by the transmission environment and is a random variable that follows the Normal Distribution $n_i \sim \mathbf{CN}(0, \sigma^2)$.

As discussed in section 1.1.2, we treat the inferences as noise. The signal-to-noise-plus-interference ratio (SINR) between transmitter T_{x_i} and receiver R_{x_i}, which is thus a critical factor for approximating the channel capacity, can be expressed as follows:

$$SINR_i = \frac{|h_{ii}|^2 p_i}{\sum_{j \neq i} |h_{ij}|^2 p_j + \sigma^2} \tag{3.3}$$

Based on the expression of $SINR$, the Shannon capacity of the channel between transmitter T_{x_i} and receiver R_{x_i} is:

$$R_i = \log_2(1 + \frac{|h_{ii}|^2 p_i}{\sum_{j \neq i} |h_{ij}|^2 p_j + \sigma^2}) = \log_2(1 + \frac{|h_{ii}|^2 \gamma_i}{\sum_{j \neq i} |h_{ij}|^2 \gamma_j + \frac{\sigma^2}{P_{\max}}}) \tag{3.4}$$

Here, we divide each transmission power by the max transmission power constrain $P_{\max}$, simplifying the objective as finding the equivalent power control ratio between $[0, 1]$.

We are interested in the sum-rate maximisation problem in this thesis, as discussed in Chapter 1. The maximisation problem, in particular, can be expressed as the following optimisation problem:

$$\begin{aligned} &\max_{\gamma_1,\cdots,\gamma_N} \sum_{i=1}^{N} R_i \\ &s.t. \quad R_i = \log_2(1+\frac{|h_{ii}|^2\gamma_i}{\sum_{j\neq i}|h_{ij}|^2\gamma_j+\frac{\sigma^2}{P_{\max}}}), \quad \forall i \\ &\qquad \gamma_i \in [0,1], \qquad\qquad\qquad\qquad\quad \forall i \end{aligned}$$

And the ultimate goal is to find the power control ratio for each transmitter, outputting a $N \times 1$ control vector Γ, where:

$$\Gamma = \left[\gamma_1 \cdots \gamma_N\right]^T \tag{3.5}$$

3.2 Method Introduction

3.2.1 Stochastic Gradient Descent

Gradient descent (GD) is a widely used iterative method for optimising *convex* functions. Beginning at a random point in the solution space, the gradient operation indicates the direction in which the current position is the steepest. Thus, by updating the current position iteratively in accordance with the gradient values and stepping distance, we can reach a critical convergence point for the final solution. Additionally, even in a high-dimensional solution space, such as a system with a large number of transmitters, gradient descent remains an efficient optimization method due to the stochastic approximation (SGD), which randomly selects one data point (R_i) from the entire data set during each iteration to significantly reduce computations.

In a formal term, defining the optimisation objective function as $J(\theta)$, where θ is the collection of all independent variables (the solution to the problem) and α is the updating step length of each iteration, the updating rules of gradient descent can be expressed as:

$$\begin{aligned} \theta_i &:= \theta_i - \alpha \frac{\partial}{\partial \theta_i} J(\theta_i) \\ &(for\ i = 1,2,\cdots,N) \end{aligned} \tag{3.6}$$

3.2.2 Conditional Constrains

The first issue with the conventional SGD algorithm is that the updating routes is completely unconstrained, which may result in solutions travelling in all directions and thus exceeding the boundary of the desired solution space.

Our solution to the conditional constraints in the conventional gradient descent method is to apply the *sigmoid* function to all independent variables, which is inspired by the concept of activation functions in the Artificial Neural Network (ANN) structure. Numerous important properties of activation functions include nonlinearity, differentiability, continuity, boundedness,

and zero-centering [4]. Here, we've inserted the Sigmoid function into our objective function (as illustrated in Fig. 3.2). We deemed this solution to be valuable for the following reasons:

- Due to the nonlinearity of the sigmoid function, we can map our entire set of independent variables into the range $(0,1)$ without affecting the original equation's convergence. Additionally, the range is identical to the range of the predefined power control ratio.
- Because the sigmoid function remains differentiable, it can be directly integrated into the objective function $J(\theta)$ without impairing its optimality. Additionally, it should be robust in a broader communication scenario, such as Channel Distortion, as long as the objective function is differentiable.

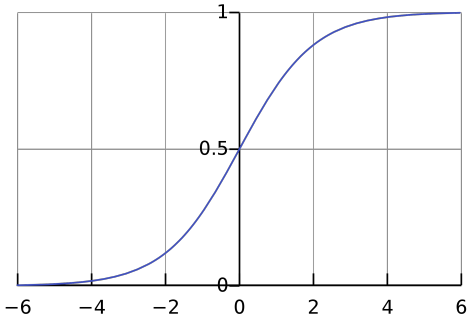


Figure 3.2: Function with a sigmoid shape. By introducing the Sigmoid function, we can effectively constrain our solution space due to its nonlinearity, differentiability, continuous, bounded, and zero-centering properties [4].

By incorporating the Sigmoid function into our objective function, the gradient descent updating rule can be improved as follows:

$$\gamma_i := \gamma_i - \gamma \frac{\partial}{\partial \gamma_i} J(\gamma_i) \tag{3.7}$$

In which:

$$\gamma_i = S(\theta_i) = \frac{1}{1+e^{-\theta_i}}$$

3.2.3 Random Restarts

Additionally, non-convex domains are ineffective for conventional gradients. When non-convex optimisation is used, the search result almost always contains a local minimum. To maximise the gradient descent algorithm's effectiveness, the following scheme may be used: This is accomplished by randomly selecting (uniformly) points in K and initiating a local optimization routine on a subset of them; this may result in the determination of some local optima, but each result is

stored in an external list; with the appropriate repeating times or leaning rate setting, global optima are expected to be included in the records.

Apart from the simplicity and robustness of the algorithm, another advantage of this concept is the massive increase in computational power. Gradient computation is more efficient than ever before due to the rapid advancement of parallel computing hardware such as NVIDIA's CUDA and Google's TPU [22], and it is the gradient-based algorithm that benefits the most from the recent five-year increase in computation power.

3.3 Algorithm: Gradient Descent with Random Restarts

As introduced in the section 3.2, we synthesised our ideas with respect to (1). dealing with gradient descent, (2). conditional constrains and (3). random initialization into our first algorithm, forming the first algorithm that we proposed. Generally Speaking, we proposed the algorithm as Algorithm Chart 1:

Algorithm 1 Gradient Descent with Random Restarts (GDR)

Require: α: Learning Rate
Require: $J(\gamma)$: Stochastic objective function
Require: N: Number of restarting points
  Initialize a null list $result$[]
  for n = 1 to N **do**
    Initialize $\vec{\theta}$ arbitrarily
    for t = 1 to T **do**
      $\vec{\gamma}_t \leftarrow 1/1+e^{-\vec{\theta}_t}$
      $\vec{\gamma}_t \leftarrow \vec{\gamma}_t + \alpha \cdot \nabla J_t(\gamma_{t-1})$
      cosine annealing α every $\frac{T}{4}$
    end for
    $result[n] \leftarrow J_t(\gamma_T)$
  end for
  ranking $result$, choose the maximum $J^*(\gamma)$ from $result$
  return $J^*(\gamma)$

Our approach consists of two iterations: an out loop for several trials and a gradient descent iteration. Because our objective is to discover the greatest extreme point, the update direction is the inverse of the gradient point's direction (or what we refer to as gradient ascent), and the independent variables are scaled to the limited range before being updated.

Apart from the design considerations covered in Section 3.2, we introduced a widely acknowledged method for adaptively modifying the learning rate over time: the cos annealing formula, which decays the learning rate periodically. The purpose of adjusting the learning rate during iteration is to handle the issue of overshooting [36], which has a substantial effect on the final outcome.

Once the results were all collected after N times random trial, the largest value will be picked, taken as the global optima.

3.4 Experiment and Results

In this section, we will introduced a series of experiment that involving our proposed GDR algorithm, targetting to verify and investigate the performance of our algorithms from three aspects, namely *Effectiveness*, *Robustness*, and *Scalability*

Before the induction of each experiment, the following table summarises the software environment and computing hardware used in this chapter.

Software Configuration	
Name	Version
Python	3.6
PyTorch	1.11.0
CUDA	10.6
MATLAB	2022a
Linux Ubuntu	20.04

Hardware Configuration	
Name	Specification
CPU	Intel i9-9900x
GPU	Geforce RTX 2080 12GB
RAM	64 GB

Table 3.1: The external environment configuration for experiments on GDR

3.4.1 Effectiveness

We used the aforementioned SISO Interference channel as our communication channel model in the first experiment. To begin, the channel configuration is simplified to a 2×2 SISO channel in order to validate the proposed algorithm's effectiveness. The object function in a 2×2 SISO channel could be expressed as:

$$\max_{\gamma_1,\gamma_2}(\log_2(1+\frac{|h_{11}|^2\gamma_1}{|h_{12}|^2\gamma_2+\sigma^2})+\log_2(1+\frac{|h_{22}|^2\gamma_2}{|h_{21}|^2\gamma_1+\sigma^2}) \tag{3.8}$$

$$s.t.\ \gamma_1,\gamma_2 \in [0,1]$$

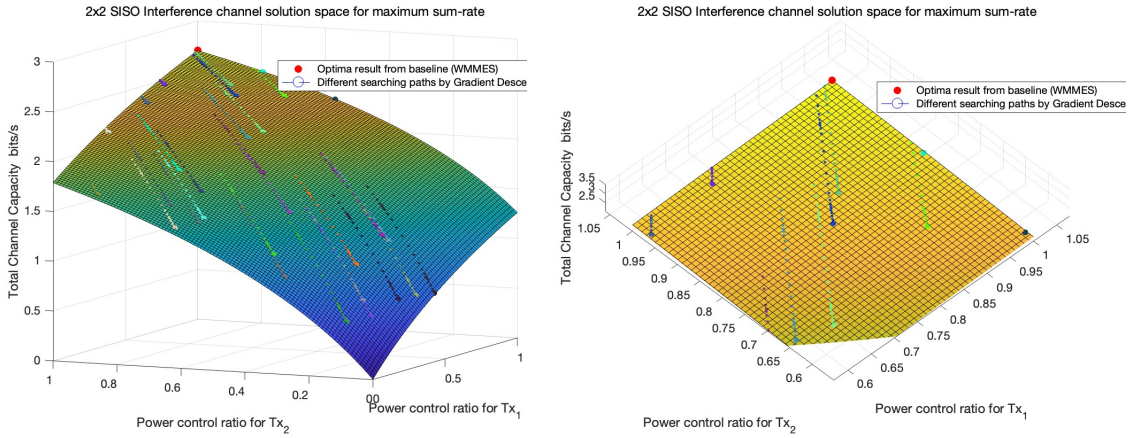
Because only two independent variables γ_1 and γ_2 exist in the 2×2 SISO channel, the entire solution space can be visualised into a three-dimensional space, along with the complete trajectories of each iteration. After generating the channel in the simulation environment, we started to implement our GDR algorithm to find the optima power control ratio, using the parameter settings in table 3.2:

During the iteration of algorithm, we kept track of each position $\{(\gamma_1,\gamma_2,R)\}$ as we searched for the optimal solutions using GDR. Following the algorithm's execution, in order to visualize the searching process and results, we drew the entire solution space of the target sum-rate function as equation 3.8 and recovered the search trajectories from the recorded data.

Parameter	Symbol	Value
Restarts Time	N	50
Iteration Epochs	T	100
Learning Rate	α	$1e-3$
Random Seed	/	308

Table 3.2: The parameter settings of GDR algorithm

In addition to the GDR algorithm, the result from the WMMES algorithm introduced in Section 1.2 was also derived and plotted on the same solution space, working as a baseline for comparison. The final result is shown as in Fig. 3.3



(a) Searching Trajectories Overview (b) Best Searching Trajectories

Figure 3.3: The searching trajectories for finding optima power control ratio in a 2×2 channel. (a) Different color of lines represents different searching trajectories by GDR at each time. Thicker points of each trajectories marks as beginning points. (b) The top view capture of the optima searching trajectories. Among 50 different trajectories, 2 of them reached to the highest point in this solution space

The curved surface in the graph represents the solution space within the constrained range (0,1). The Z-axis represents the sum-rate that the system could achieve given it's position in the solution space. Different color on the plan illustrates different searching route of gradient descent. Each color points stands for a solution that the algorithm found in one iteration. The red dot was the solution found by WMMES, which is also visually proofed as the baseline, since the red dot reaches the highest position in the solution space.

Among the trends that the graph shows, We observed serial valuable trends and phenomena in accordance with the our design expectations :

- All search paths were scaled to fit within the predefined range. As the searched point approached the boundaries, the route was constrained by using the sigmoid function and hovering near the boundaries.

- Two of those trials found the optimal solution among 50 randomly chosen trajectories, returning the optimal sum-rate as 2.84 bit/s with a minor error range of $+-0.02 bits/s$ in comparison to the baseline result.

The preceding observations confirmed our assumptions about the algorithm's design. Thus, in the subsequent stage, we will extend the channel configuration and evaluate the performance of the GDR algorithm.

3.4.2 Robustness

The first experiment presented an intuitive interpretation and comparison in a simplified channel configuration, initially implying the effectiveness of our algorithm. Based on the positive result, we designed a more complicated environment to further investigate the robustness of the algorithm.

As our optimisation objective function *equation 3.4* indicated, there are two random variables that affect the channel capacity apart from the dominates factor (transmission power ratio):

1. *Channel Coefficients* $\{|h_{ij}|\}$: Where the channel between each Tx and each Rx is randomly generated according to a Rayleigh fading distribution

2. *Signal-to-Noise Ratio(SNR)*: Here, the average SNR is controlled by the AWGN noise parameter $1/\sigma^2$ in our channel formulation, larger values reflect larger average SNR.

These two factors are heavily affected by the practical surrounding scenarios. Therefore, we designed two independent experiment to investigate the performance of our algorithms considering the noise level ($\frac{1}{\sigma^2}$) and channel gains ($\{|h_{ij}|^2\}$) respectively.

On the other hand, in order to compare the algorithm broadly, apart from the popular baseline algorithm WMMES, we also make our comparisons against the rest of existing heuristic methods for solving the problem. We primarily consider following algorithms for comparison:

1. *WMMSE algorithm*: Which is widely adapted due to it's guarantee performance

2. *Equal distribution*: Assign equal power $P_{\max/N}$ to all users

3. *Random distribution*: Randomly distributed power $P_{\max}/P_0$ to all users.

In the experiment to investigate the algorithm performance among different SNR ratio, we first unified the non-related variables across the channel. Here we fixed our number of Channel link $N = 4$, and set both of the random distribution with zero mean and unit variance. After those settings, we evaluate the sum-rate result of the our algorithms over the baselines in different SNR regimes. To vary the average SNR, we set the AWGN noise parameter $1/\sigma^2$ to vary between [0,10], where a larger value reflects a higher average SNR. The simulation result is shown in Fig. 3.4.

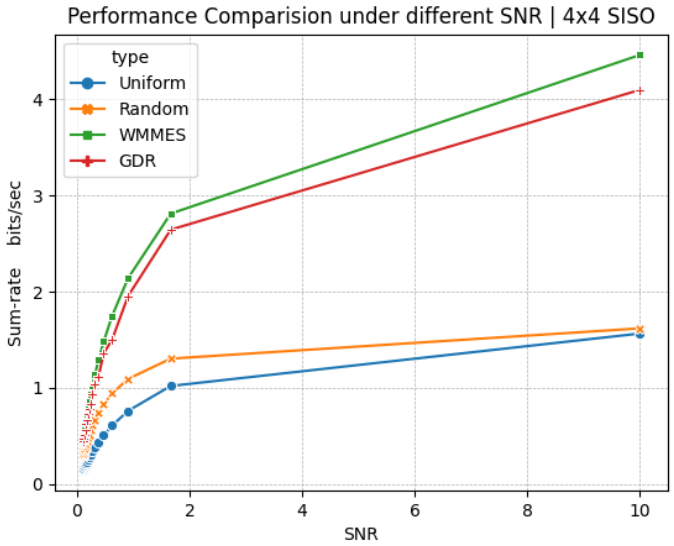


Figure 3.4: Algorithm Comparison of performance against baselines under various SNR regimes. The capacity is categorically divided into two clusters, with the upper group indicating a higher level of performance in terms of achieving the higher channel capacity.

As demonstrated, four algorithms' performance may be categorised into two distinct groups. WMMES and GDR algorithms function almost identically to heuristic algorithms. However, we can see that the performance difference between the WMMSE and GDR algorithms is tiny at first but steadily increases as SNR increases. According to our prior analysis of the interference channel, when the signal-to-noise ratio is large, the desired signal strength takes precedence over the channel capacity rather than the interference links, flattening the solution space. The increase of the disparity suggests that our GDR is overshooting the global optima during search in this case. On the plus side, in our experiment's worst 10 dB SNR condition, the GDR algorithm outscored the heuristic algorithms by roughly twice. This finding supports the hypothesis that while the GDR is capable of withstanding external environment affections, it is easily influenced by parameter choices such as the updating step length.

Apart from noise, another source of uncertainty during signal transmission is the variation in channel gains. In the second experiment, we fixed the AWGN noise to 1dB but randomly generated 1,000 4×4 channels following the Rayleigh distribution. We continued to apply the aforementioned four algorithms to each of the channels in this case. Here, we recorded the solutions from each algorithm, computed the sum rate based on the solutions, and calculated the cumulative probability of the algorithm achieving different results over the 1,000 channels.

The cumulative probability of four algorithms were plotted from recorded findings in Fig 3.5. The WMMES method is anticipated to reach a maximum 2.4 bit/s channel capacity with a probability of 0.8 over 1,000 trials on channels with varied degrees of attachment, while the GDR algorithm achieves a similar performance of around 2.2 bit/s with the same probability.

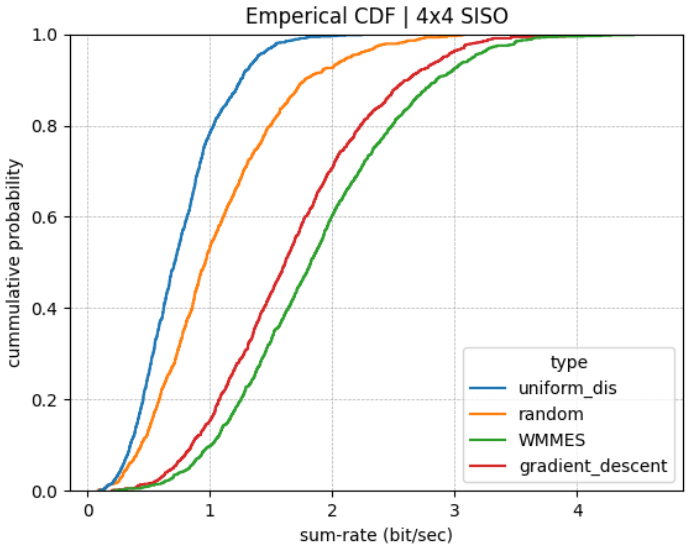


Figure 3.5: The cumulative distribution function (CDF) that describes the rates achieved by different algorithms, over 1,000 randomly generated 4x4 SISO channels

In compared to the heuristic method, the GDR anticipates a maximum rate about double that of the Uniform and Random distribution schemes with the same probability. This observation corroborates the first experiment's conclusion. Due to the fact that changes in noise level or channel gain represent all external affections across the entire signal transmission, the similar performance of GDR indicates that when external affections are small, the GDR is expected to find the optimal power allocation strategy just like WMMES, whereas when external affections are strong, the GDR demonstrates its capacity-enhancing ability but is expected to be improved through parameter adjustment.

3.4.3 Scalability

In the third series experiment, We increase the total number of links to investigate the scalability of our algorithm. Here we applied five channel configurations where we increased the number of transmitter-receiver pairs N from the previous configuration 2 up to 10.

The simulation result were presented in Fig. 3.6. Where the various groups denote the sum-rate values obtained for each channel configuration.

It is noticeable that under the simple channel configuration i. e. the number transceiver pairs under 4, the GDR algorithms and WMMES achieved a similar performance of about 1.3 bit/s, meanwhile the relatively short error bar illustrates the output is stable. However, unlike the observations from the robustness experiment, as the increase of the numbers of pairs, the performance of GDR significantly decays compared with the baseline. In particular, as the number of pairs increased from 6 to 10, the result from the GDR decreases from 2.2 bit/s to 2.0 bit/s. The

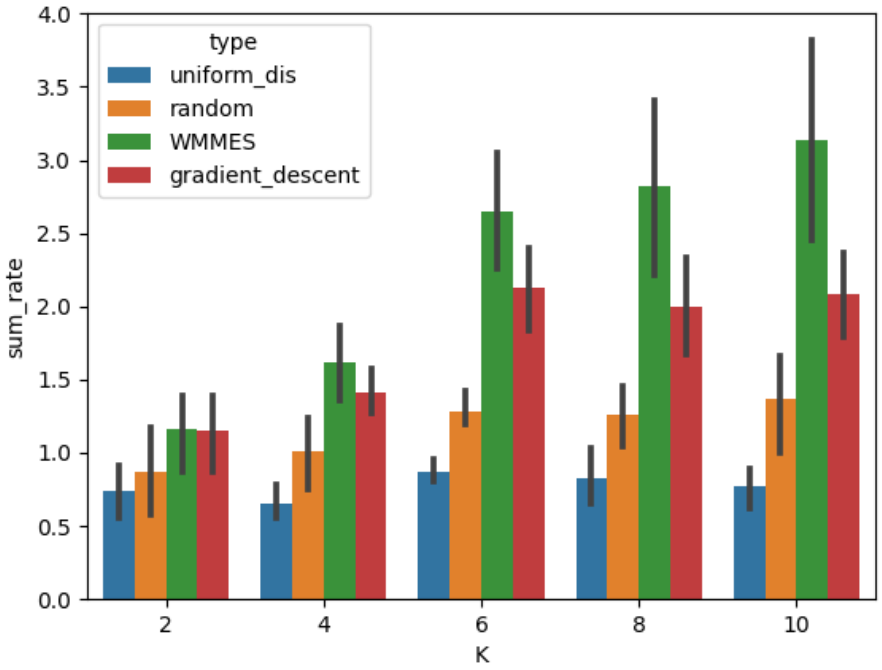


Figure 3.6: The channel capacity achieved by the algorithm in various channel configurations ranging from 4x4 to 10x10 SISO channels. The algorithm was applied to 16 channels with varying channel gains in each configuration; the WMMES algorithm was also applied in the same manner for comparison.

degeneration of GDR algorithm indicates it's performance is heavily affected by complexity of the environment. Furthermore, the degeneration of GDR algorithm indicates that the increase of transceiver pairs results in a much more complicated solution space for the GDR compared with the change of SNR and channel gains. And on the other hand it highlights the drawback of stochasticity of GDR algorithm, where an optima result is not guaranteed from GDR.

3.5 Conclusion and Reflection

In this chapter we first proposed our optimisation algorithm: gradient descent with random restarts (GDR) for the sum rate maximisation problem.

The effectiveness was first confirmed.We verified that the designed functionality of boundary constraint and result ranking were all achieved, and that the final result is as excellent as the existing baseline method. However, the GDR algorithm has some flaws in the following tests. The complexity of the channel, which distorts the solution space, affects the GDR algorithm's optimality. For example, while SNR increases in the second trial, GDR finds no improvement in the solution. The non-adaptive upgrading step length of GDR made it easy to miss the global

optima during iteration; furthermore, the growth of transceiver pairs, resulting in a more complex solution space, made the GDR searching fall into a local convergence point.

On the plus side, the GDR algorithm continues to outperform the other two heuristic algorithms, following the same trend as WMMES. To further enhance the GDR algorithm in future work, the following enhancements should be considered:

- Developing a more effective tuning strategy for hyper-parameters. Thus far, the primary source of error in the GDR algorithm has been defined as overshooting/missing global optima due to inefficient searching routes. While currently, only the cosine annealing function is used to optimise the search route. A more precise method of controlling the hyper-parameters like learning rate and initial starting point is desired for future work.

- Taking into account the repeating durations of random restarts. The GDR algorithm's central idea is to attempt to cover the global optima by searching multiple local optima. Currently, the repeating time for all objectives is set to 50 by default in the current settings considering the searching efficiency. For future work, it is desired to determine the relationship between the number of random restart points required for various objective functions. So that we can achieve a more favourable balance of search speed and accuracy.

CHAPTER 4

REINFORCEMENT LEARNING FRAMEWORK

Despite machine learning has been the popular techniques that widely adapted to solve a wide range of non-convex optimisation problems in the industry, the bottleneck that consistently exists is the demand for large amount of high-quality labelled data, including the solution schemes to the sum-rate maximisation problem. Inspired by my previous research work, reinforcement learning came to the mind as a promising way to over come this obstacle. In this chapter, we will introduce the theory and background of reinforcement learning, tailoring our own reinforcement learning training environment, implementing the popular policy gradient based algorithm and ending with the comparison between the reinforcement learning algorithm and the mainstream algorithms in this field.

4.1 Method Introduction

4.1.1 Reinforcement Learning

Reinforcement learning dates back to the 1990s of work in statistic and computer science [16]. It holds the attraction that it offers a way of programming a decision bot by reward and punishment without needing to specify how the task is to be achieved. In the recent decades, it has enjoyed a wide variety of successes like teaching computer to play Go and video games.

Fig 4.1 shows the main character of Reinforcement Learning are the *agent* and the *environment*. The environment is the world that the agent interacts with. At every step of interaction, the agent receives the observation of the state of the environment, and then decides on an action to take. The environment changes when the agent acts on it (but may also change on its own). And the best decision policy for the task can be gradually formed by receiving the rewards (or punishment) from the environment.

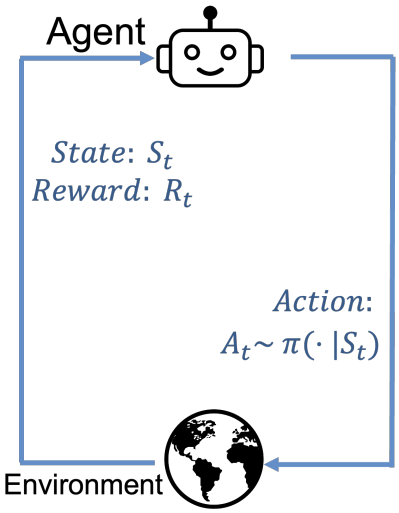


Figure 4.1: The Schematic of Reinforcement Learning

In the formal expression, the iteration can be described as the Markov Decision Process (MDP) **M**, which comprises 5 elements $< \mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{R}, \rho >$, representing the state space, action space, transition probability, reward and discount factor respectively. Since the decision policy is changed according to the different state that the agent receives, it can be expressed as a probability distribution following the current state:

$$a_t \sim \pi(\cdot|s_t) \tag{4.1}$$

Where π denotes the policy, s denotes the state and the action of the agent at time t : $a_t$is given by the probability $\pi(\cdot|s_t)$

The rewards is accumulated while the interaction between the agent and environment. Suppose the environment transitions and the policy are stochastic, the reward can be expressed as mathematical expectation of sum of the reward that collected during the iteration:

$$J(\pi) = E_{\tau\sim\pi}[R(\tau)] \tag{4.2}$$

Where τ is the set of all states and actions that produced in one iteration, or saying 'trajectory':

$$\tau = (s_0, a_0, s_1, a_1, \cdots) \tag{4.3}$$

The ultimate goal of reinforcement learning training is to maximize the expected return, by finding the best decision policy:

$$\pi^* = \arg\max_{\pi} J(\pi) \tag{4.4}$$

4.1.2 Policy Gradient

Policy Gradient is one of the most popular method class in reinforcement learning, resulting in a variety of State-of-the-art algorithms in recent years. The idea underlying policy gradients is to push up the probabilities of actions that lead to higher return, and push down the probabilities of actions that lead to lower return, until reaching at the optimal policy.

Considering the aforementioned expression of the objective function $J(\pi_\theta)$ which is affected by the decision policy π_theta. since the it's differentiablity, the value could still be optimized through gradient descent, which was shown in Chapter 3:

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta J(\pi_{\theta t}) \tag{4.5}$$

Where θ is the dominant factors that control the policy. However, it is often hard to formulate the decision policy π and it's variables θ considering the variety of decision scenario. Instead, the Artificial Neural Network (ANN) is utilized to estimate the policy function π_θ.

Taking the differentiation of the objective function in more detail, a key formulation in policy gradient reinforcement learning is found from the derivation [34] :

$$\nabla_\theta J(\pi_{\theta t}) = E_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t] \tag{4.6}$$

Two important expression can be observed from equation 4.6: 1) $\log_{\pi_\theta}(a_t|s_t)$ represents the log probability of the current action selected by the policy; 2) $\hat{A}_t$ is the advantage estimator, which evaluates the goodness of the current action. Intuitively speaking, the optimization on the reward function $J(\theta)$ is achieved by increasing the probability of 'good' action and vice versa.

4.2 Reinforcement Learning Environment Design

In a typical Reinforcement Learning (RL) problem, there is a learner and a decision maker called agent and the surrounding with which it interacts is called environment. The environment, in return, provides rewards and a new state based on the actions of the agent. So, in reinforcement learning, we do not teach an agent how it should do something but presents it with rewards whether positive or negative based on its actions. So our root question for this blog is how we formulate any problem in RL mathematically.

As mentioned in the last section, a MDP comprises 5 elements: $< \mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{R}, \rho >$. By properly design the mathematical expression according to our problem setting, a reinforcement learning algorithm like VPG could work. Considering the SISO environment setting, we proposed the following design of the Markov Decision Process:

- State Space: **S**

S is a set of countable nonempty states, which is a set of all possible states of the system. In the SISO interference channel, the current channel matrix $|h_{ij}|$ and power control level are the factors that affects the next decision of power allocation.

Additionally, we emphasise the issue that in the original iteration, different transmitter are expected to be allocated different level of power control ratio synchronously, resulting in a multi-dimensional action space. However, the complexity of action space could lead the original problem setting upgrade to a Markov Game, and multi agents instead of one takes charge of the decision of one Transmitter. However the evolution of the environmental state and the reward function that each agent receives will thus, be determined by all agents' joint actions. As a result, agents need to take into account and interact with not only the environment but also other learning agents.

To simplify this process, we add the third type of information in S, which is the index of each transmitter. By adding the information of decision target in the S, we hope to compress the action space into a vector, manually change the index label at each iteration and thus simplifying the process as a sequential decision problem. Compared with the Multi-agent frame work, our simplification is expected to achieve a similar performance but takes longer training time.

Thus, the our design of the state space S can be expressed as:

$$\mathbf{S}: \{\underbrace{h_{11}, h_{12}, \cdots, h_{NN}}_{N^2 \ in \ total}, \underbrace{\gamma_1, \gamma_2, \cdots, \gamma_N}_{N \ in \ total}, n\} \tag{4.7}$$

- Action Space: A

  Since the simplification in the state design has done by adding extra index information of transmitter, the action space can thus be identical during the whole decision process. Here the action is the addition/subtraction value of power control ratio within the continuous range of [-1,1]:

$$\mathbf{A}: a, a \in [-1, 1] \tag{4.8}$$

- Reward: R

  The reward design is expected to set up the positive connection between the action and consequence. Thus, the sum-rate function, which links the power allocation and channel capacity is taken as the reward design:

$$\mathbf{R}: \sum_{i=1}^{N} \log_2(1 + \frac{|h_{ii}|^2 \gamma_i}{\sum_{j \neq i} |h_{ij}|^2 \gamma_j + \sigma^2}) \tag{4.9}$$

- State Transition Probability: P

  P describes the probability that the state at the time step t, given that action A, transit to the successor state S'. We assume no external affection in our environment, thus the state transition probability is purely determined by the decision policy.

(4.10) $$\mathbf{P} : P(s'|s,a)$$

- Discount Factor: ρ

  When evaluating the quality of an action at time step t in MDP, instead of counting the total rewards of it's trajectory, discount count factor exponentially decaying the affects of rewards that away from time step t. This is because an infinite-horizon sum of rewards may not converge to a finite value, and is hard to deal with in equations. Or intuitively saying: cash now is better than cash away. The factor is usually a decimal between [0,1], in our setting:

(4.11) $$\rho : 0.9$$

Once finished the process design, we coded and packaged these ideas as a library through OpenAI Gym, established a standard reinforcement learning environment for the subsequent experiment. OpenAI Gym is a toolbox in Python that offers the instant APIs to defining the customized reinforcement learning environment.

4.3 Algorithm: Vanilla Policy Gradient

As introduced in the first section, we applied a class of policy gradient method reinforcement learning algorithm in our environment setting. Specifically speaking, we took the vanilla policy gradient (VPG). The difference is mainly about the different choice of the advantage function estimator. As we discussed in section 4.2, the gradient estimation is:

(4.12) $$\nabla_\theta J(\pi_{\theta t}) = E_{\tau\sim\pi_\theta}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)\hat{A}_t]$$

In VPG, our advantage estimator is:

(4.13) $$\hat{A}_t = \sum_{t'=t}^{T} R(s_{t'}, a_{t'}, s_{t'+1}) - V_\phi(s_t)$$

In equation 4.13, the subtracted number is reward-to-go, literately represents the sum of rewards that obtained after the current action. Compared with the sum of rewards of the whole trajectory, this evaluation of the quality of the action significantly avoids the bias since agents are supposed to only reinforce actions on the basis of their consequences.

On the other hand, the subtractor in equation 4.13 works as an baseline, namely the value function $V^{\pi}(s_t)$ here. According to the investigation in [], the choice of the value function has the desired effect of reducing variance in the sample estimate for the policy gradient.

However, the value function $V^{\pi}(s_t)$ cannot be computed exactly, so it has to be approximated. This is usually done with a neural network, $V_{\phi}(s_t)$, achieving by minimizing the mean-squared-error between the reward-to-go and the estimation from the neural network.

In conclusion, two neural network were used in the vanilla policy gradient algorithm, targetting the following objective function:

$$\begin{cases} \theta_k = \arg\max_{\theta} E_{\tau\sim\pi_\theta}[\sum_{t=0}^{T} \log \pi_\theta(a_t|s_t)\hat{A}_t] \\ \phi_k = \arg\min_{\phi} E[(V_\phi(s_t) - \hat{R}_t)^2] \end{cases} \tag{4.14}$$

Synthesising the choice the advantage function estimator $\hat{A}$ and our customized training environment, the overall procedures can be summarized as the following pseudocode:

Algorithm 2 Vanilla Policy Gradient Algorithm

Require: θ_0: initial policy parameters
Require: ϕ_0: initial value function
Require: T: Maximum time step of each interaction
Require: N: Maximum number of iteration
for $k = 0, 1, 2, \cdots, N$ **do**
Collect set of trajectories $\mathbf{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
Compute rewards-to-go $\hat{R}_t$.
Compute advantage estimates as

$$\hat{A}_t = \sum_{t'=t}^{T} R(s_{t'}, a_{t'}, s_{t'+1}) - V_\phi(s_t)$$

Estimate policy gradient as

$$\nabla_\theta J(\pi_{\theta t}) = \frac{1}{|\mathbf{D_k}|} \sum_{\tau \in \mathbf{D_k}} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)|_{\theta_k} \hat{A}_t$$

Compute policy update using gradient descent as

$$\theta_{k+1} = \theta_k + \alpha_k \nabla_\theta J(\pi_{\theta t})$$

Fit value function by regression on the mean-squared error via gradient descent

$$\phi_{k+1} = \arg\min_\phi \frac{1}{|\mathbf{D_k}|T} \sum_{\tau \in \mathbf{D_k}} \sum_{t=0}^{T} (V_{\phi(s_t)} - \hat{R}_t)^2$$

end for

4.4 Experiment and Results

Targeting the same sum-rate maximisation object, we designed our own reinforcement learning environment this time to apply the reinforcement learning algorithm. The problem model has been introduced in the section 3.1. Here, we implemented our algorithms considering the following settings:

1. We implemented the algorithm into 3 channel configurations: 6x6 SISO channel; 8x8 SISO channel; 10x10 SISO channel

2. We assume the same level of AWGN noise where $\sigma^2 = 1$ for all channels

3. For simplicity, we set the maximum transmission power $P_{\max} = 1$

Apart from the channel configuration, we applied the hyper-parameter settings as shown in Table.4.1 in the VPG algorithm:

Parameter Settings for VPG		
Parameter Name	Symbol	Value
Episode Number	N	2000
Iteration Time	T	100
Learning Rate	α	1e-4
Discount Factor	γ	0.98
Hidden Layer for a	/	(128,128,128)
Hidden Layer for b	/	(128)

Table 4.1: The parameter settings of VPG

In the first experiment, we applied four different algorithms under the same 4x4 SISO channel. As introduced before, the testing algorithms comprises the baseline algorithm: WMMES, two heuristic algorithm: the uniform and random distribution schemes and our reinforcement learning algorithm: VPG.

The recording of the performance variation during the modeling training was plotted in Fig 4.3, where x axis represents the time step of iteration and y axis represents the sum rate. In order to gain an evaluation of the performance that the model achieved, the rest of three baseline were also applied here. For the results of VPG, the curve had smoothed and flattened, indicating that it's policy had converged.

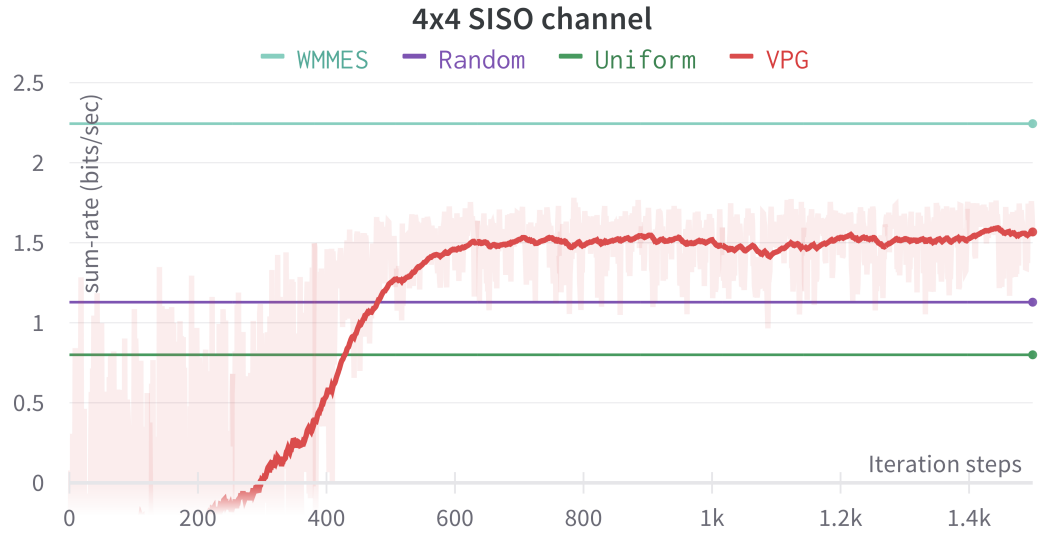


Figure 4.2: The sum-rate obtained by each of algorithms during the training stage of VPG

Additionally, the WMMES maintained the optimal results of approximately 2.3 bit/s throughout the training. It can be seen that at the start of training, the VPG model has limited knowledge

about its surroundings, and thus the current strategy produces the worst results; however, as its interaction with the environment increases to approximately 400 steps, its policy begins to improve significantly, indicating that our neural network weights (θ, ϕ) are implicitly improving. At this stage, VPG's performance significantly exceeds that of the Random and Uniform algorithms. However, We have to admit that at the convergence stage, the performance the the model outputs is maintained within the range around 1.5 bit/s, about 35% below of the WMMES algorithm.

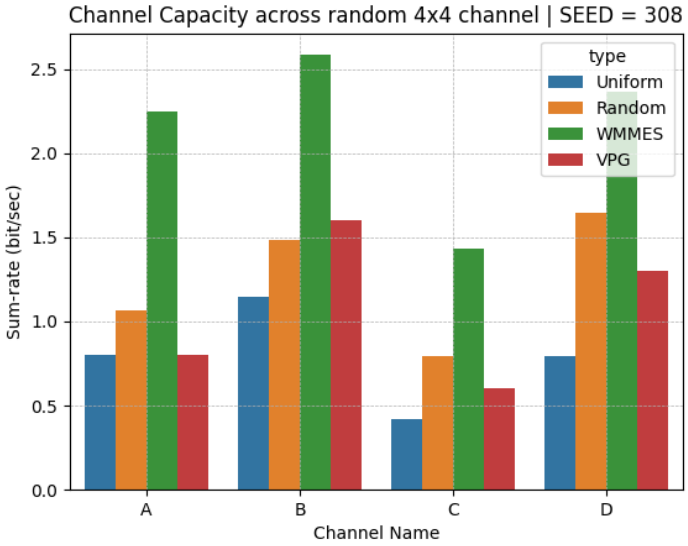


Figure 4.3: The sum-rate obtained by each of algorithms during the training stage of VPG

The first experiment shows the convergence of training on VPG, but to study the performance difference in more details and investigate the potential solutions, we thus carried out the VPG algorithm in a wider scenarios after finished training. As Fig.4.3 shows, in the second experiment, we applied our VPG model, along with the rest of three baselines on 4 4×4 channels with random channel realizations. Across the all channels, the VPG algorithm has an average performance difference of about 40% from the WMMES algorithm. In 3 out of 4 cases, the sum-rate the VPG derived is about 20% lower than the results from Random algorithm. These differences indicates the difficulty of training a reinforcement learning task from the aspect:

- During the model training stage, the strategy of balancing the resource allocation across the whole channel is difficult for the single agent. This results that despite the reward is increasing, the strategy of agent can not return the best solutions.
- During the model validation stage, the adaptation to the variation of the environment is difficult for a trained model. Through the comparison of VPG performance between Fig 4.2 and Fig 4.3 illustrates that the effectiveness of training only appears in a certain range of environment. The changes in Fig 4.3 is relates to the input state $\mathbf{S}$, which means that any significant change during the Markov Decision Processes will affect the final performance.

Furthermore, the above experiment presented the limitation of our Markov Decision Process design but the previous trials was only focused on a single environment configuration. Thus, it is desired to investigate the effectiveness of the MPD design in more detail. In the third experiment, we extend the channel configuration from 4x4 SISO channel to 10x10 SISO channel using the same VPG algorithm.

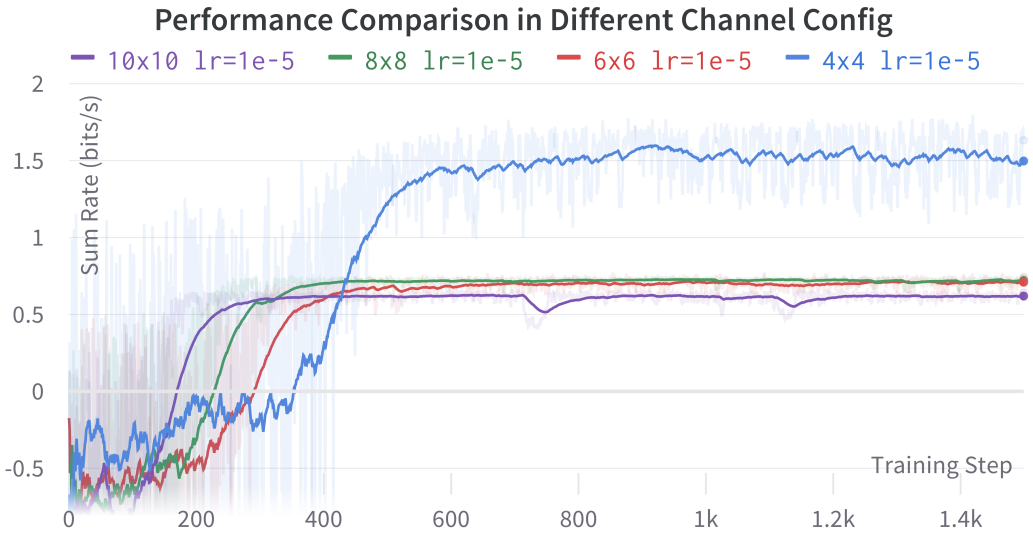


Figure 4.4: The training result under different channel configurations using VPG

The training results under 4 channel configurations were plotted in Fig. 4.4. Using the same learning rate, it can be observed that the training under 10×10 channel environment was first appears to converge at the time step of about 180. Among the other trials, the time of emerging convergence of 8×8 was slightly increased by 20 steps, starting to improving the it;s policy at 200 time step. The similar trend is applied to the rest of channel, where we can find that the time of starting convergence is reversely proportional to the number of transceiver pairs. As the conclusion that we found from the experiments in Chapter 2, the complexity of solution space is highly affected by the channel configuration. While here, the more complex channel results in a earlier time of convergence, which indicates that:

1. Our design of environment (MDP) is effective since it offers the positive rewards to the proper actions of agent

2. The approximation for the decision strategy (Policy), which is achieved by deep learning networks in VPG, is easily to get fall into a local minimum as the complexity of solution space increases.

From another prospective to analyse, apart from the performance under 4×4 channel, it is theoretically expected that the channel capacity would increase as the increase number of

transceiver pairs, due to the increasing of multiple access within the given channel frequency. However, the similar final results were found in the $6 \times 6, 8 \times 8$ and 10×10 channels, whose results all hovers around 0.6 bit/s. It shows that the approximation by the agent was significantly constrained in a limit range as the increase complexity of the environment.

4.5 Conclusion and Reflection

In this chapter, We investigate the Reinforcement Learning (RL) framework. Training Environment and Training Algorithm are two key factors in the reinforcement learning. We first proposed our own environment design based on the target problem, following by the introduction of the classical policy-gradient based RL algorithm - Vanilla Policy Gradient (VPG).

Our initial experiment validated our proposed method's effectiveness while also revealing RL's poor performance compared to existing baselines.

Comparing VPG performance across several channel configurations revealed two key reasons for our RL framework's poor performance. (1). With more transceiver pairs, the deep learning structure becomes repetitive and cannot approach the complex solution space. (2). The single agent architecture is difficult to train to handle a large number of receivers because it performs best with the fewest couples. Based on these findings, two further directions are expected:

1. Investigating a richer structure design in VPG algorithm.

   Considering the ability of the deep learning prediction, it is the multi-layer feed-forward artificial neural network with finite amount neurons in hidden-layer that can predict each relationship between the dependent and independent parameters of a desired variable.[17]. Simplifying the action space is one of our RL design ideas. In this way, we want agent to implicitly examine the relationship between different solutions to different transmitters. The last experiment shows that increasing nonlinearity complexity becomes the algorithm's bottleneck. Thus, a well-designed structure is expected to boost the DNN's nonlinearity prediction abilities.

2. Considering the multi-agent design for the problem. Since the power limitation is individual applied for each transmitter, the whole process of balance could be considered as a no-cooperative game [14]. As a result, a recent innovation in reinforcement learning has been made: multi-agent reinforcement learning [**?**] developed exclusively for this gaming process. Despite the fact that this framework demands an exponential increase in computational capacity, it is projected to deliver the greatest outcomes during dynamic allocation.

CHAPTER 5

SUMMARY

Spectrum resources is of great significance in the informatics age. In this thesis, we mainly studied an optimisation problem that aims to maximize utilization of our limited spectrum resource, namely, the sum-rate maximisation problem. The main change of this optimisation problem mainly involves with it's non-convexity due to treating interference as noise (TIN), the key idea to formulate the target.

Apart from current methods, we offered two novel approaches to solve this non-convex optimisation issue, which were inspired by recent advances in computational technology and earlier research. More precisely: (1) We proposed a new algorithm: Gradient Descent with Random Restarts. (2) We brought the Reinforcement Learning (RL) algorithm Vanilla Gradient Descent (VPG) to this problem by presenting a fresh architecture of the reinforcement learning training environment. All of our proposals were developed mostly in Python and validated through a series of experiments conducted in a simulation environment.

We conducted a series of experiments to validate the performance of our initial method, Gradient Descent with Random Restarts (GDR). Across all studies, we compared our GDR method to three alternative baselines, evaluating its performance on three dimensions: effectiveness, robustness, and scalability.

In the first stage of the experiment, we produced an inspiring outcome by validating the GDR algorithm's effectiveness. By viewing the entire optimization process, we ensured that our intended functionality of boundary constraint and result ranking was implemented correctly and that the final result is on a par with the existing baseline algorithm.

However, certain key difficulties with the GDR algorithm were identified in the following trials where we varied the channel complexity. During the robustness analysis, it was discovered that the complexity of the channel, which results in solution space distortion, has a substantial

effect on the GDR algorithm's optimality. By examining the relationship between the channel and the solution space, we conclude that the gradient iteration nature of GDR made it easy to miss the global optima during iteration; additionally, the addition of transceiver pairs, which results in a more complex solution space, caused the GDR search to fall into a local convergence point and terminate far from the global optima.

We then study the Reinforcement Learning (RL) framework in the second stage. Following the construction of the Markov Decision Process (MDP), the fundamental mathematical expression of training environment design, we conducted a series of experiments using the VPG algorithm in conjunction with the established training environment. We demonstrate the success of our suggested strategy in the first experiment, but have to concede that utilising RL performs poorly in comparison to established baselines. We are inspired to validate the applicability of our created MDP based on our initial observation of the findings of the first experiment comparison.

According to our assessment of VPG performance across various channel configurations, we discovered two primary reasons for our RL framework's low performance. (1). The deep learning framework we used is too repetitive to approximate the complicated solution space, trapping the agent as the number of transceiver pairs increases. (2). The single agent architecture is difficult to train to centrally control a large number of receivers, as it performs optimally in configurations with the fewest couples.

Both of our proposed strategies are demonstrated to be effective in the preceding experiment, although their effectiveness is considerably limited by serial disadvantages, such as the capacity to overcome local convergence spots. Apart from future works that we conclude on each chapter, we continue to value these two recommendations. Because (1).our first algorithm is universal, it can be readily applied to the distortion channel. (2). Our training environment eliminates the requirement for manual data labelling during deep learning training. As may be seen in the future, more ways based on recent advancements in machine learning techniques will be proposed, and we intend to continue this research based on the results and experience gained thus far.

BIBLIOGRAPHY

[1] K. ASANOVIC, R. BODIK, B. C. CATANZARO, J. J. GEBIS, P. HUSBANDS, K. KEUTZER, D. A. PATTERSON, W. L. PLISHKER, J. SHALF, S. W. WILLIAMS, ET AL., *The landscape of parallel computing research: A view from berkeley*, (2006).

[2] K.-L. BESSER, B. MATTHIESEN, A. ZAPPONE, AND E. A. JORSWIECK, *Deep learning based resource allocation: How much training data is needed?*, in 2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), IEEE, 2020, pp. 1–5.

[3] S. S. CHRISTENSEN, R. AGARWAL, E. DE CARVALHO, AND J. M. CIOFFI, *Weighted sum-rate maximization using weighted mmse for mimo-bc beamforming design*, IEEE Transactions on Wireless Communications, 7 (2008), pp. 4792–4799.

[4] L. DATTA, *A survey on activation functions and their relation with xavier and he normal initialization*, arXiv preprint arXiv:2004.06632, (2020).

[5] C. DEPARTMENT FOR DIGITAL, *Spectrum strategy*, Mar 2014.

[6] M. EISEN AND A. RIBEIRO, *Optimal wireless resource allocation with random edge graph neural networks*, ieee transactions on signal processing, 68 (2020), pp. 2977–2991.

[7] M. EISEN, C. ZHANG, L. F. CHAMON, D. D. LEE, AND A. RIBEIRO, *Learning optimal resource allocations in wireless systems*, IEEE Transactions on Signal Processing, 67 (2019), pp. 2775–2790.

[8] C. GENG, *Fundamentals of treating interference as noise*, University of California, Irvine, 2016.

[9] C. GENG, N. NADERIALIZADEH, A. S. AVESTIMEHR, AND S. A. JAFAR, *On the optimality of treating interference as noise*, IEEE Transactions on Information Theory, 61 (2015), pp. 1753–1767.

[10] I. GOODFELLOW, Y. BENGIO, AND A. COURVILLE, *Deep learning*, MIT press, 2016.

[11] N. U. HASSAN, C. YUEN, S. SAEED, AND Z. ZHANG, *Power control for sum-rate maximization on interference channels under sum power constraint*, IEEE Transactions on Vehicular Technology, 64 (2014), pp. 593–609.

[12] M. HONG AND Z.-Q. LUO, *Signal processing and optimal resource allocation for the interference channel*, in Academic Press Library in Signal Processing, vol. 2, Elsevier, 2014, pp. 409–469.

[13] E. HOSSAIN, M. RASTI, AND L. B. LE, *Radio resource management in wireless networks: an engineering approach*, Cambridge University Press, 2017.

[14] L. HUI, L. CHUN, AND L. HUA, *Power control based on non-cooperative game in wireless sensor networks*, in 2013 6th International Conference on Intelligent Networks and Intelligent Systems (ICINIS), IEEE, 2013, pp. 135–138.

[15] P. JAIN AND P. KAR, *Non-convex optimization for machine learning*, arXiv preprint arXiv:1712.07897, (2017).

[16] L. P. KAELBLING, M. L. LITTMAN, AND A. W. MOORE, *Reinforcement learning: A survey*, CoRR, cs.AI/9605103 (1996).

[17] A. KARIMIPOUR, S. A. BAGHERZADEH, A. TAGHIPOUR, A. ABDOLLAHI, AND M. R. SAFAEI, *A novel nonlinear regression model of svr as a substitute for ann to predict conductivity of mwcnt-cuo/water hybrid nanofluid based on empirical data*, Physica A: Statistical Mechanics and its Applications, 521 (2019), pp. 89–97.

[18] F. LIANG, C. SHEN, W. YU, AND F. WU, *Power control for interference management via ensembling deep neural networks*, in 2019 IEEE/CIC International Conference on Communications in China (ICCC), IEEE, 2019, pp. 237–242.

[19] Z.-Q. LUO AND S. ZHANG, *Dynamic spectrum management: Complexity and duality*, IEEE journal of selected topics in signal processing, 2 (2008), pp. 57–73.

[20] A. F. MOLISCH, *Wireless communications*, John Wiley & Sons, 2012.

[21] N. NADERIALIZADEH, *Contrastive self-supervised learning for wireless power control*, in ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2021, pp. 4965–4969.

[22] C. A. NAVARRO, N. HITSCHFELD-KAHLER, AND L. MATEU, *A survey on parallel computing and its applications in data-parallel problems using gpu architectures*, Communications in Computational Physics, 15 (2014), pp. 285–329.

[23] K. PAHLAVAN, *Wireless intraoffice networks*, ACM Transactions on Information Systems (TOIS), 6 (1988), pp. 277–302.

[24] N. PAPANDREOU AND T. ANTONAKOPOULOS, *Bit and power allocation in constrained multicarrier systems: The single-user case*, EURASIP Journal on Advances in Signal Processing, 2008 (2007), pp. 1–14.

[25] J. PAPANDRIOPOULOS AND J. S. EVANS, *Scale: A low-complexity distributed protocol for spectrum balancing in multiuser dsl networks*, IEEE Transactions on Information Theory, 55 (2009), pp. 3711–3724.

[26] Q. QI, A. MINTURN, AND Y. YANG, *An efficient water-filling algorithm for power allocation in ofdm-based cognitive radio systems*, in 2012 International Conference on Systems and Informatics (ICSAI2012), IEEE, 2012, pp. 2069–2073.

[27] L. P. QIAN, Y. J. ZHANG, AND J. HUANG, *Mapel: Achieving global optimality for a non-convex wireless power control problem*, IEEE Transactions on Wireless Communications, 8 (2009), pp. 1553–1563.

[28] Z. ROSBERG AND J. ZANDER, *Toward a framework for power control in cellular systems*, Wireless Networks, 4 (1998), pp. 215–222.

[29] M. SCHUBERT AND H. BOCHE, *QoS-based resource allocation and transceiver optimization*, Now Publishers Inc, 2006.

[30] X. SHANG, B. CHEN, G. KRAMER, AND H. V. POOR, *Noisy-interference sum-rate capacity of parallel gaussian interference channels*, IEEE Transactions on Information Theory, 57 (2010), pp. 210–226.

[31] Y. SHEN, Y. SHI, J. ZHANG, AND K. B. LETAIEF, *A graph neural network approach for scalable wireless power control*, in 2019 IEEE Globecom Workshops (GC Wkshps), IEEE, 2019, pp. 1–6.

[32] Q. SHI, M. RAZAVIYAYN, Z.-Q. LUO, AND C. HE, *An iteratively weighted mmse approach to distributed sum-utility maximization for a mimo interfering broadcast channel*, IEEE Transactions on Signal Processing, 59 (2011), pp. 4331–4340.

[33] H. SUN, X. CHEN, Q. SHI, M. HONG, X. FU, AND N. D. SIDIROPOULOS, *Learning to optimize: Training deep neural networks for wireless resource management*, in 2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), IEEE, 2017, pp. 1–6.

[34] R. S. SUTTON, D. MCALLESTER, S. SINGH, AND Y. MANSOUR, *Policy gradient methods for reinforcement learning with function approximation*, Advances in neural information processing systems, 12 (1999).

[35] N. C. THOMPSON, K. GREENEWALD, K. LEE, AND G. F. MANSO, *The computational limits of deep learning*, arXiv preprint arXiv:2007.05558, (2020).

[36] U. B. TRIVEDI AND P. MISHRA, *Accelerating stochastic gradient descent by minibatching, learning rate annealing and momentum*, in International Conference on Futuristic Trends in Networks and Computing Technologies, Springer, 2020, pp. 247–255.

[37] P. TSIAFLAKIS, M. DIEHL, AND M. MOONEN, *Distributed spectrum management algorithms for multiuser dsl networks*, IEEE Transactions on Signal Processing, 56 (2008), pp. 4825–4843.

[38] W. YU, G. GINIS, AND J. M. CIOFFI, *Distributed multiuser power control for digital subscriber lines*, IEEE Journal on Selected areas in Communications, 20 (2002), pp. 1105–1115.

[39] J. ZANDER, *Performance of optimum transmitter power control in cellular radio systems*, IEEE transactions on vehicular technology, 41 (1992), pp. 57–62.

[40] J. ZANDER AND S.-L. KIM, *Radio Resource Management for Wireless Networks*, Artech House, 2001.

Appendix

Software Used in Final Year Projects

Filename/Algorithm/ Package	Supplier/Source/Author/ website	Use/Modifications made/ Student written
Pytorch	Pytorch.org	The machine learning package
Vanilla Policy Gradient	https://spinningup.openai.com/en/latest /algorithms/vpg.html	Student's coding from the referred Pseudocode
Weighted Minimum Mean Square Error	https://github.com/navid-naderi/ContrastiveSSL_WirelessPower Control	Written by Navid from University of Pennsylvania, Student used it for baseline algorithm comparison
OpenAI Gym	OpenAI	RL environment establishment library by OpenAI
Gradient Descent with Random Restarts	https://github.com/duanwenbo/ug_proj ect	Student's own algorithm and coding
VPG training Environment	https://github.com/duanwenbo/ug_proj ect	Student's own coding the MDP design based on OpenAI Gym Library