Master's thesis

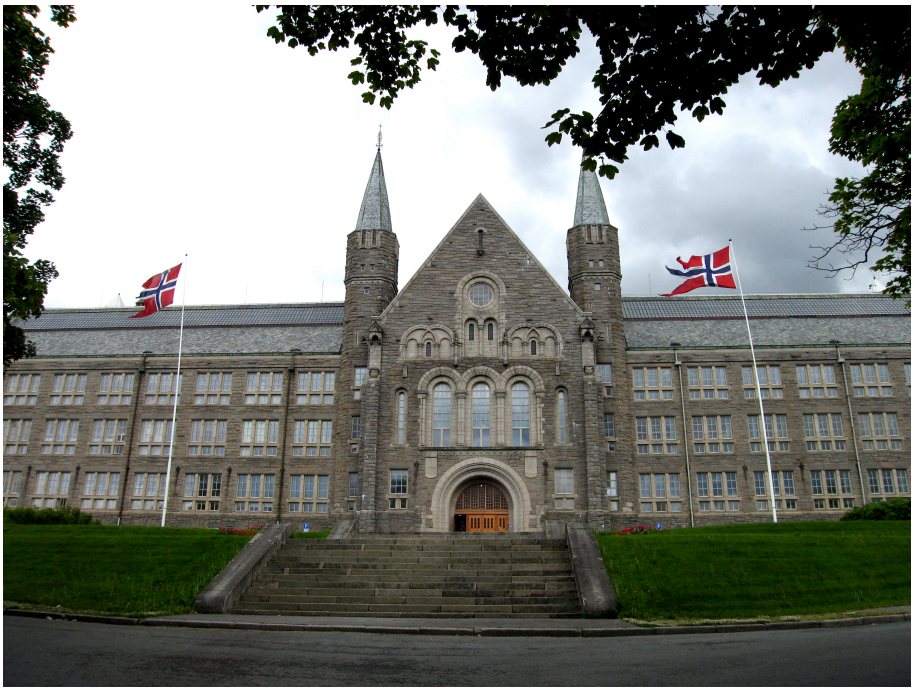
NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Thomas Aleksander Frekhaug

Safe Reinforcement Learning using Model Predictive Control

An analysis of utilising anisotropic exploration with deterministic policy gradients

Master's thesis in Cybernetics
Supervisor: Sebastien Gros
June 2020



NTNU
Norwegian University of Science and Technology

Thomas Aleksander Frekhaug

Safe Reinforcement Learning using Model Predictive Control

An analysis of utilising anisotropic exploration with deterministic policy gradients

Master's thesis in Cybernetics
Supervisor: Sebastien Gros
June 2020

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Abstract

This thesis is a study in the utilisation of anisotropic exploration in safe Reinforcement Learning (RL). Safe RL is a domain where the goal is to develop algorithms that may learn optimal policies while simultaneously ensuring that certain safety constraints are respected during the training process. Safety constraints limit the possible exploration space of a policy, and it inevitable that any isotropic exploration schemes must be morphed, giving anisotropic exploration.

The thesis considers the predictive controller algorithms Linear Quadratic Regulator (LQR) and Nonlinear Model Predictive Controller (NMPC) as function approximators in the actor-critic policy gradient method. These approximators form an estimate of the performance gradient as given by the deterministic policy gradient under an anisotropic exploration scheme.

The estimated performance gradient under anisotropic exploration is the focus of the research in the thesis. An analytical evaluation of the estimated performance gradient yielded an estimate purely dependent on the state trajectory. From this, a modified function approximator became apparent, and it is shown that the resulting estimated performance gradient converges to the true performance gradient, regardless of anisotropic exploration.

Further analysis of the estimated performance gradient yielded two bounds on a potential error. The first bound on the error was established by means of calculating the relative error between the estimated and true performance gradient and concludes that the estimate is bounded within a relative factor of 2 from the true performance gradient. The second bound was found through the means of a Taylor approximation and shows that any error from the estimate is strictly proportional with the covariance of the state trajectory and to the curvature of the estimate. Furthermore, under conditions and arguments from the delta method, the error of the estimate is shown to be reduced to zero.

Experiments using a linear system and quadratic reward function were performed, and they support the theoretical bounds provided by the theory, but also suggest that the estimated performance gradient is an exact estimate.

Sammendrag

Denne avhandlingen er en studie i Trygg Forsterkende Læring (eng: Reinforcement Learning, RL) der det blir benyttet utforskningsagenter (eng: policy) som bruker anisotropisk utforskning. Trygg RL er en ny kategori av RL hvor målet er å utvikle algoritmer som kan lære seg en optimal handlemetode samtidig som at tilstandene i systemet aldri bryter noen spesifikke begrensinger under treningsperioden. Dette begrenser derimot utforskningsmulighetene til agenten, og det er, under trygg RL, uunngåelig at utforskningen til agenten blir anisotropisk.

Avhandlingen tar også for seg to kontrollerne som hvis formål er å approksimere enten agenten eller verdifunksjonene. De to kontrollerne som blir brukt er den Lineære-Kvadratiske regulatoren (eng: Linear Quadratic Regulator , LQR), og den Ulineære Modell-Prediktive Regulatoren (eng: Nonlinear Model Predictive Control, NMPC).

Hoveddelen av avhandlingen tar for seg en analytisk evaluasjon av konsekvensene ved anisotropisk utforskning når en NMPC brukes som en agent samt ved bruk av aktør-kritikk (eng: Actor-Critic) metoden for å approksimere prestasjonsgradienten (eng: Performance gradient). Fra denne analysen, så kommer det klart fram at det er mulig å garantere konvergens av den estimerte prestasjonsgradienten, uavhengig av kovariansen til utforsknings algoritmen, ved å gjøre en liten endring i den kompatible funksjons approksimatoren (eng: Compatible Function Approximator).

Videre ble det funnet to begrensninger på eventuelle feil ved bruk av den estimerte prestasjonsgradienten under anisotropisk utforskning; Ved å bruke relativ feil, så kommer det fram at den estimerte prestasjonsgradienten er øvre begrenset i forhold til den faktiske prestasjonsgradienten med en faktor på 2. Videre, ved bruk av en Taylor ekspansjon, så ble det funnet at en eventuell feil vil gå mot null om kovariansen til tilstandsutviklingen går mot null over tid. Det ble også vist at en eventuell feil vil være maksimalt proporsjonal til kovariansen i tilstandsutviklingen.

Lineære kvadratiske eksperimenter ble gjennomført, der alle samsvarer med teorien som har blitt utviklet i avhandlingen. Videre så indikerer eksperimentene at, i et lineært kvadratisk system, så er den estimerte prestasjonsgradienten eksakt, men dette var det derimot ikke grunnlag for å teoretisk bekrefte.

Preface

This master thesis is submitted as the final requirement for completing a masters degree at the Department of Engineering Cybernetics at the Norwegian University of Science and Technology in Trondheim.

This thesis considers Safe Reinforcement Learning, a new domain of Artificial Intelligence that may have huge potentials in the near future. It discusses the combination of state-of-the-art Reinforcement Learning together with state-of-the-art control engineering, a combination which may one day bring the darkness and uncertainty of Reinforcement Learning into the analytical and observable world of control engineering.

This thesis is the culmination of five incredible years of hard work and experiences, of which 6 years ago I ought not obtainable by me. With this, I would like to extend my appreciations to the many people I had the chance to enjoy a students life with. I would also furthermore appreciate the many technical student organizations at NTNU, especially Revolve NTNU, whom presented me with both challenges and experiences that drew out the best in me.

I would like to thank my family, girlfriend and my friends for their support during my frequent isolations and disappearances in Trondheim. Furthermore, I want to appreciate the inputs provided by father Tore and my brother Christer for this thesis.

As a final remark, I would also like to especially thank my middle school teachers Marita and Terje for being the fantastic teachers that they were.

Thomas Aleksander Frekhaug

Trondheim, June 2020

Contents

List of Figures

List of Tables

Abbreviations

Notation	Description	Page List
CLT	Central Limit Theorem.	22, 40
IVT	Implicit Value Theorem.	29
KKT	Karush Kuhn Tucker.	6, 18, 19
LQR	Linear Quadratic Regulator.	i, 3, 6, 16–18, 28, 30
MDP	Markov Decision Process.	7, 8
ML	Machine Learning.	4, 7
NMPC	Nonlinear Model Predictive Controller.	i, 3–6, 16, 18, 19, 23, 24, 26–33, 37, 51, 53
RL	Reinforcement Learning.	i, 1–7, 9–11, 14, 16, 23–27, 29, 31, 32, 37, 41, 51, 53, 54

Notation	Description	Page List
SVD	Singular Value Decomposition.	36, 40
TD	Temporal Difference.	11, 14

Glossary

Notation	Symbol	Description	Page List
ϵ-greedy	π_θ^ϵ	One specific policy, achieved by selecting the best action based upon the values in the action value function.	9–11
action-value function	$Q(s, a)$	The action-value function is the total discounted future reward when in a given state s,taking an action a, and then subsequently following the value functions corresponding policy.	xiii, 9–13, 24–26
actor-critic	N/A	A policy gradient algorithm that uses value function approximations in order to aid the policy improvement.	i, 12–14, 28, 29, 31, 34
advantage function	$A(s, a)$	The advantage function are the difference between the value and action value functions, and describes the advantage of taking one action above the other.	ii, xiii, 10, 12, 13, 34, 36, 40, 44
performance objective	$J(\pi_\theta)$	The performance of a specific policy π_θ.	9, 10, 12, 13, 26, 32
policy	π	A policy is either a stochastic or deterministic probability of choosing an action a.	8–12, 14, 24, 28

Notation	**Symbol**	**Description**	**Page List**
policy gradient	N/A	A class of reinforcement learning algorithms that attempts to find a policy that maximizes or minimizes the performance objective.	12, 24, 35, 40, 51, 53
Q-learning	N/A	A common and useful reinforcement learning algorithm. Based upon instantaneous temporal difference rewards.	6, 7, 10–12, 14, 24–27, 53
reward	$r(a_t, s_t, s_{t+1})$	Either the reward or penalty an actor receives from applying one action in a given state.	8, 9, 28, 34
transition probability	$\mathbb{P}(s_{t+1}\|s_t, a_t)$	The probability of transitioning to the next state s_{t+1}.	8
value function	$V(s)$	The Value function is the total discounted future reward when in a given state s and following the value functions corresponding policy.	9, 10, 14, 25, 26, 35

Nomenclature

θ	Parametric weights for policy	$\theta \in \mathbb{R}^m$
a	Actions to be taken by agent,	$a \in \mathbb{R}^{n_a}$
s	Current state of the environment,	$s \in \mathbb{R}^{n_s}$
x	Predicted future of enviroment,	$x \in \mathbb{R}^{n_s \times N}$
u	Predicted future inputs,	$u \in \mathbb{R}^{n_a \times N}$
$r(s, a)$	Sometimes denoted L, it is the instantaneous reward,	scalar
γ	The discounting in the performance objective,	scalar, $\gamma \in [0, 1]$
Σ	Standard deviation for the exploration policy,	$\Sigma \in \mathbb{R}^{n_a \times n_a}$
ν	Parametric weights for the baseline function V_ν,	$\nu \in \mathbb{R}^{m_\nu}$
ω	Parametric weights for critic,	$\omega \in \mathbb{R}^{m_\omega}$
$\pi_\theta(s)$	policy probability density,	$\pi_\theta \in \mathbb{R}^{n_a}$
$\nabla_\theta \pi_\theta(s)$	Sensitivites of the parametric weights in the agent,	$\nabla_\theta \pi_\theta \in \mathbb{R}^{m \times n_a}$
$\nabla_a Q_{\pi_\theta}(s, a)$	Sensitivites of actions in the action-value function,	$\nabla_a Q_{\pi_\theta} \in \mathbb{R}^{n_a}$
$\nabla_a A_{\pi_\theta}(s, a)$	Sensitivites of actions in the advantage function,	$\nabla_a A_{\pi_\theta} \in \mathbb{R}^{n_a}$

Introduction

This thesis was written at the Institute of cybernetics at the Norwegian University of Science and Technology and covers primarily topics in RL and predictive controllers. The thesis comprises four chapters in total, where this introductory chapter is the leading chapter and seeks to introduce the reader to the thesis. The second chapter aims to present the necessary background theory in the topics that will be covered in the third chapter. The penultimate third chapter is the main chapter of the thesis and covers the research performed by this study. Finally, a brief review of the theory and results is presented in the final chapter.

This introductory chapter initiates by presenting the motivation for thesis and the domain of which this thesis is a part of. The research objective of this study is presented in the second section, together with two precise research questions that will be investigated in the latter parts of the thesis. The introductory chapter finalises with a complete outline of the thesis' structure.

1.1 Motivation

Safe RL is a new approach to RL that, when properly established, may expand the possibilities of RL algorithms extensively. Garcia et al. defined in [1] safe RL as

> *" ...the process of learning policies that maximize the expectation of the return in problems in which it is important to ensure reasonable system performance and/or respect safety constraints during the learning and/or deployment processes".*

Looking purely at the first part of this definition, it is a simple description of any RL algorithm. The goal of the RL algorithm is to always find a solution that ensures the maximum amount of reward possible. Such a goal may be a dangerous one in a real-life setting. There are many obstacles and hindrances that must be obeyed and prioritized above the maximization of some reward. One cannot design a car that drives the most efficient path, if that is at the cost of the car never stopping for pedestrians.

RL algorithms are typically trained in simulated environments, where safety concerns may be considered merely as a penalized act. In these simulated environments, it is possible for RL algorithms to experience the consequences of breaking the safety constraints. However, it is difficult to enumerate and model all safety critical hindrances, and RL algorithms that are trained in simulations may encounter situations in real-life not considered in the simulations. Importantly, it is typically much simpler to define a safe set of behaviour than it is to define all the unsafe behaviours.

This leads into the second part of the definition by Garcia et al. A safe RL algorithm must always be able to obey certain safety constraints, regardless of it being trained or deployed. Different methods for achieving this are still in its research infancy. If it were possible to a priori state the outcome of a small change of parameters in the RL algorithm, then implementing safe RL could be simply the limiting of such parametric changes. Such an approach is similar to the domain of adaptive control, where algorithms improve the performance of controllers, while simultaneously guaranteeing that certain safety constraints are never broken. These are algorithms that intertwine with state-of-the-art controllers in order to improve or adapt their performance to its environment. Adaptive control is mentioned here as an alternative to RL that may serve similar approaches.

The primary challenge for a priori determining the outcome of certain parametric changes in RL algorithms boils down the most typical implementation of the algorithms. The most common approach to RL, and indeed other Artificial Intelligence algorithms, is to utilise Neural Networks as its "main body". A neural network, visualised in Figure 1.1, is a set of nodes that each have a parametric weight associated with it. These nodes augment the input it receives in a manner defined by its parametric weight. The complete network works by simply

propagating an input signal through many nodes, giving an output signal on the other side. The parametric weights for each node is then altered iteratively to improve the overall performance. With a small set of nodes, it may be possible to completely define what any parametric changes may introduce to the system. However, these networks are commonly too large for any such specific knowledge, and the networks are typically just considered as a black box.

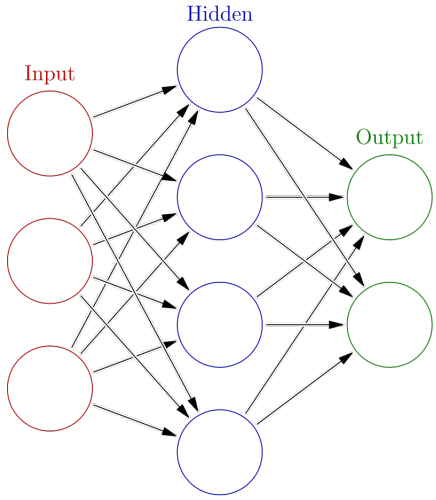


Figure 1.1: Basic concept of a neural network with the hidden neurons in the centre

As mentioned previously, adaptive controllers augment the parameters of a controller in a way to improve its performance. One approach to safe RL may be to utilize RL in a similar fashion as adaptive controllers. It is shown in [2] that RL algorithms may work in synergy with predictive controller algorithms as LQR and NMPC. The predictive controller algorithms ensure that the system is always inside the environments safe set, while the RL algorithms improve the controllers' performance in said environment. If properly implemented, such a safe RL algorithm may be trained on-line in real environments. However, some challenges that must be overcome still persist, and one specific challenge is that of exploration.

Any RL algorithm must be able to explore its environment, and the theory for established RL suggest that it has to have unrestricted exploration, i.e. the exploration behaviour must be isotropic. This is not possible in neither safe RL nor with predictive controllers limiting the RL algorithms exploration. The exploration must be restricted, and the consequences are anisotropic exploration behaviour. This thesis is primarily an investigation in the consequences of such anisotropic exploration, and if the RL algorithm is able to improve the performance under anisotropic circumstances.

1.2 Research objective

This goal for this thesis is to further the research within the context of safe reinforcement learning. It aims to investigate how predictive control algorithms might be utilized in synergy with RL algorithms. Furthermore, it endeavours to produce results that might be beneficial for the future of safe RL. Formalized, the research objective of this thesis is to:

> *Look into the combination of predictive control algorithms together with RL, and to research some of the challenges that face safe RL.*

The research objective is divided into two specific research questions in order to structure both the research and the thesis itself:

1. How are NMPC integrated with RL?
    - How to use NMPC as a value function approximator for Q-learning?
    - How to use NMPC as a policy in the policy gradient method?
2. What are the consequences of using anisotropic exploration in the context of NMPC and RL?
    - Does the approximate policy gradient depend on the shape of the exploration?

Safe RL consists of state-of-the-art algorithms within both Machine Learning (ML) and control engineering. It is important to get a proper grasp on presented work, and as such, the first part of the research in this thesis is dedicated to a literature study with the end goal given as the first research question. The first research question is also in line with the first part of the research objective "*To look into the combination of predictive control algorithms together with RL*".

The second research question tackles the second part of the research objective and is looking at a key element of RL, namely exploration policies. The purpose of the second question is to analytically investigate the anisotropic exploration policies and attempt to bound the consequences of such policies.

1.3 Thesis outline

This thesis consists of 4 chapters divided into several sections. The first chapter is the current introductory chapter that provides motivation and the goals for the thesis. It also presents the research objective and goals. The second chapter is the literature and background studies that present basic key concepts. In this chapter, the basics of RL and NMPC will be thoroughly presented, along with other essential material that is necessary for the later chapters. The two leading sections in chapter three will be directed at the investigation of the research goals, where each of the two sections will pertain its own research question as a whole. The third chapter concludes by presenting some experiments and discussions about the results. The fourth and final chapter is a summary of the thesis, together with concluding remarks and discussions about further work around the research objective.

II Basic Concepts

This chapter entails three sections that each aims to present some fundamental concepts in the three categories of Reinforcement Learning (RL), control theory, and statistics. The chapter is presented as such to enable the reader to get a rudimentary understanding of the primary concepts that is to be used in the next chapter.

The first section discusses RL and starts by presenting some important notations that are fundamental in the domain of RL. The section proceeds further by introducing the most basic RL algorithm Q-learning. A brief discussing on some subtle differences between stochastic and deterministic policies are mentioned before the section continues by presenting the deterministic policy gradient. The first section finalises by discussing the least squares solution for linear equations.

The second section presents the predictive controllers that will be utilised later on in this thesis. This section starts by presenting the linear-quadratic system before presenting an optimal solver in the form of the Linear Quadratic Regulator (LQR). The second predictive method, a generalisation of the LQR, the Nonlinear Model Predictive Controller (NMPC) is then introduced before the section finalises by briefly discussing the Karush Kuhn Tucker (KKT) conditions which are essential in order to calculate the sensitivities of the NMPC scheme.

The final section is a mathematical section that describes some statistical tools and results that will be used in the analytical evaluation performed in chapter three. Primarily, it discusses the expected value and its generalisation to moments. It also describes how to extend these moments of variables to encompass vectors and matrices. Finally, the Taylor expansion of moments of functions is detailed, in addition to its relationship to the delta method.

2.1 Reinforcement Learning

This thesis tackles theoretically complex problems, and proper fundamentals are essential for further review. This section serves to introduce fundamental concepts in RL that should familiarise the reader with the subject. It primarily considers background material specialised towards the applications for this thesis. Therefore, there might be some definitions and assumptions given here that may be of importance for later results (i.e., type of performance indicator).

This section starts by presenting the key concepts for the RL algorithms used in this thesis. With the concepts laid out, the section then presents one of the more basic RL algorithm Q-learning, which will be used later. It briefly touches upon some small differences between stochastic and deterministic policies before properly presenting policy gradient methods. It finalises by shortly discussing some least-squares solutions that are essential for later results. The following section is primarily based upon the book *Reinforcement Learning: An introduction*, [3]

2.1.1 Background

RL is a subcategory of Machine Learning (ML) that has a vast application space. It is a class of algorithms that learn more analogous to human beings, in opposition to its counterparts in the class of ML. The working principles of RL, summarised in Figure 2.1, are as follows:

An agent is placed in an environment. The agent is aware of the current state of the environment s_t, and has a set of actions, called policies $\pi_k(s_t)$, it may deploy. It chooses some action a_t and applies it to the environment. The environment's state s_t is updated to the next state, and the agent receives some reward $r(s_t, a_t)$ based on the performance of the action. From this reward, the agent updates its internal parameters in order to tweak its policy to be better than the previous policy $\pi_{k+1} > \pi_k$.

The analogy to human learning behaviour becomes clear if one replaces the agent with a human, and the rewards with sensory inputs. This analogy is practical as it provides a reasonable method of understanding the principles of how the algorithms learn.

2.1.2 Basics Concepts and Notation

RL algorithms may be described in two parts: Its environment and the agent itself. The environment is usually described as a Markov Decision Process (MDP). The MDP is a discrete-time stochastic method for modelling decision processes of partially random systems, and may, for a discrete system, be summarised in the following four elements:

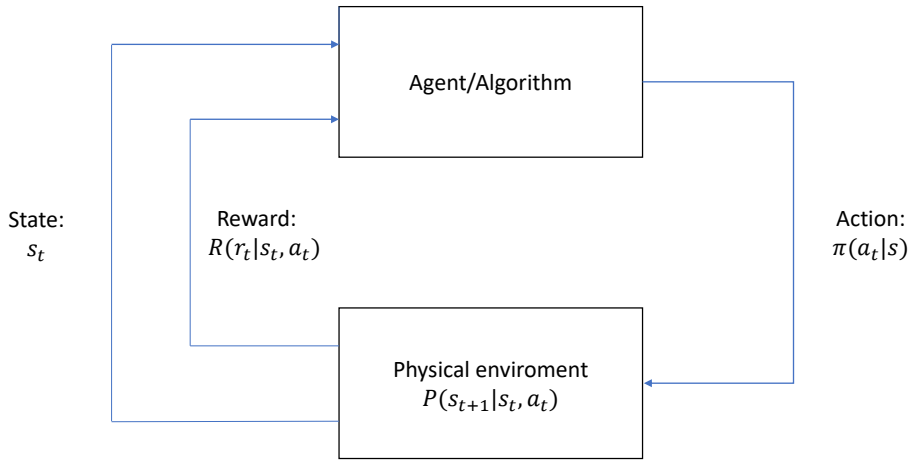


Figure 2.1: Classical feedback loop visualisation of RL

1. $\mathcal{S}$: A set of all possible states
2. $\mathcal{A}$: A set of all possible actions
3. $\mathbb{P}(s_{t+1}|s_t, a_t)$: The transition probability of reaching state s_{t+1}, given state s_t and action a_t
4. $r(s_t, a_t)$: The immediate reward when taking action a_t in state s_t

An equivalent representation of the environment that is more common in control engineering is that of the state space representation. The MDP share a couple of elements in common with the state space representation, namely the sets of possible states and actions. In addition, state progression and immediate rewards are commonly noted as

$$x_{t+1} = f(x, u, \xi), \quad L(x_t, u_t), \tag{2.1.1}$$

where ξ denotes the randomness in the system. In addition, both f and L may be nonlinear. In this thesis, the state space representation will be preferred. However, the MDP representation has its nuances, and it will be used where appropriate.

Agent and Policy

The agent uses its policy as its method of asserting its presence in the environment. A policy is typically a stochastic function where $\pi(s_t)$ denotes the probability density of choosing any possible action in the action space $\mathcal{A}$. Whereas $\pi(s_t)$ and $\pi(a_t|s_t)$ represents the densities and probabilities, a_t refers to a specific action in the state s_t. The final part of the actor is its internal parameters

θ. These are parameters that tweak and change the policy π and are the sole method of improving the agent's performance. The agents' policy is described by its internal parameters θ and is denoted π_θ. The policies may have many different shapes and forms, but one common policy is that of the ϵ-greedy policy π_θ^ϵ. This policy is derived from selecting an action in the action-value function that yields the best value.

$$\pi_\theta^\epsilon = \arg\min_a Q_{\pi_\theta}(s_t, a_t). \tag{2.1.2}$$

Reward Function and Performance Objective

The reward function $r(a_t, s_t, s_{t+1})$ is a part that enables the actor to perceive the difference between good and poor performance. The reward function also defines the optimal solution, even though this solution is not apparent. The design of the reward function is therefore a critical part when designing the RL algorithm. The performance of the actor is also indicated through the reward function as the performance objective $J(\pi_\theta)$. The performance objective can have some different shapes and designs [4], depending on the type of system. For this thesis, the performance objective that is of interest is that of a discounted episodic performance objective,

$$J(\pi_\theta) = \mathbb{E}[\sum_{t=1}^{N} \gamma^{t-1} r(a_t, s_t, s_{t+1})]. \tag{2.1.3}$$

The interpretation of this performance objective is that it is the sum of all accumulated discounted reward in the future. The discount γ is a value that enables the prioritisation of long- or short-term rewards. It is essential in stochastic systems, as such a system will never reach a steady state. They will continue to accumulate rewards in infinite time. Discounting ensures that the performance objective will stabilise at some value. The discounting will affect the optimal solution, so it is also a design variable. The performance objective defines the goal of any RL algorithm, and if $r(a_t, s_t, s_{t+1})$ is defined as a reward, then the goal would be to maximise the amount of reward and therefore also maximise the performance objective. Similarly, in the case of $r(a_t, s_t, s_{t+1})$ indicating penalties, the goal would be to minimise the performance objective.

Value Functions

With the environment, agents, and reward functions properly described, the next concept to be presented is value functions. These represent the value of being in some specific state in the environment. There are two basic value functions, typically (and confusingly) named Value function $V(s)$, and Action-value function $Q(s, a)$. The value function (eq 2.1.4) is the discounted value of being in the current state s_t, and only selecting actions based upon the current policy π_θ for all future. Similarly, the action-value function (eq 2.1.5) is the discounted value of being in the current state s_t, immediately applying some action a_t, before thereafter selecting actions based upon the current policy for all future. A formal

definition of the value functions may be found in [3]. In this thesis, it is the Bellman equations of the value functions that are of interests, which are defined as:

Definition 2.1.1. Bellman value functions: The Bellman equations for the value function and action-value function are given as [3]

$$V_{\pi_\theta}(s_t) = \mathbb{E}_{a\sim\pi_\theta}[r(s_t, a] + \gamma\mathbb{E}_{s\sim\rho}[V(s_{t+1}|\pi_\theta)] \tag{2.1.4}$$
$$Q_{\pi_\theta}(s_t, a_t) = r(s_t, a_t) + \gamma\mathbb{E}_{s\sim\rho}[V(s_{t+1}|\pi_\theta)]. \tag{2.1.5}$$

ρ is the distribution for the state trajectory $\mathcal{S}_n$, $\rho = \int \sum_{t=0}^{\infty} \gamma^t p(s_0)p(s_t|s_0, t, \pi_\theta)ds_o$ [5] . The state trajectory is simply the sequence of states s_t encountered during training. π_θ is the distribution of the policy.

There exists at least one optimal policy, denoted $\pi^*_\theta(s)$, such that it performs better than all other policies. The value functions for this policy are typically named optimal value functions.

Definition 2.1.2. Optimal Value functions: The optimal value functions and the relationships with the optimal policy may be written through the Bellman equations as [3]

$$V^*_{\pi_\theta}(s_t) = \mathbb{E}_{a\sim\pi^*_\theta}[r(s_t, a)] + \gamma\mathbb{E}_{s\sim\rho}[V^*_{\pi_\theta}(s_{t+1}|\pi^*_\theta)] \tag{2.1.6}$$
$$Q^*_{\pi_\theta}(s_t, a_t) = r(s_t, a_t) + \gamma\mathbb{E}_{s\sim\rho}[V^*_{\pi_\theta}(s_{t+1}|\pi^*_\theta)] \tag{2.1.7}$$
$$\pi^*_\theta = \arg\min_a Q^*_{\pi_\theta}(s_t, a_t). \tag{2.1.8}$$

Another important value function is that of the difference between the action-value function and value function, the Advantage function $A(s, a)$ (2.1.9). The advantage function represents the advantage of choosing one action above another. It is always either positive or negative, depending on if the performance objective is that of a maximising or minimising goal.

Definition 2.1.3. Advantage functions: The advantage of utilising one specific action instead of the policy may be quantified through the advantage function.

$$A_{\pi_\theta}(s_t, a_t) = Q_{\pi_\theta}(s_t, a_t) - V_{\pi_\theta}(s_t). \tag{2.1.9}$$

2.1.3 TD and Q-learning

With the basics concepts properly described, it is time to move on to the first RL algorithm that will be of use in this thesis, Q-learning. Q-learning is an algorithm that utilises a function approximator $\hat{Q}_{\pi_\theta}(s, a)$ that attempts to approximate the optimal action-value function $Q_{\pi_\theta}(s, a)$. From this approximation, the agent utilises the ϵ-greedy policy, which consists of choosing an action as $\pi^\epsilon_\theta = \arg\min_a \hat{Q}_{\pi_\theta}(s, a)$. The function approximator may be anything from tabular methods in discrete approximation schemes to linear and neural networks

in continuous approximation schemes. It relies on estimating the optimal action-value function, typically by using gradient descent in order to minimise the mean squared error $\mathbb{E}[(Q_{\pi_\theta} - \hat{Q}_{\pi_\theta})^2]$ [3]. The solution, as given by the gradient descent, may be found as

$$\delta = r + \min_{a_{t+1}} \hat{Q}_{\pi_\theta}(s_{t+1}, a_{t+1}) - \hat{Q}_{\pi_\theta}(s_t, a_t), \tag{2.1.10}$$

$$\theta_{k+1} = \theta_k + \alpha \mathbb{E}_{s\sim\rho}[\delta \nabla_\theta \hat{Q}_{\pi_\theta}(s_t, a_t)], \tag{2.1.11}$$

where δ is known as the Temporal Difference (TD) error and represents the immediate error in the estimate. The policy in the classic Q-learning method is typically chosen as the ϵ-greedy policy and the internal parameters θ of the policy are also the parameters that parametrise the function approximator. Therefore, an improvement of the estimate $\hat{Q}$ also yields an improved policy π_θ.

A note on subscripts; in this thesis, the subscript k indicates a batch update. A batch is a collection of evaluations such that the expected value may be properly evaluated. The subscript t indicates immediate time changes and are typically constrained within each episode. One batch may consist of several hundreds of episodes.

It is also worth mentioning that there are other similar RL algorithms and concepts that are based upon the TD error, like the on-policy algorithm SARSA [6]. These algorithms and terms are not used here, so for the sake of brevity, they are excluded.

2.1.4 Stochastic vs deterministic policies

Before proceeding to the next algorithm, it is necessary to discuss the differences between a stochastic and deterministic policy. This distinction becomes crucial in the next subsection, as the policy gradient [7] is different in both cases. Previously, $\pi(s_t)$ has been used to describe the probability density of choosing an action in each state s_t. When the agent is in some state s_t, and it has a stochastic policy, it will not always choose the same action a_t, even if the action is derived from the policy. The probability that the agent selects a specific action a_t is chosen is given by $\pi(a_t|s_t)$. Such a policy is often not that beneficial in a controller setting, as a predictable outcome of the controller is preferred. A deterministic policy will typically be a much better choice in such a setting. Deterministic policies ensure that some specific action a_t is chosen deterministically when in a specific state s_t. The probability density $\pi(s_t)$ is therefore reduced to a Dirac Delta function centred at the optimal action a_t.

One problem encountered when utilising deterministic policies is that of exploration. As will be apparent in the next section, the deterministic policy gradient is dependent on choosing actions that differ from the policy. Therefore, it is typical that any deterministic policies are followed by a stochastic behaviour policy, denoted here as $\beta(s_t)$. This behaviour policy has some constraints, which will be discussed later.

2.1.5 Policy gradient

While Q-learning works well at tackling many problems, it has one disadvantage. It does not directly attempt to solve the problem; its primary goal is that of attempting to fit some function approximation to the optimal action-value function. The resulting policy may be interpreted as more of a side effect of what it is actually trying to do. A more hands down direct method would be to find a policy that either maximises or minimises, depending on the goal, the performance objective directly. Such methods are typically referred to as policy gradients and have some advantages above the value-function based algorithms.

Definition 2.1.4. Policy gradient: The deterministic policy gradient, introduced in [5], is a method of updating the policy parameters θ, and are given as

$$\theta_{k+1} = \theta_k + \alpha \mathbb{E}_{s\sim\rho}[\nabla_\theta J(\pi_\theta(s))], \tag{2.1.12}$$

where ρ represents the probability density for the state trajectory $\mathcal{S}_n$ [5], and J is the performance objective.

The policy gradient utilises the gradient ascent/descent optimisation technique applied to the performance objective, giving the policy parameter update law as (2.1.12). These algorithms also typically utilise value function approximations in order to improve the policies. One common policy gradient method to be discussed soon is the actor-critic method, visualised in Figure 2.2. This method updates its policy (agent) based upon the policy gradient, while also utilising value function approximation (critic) in order to improve these updates.

The performance gradient in (2.1.12) may furthermore be written as [5],

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\pi_\theta}} [\nabla_\theta \pi_\theta(s_t) \nabla_a Q_{\pi_\theta}(s_t, a_t)] \tag{2.1.13}$$

$$= \mathbb{E}_{\substack{s\sim\rho \\ a\sim\pi_\theta}} [\nabla_\theta \pi_\theta(s_t) \nabla_a A_{\pi_\theta}(s_t, a_t)], \tag{2.1.14}$$

where the definition of the advantage function (2.1.9) has been used, and $\nabla_\theta \pi_\theta$ are the first order sensitivities of the policy π_θ. One issue when utilising a deterministic policy is that of inadequate exploration. This may be solved through defining a separate exploration policy β that ensures sufficient exploration of the state space $\mathcal{S}$. This exploration policy is stochastic by nature and as such, the performance gradient must be additionally sampled across the distribution β.

Definition 2.1.5. Exploration policy: The exploration policy is the behaviour policy that is used in order to generate actions differing from the agent policy π_θ.

$$\beta : e = a - \pi_\theta(s), \quad a \sim \mathcal{N}(\pi_\theta, \Sigma\Sigma^\top) \quad \Rightarrow \quad e \sim \mathcal{N}(0, \Sigma\Sigma^\top), \tag{2.1.15}$$

Assuming that e results from a Normal distribution via a polynomially bounded function [5],[8], and that e constitutes an isotropic exploration scheme then the true performance gradient, as used in this thesis, may finally be defined:

Definition 2.1.6. True Performance gradient: The true performance gradient is the gradient of the performance objective, and are given as

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_a A_{\pi_\theta}(s,a)] \tag{2.1.16}$$

where ρ represents the probability density for the state trajectory $\mathcal{S}_n$ [5] and β represents the probability density of the exploration policy.

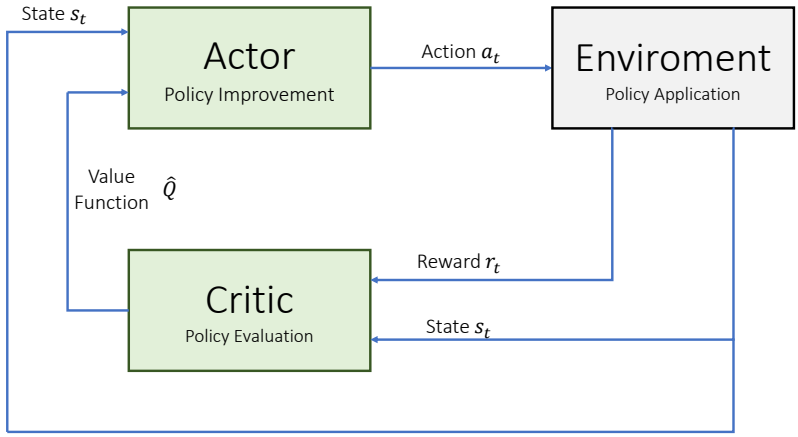


Figure 2.2: Feedback loop visualisation of the Actor Critic method

Actor-Critic

The policy gradient requires the gradient of the advantage function $\nabla_a A_{\pi_\theta}$ to be known; however, the advantage function is typically unknown, as it depends on the true environment and system dynamics. Consequently, in order to have a correct policy gradient, it must be estimated. This estimation is something that the actor-critic method introduces to the policy gradient. The critic in the actor-critic is an estimation of the advantage function in (2.1.16). However, instead of estimating the gradient $\nabla_a A_{\pi_\theta}$ directly, the critic estimates the advantage function A_{π_θ} function of which the gradient is subsequently calculated. Therefore, the estimation of A_{π_θ} must be such that it enables the direct calculation of the gradient $\nabla_a A_{\pi_\theta}$. A function approximator that entails this property is known as a compatible function approximator, and one such approximator is defined as:

Definition 2.1.7. Compatible Function Approximator: A compatible function approximator is any function approximator that preserves the true action-value function gradient[5]:

$$\hat{Q}_{\pi_\theta}(s,a) = (a - \pi_\theta(s))^\top \nabla_\theta \pi_\theta^\top \omega + \hat{V}_\nu(s), \tag{2.1.17}$$

where ω are the weights that parametrise the function approximator $\hat{Q}_{\pi_\theta}$, and $\hat{V}_\nu$ may be any baseline function that is not dependent on the action space $\mathcal{A}$.

Whereas $\hat{V}_\nu$ may be any function not dependent on the action a, for reasons specified later, it is advantageous that $\hat{V}_\nu$ is an approximation of the value function V_{π_θ}. The parameters ω of the critic $\hat{Q}_{\pi_\theta}$, and ν of the baseline function $\hat{V}_\nu$ may be learned through utilising the Q-learning algorithm discussed previously. It is important to note that, as $\hat{V}_\nu$ is an approximation of V_{π_θ}, ν is the solution to the minimisation problem $\mathbb{E}[(V_{\pi_\theta} - \hat{V}_\nu)^2]$, and the TD error is therefore slightly different. The parameter update laws for the critic and baseline functions are given as:

$$\omega_{k+1} = \omega_k + \alpha_\omega \mathbb{E}[(r + \gamma \hat{Q}_{\pi_\theta}(s_{t+1}, \pi_\theta(s_{t+1})) - \hat{Q}_{\pi_\theta}(s_t, a_t)) \nabla_\omega \hat{Q}_{\pi_\theta}(s_t, a_t)], \tag{2.1.18}$$

$$\nu_{k+1} = \nu_k + \alpha_\nu \mathbb{E}[(r + \gamma \hat{V}_\nu(s_{t+1}) - \hat{V}_\nu(s_t)) \nabla_\nu \hat{V}_\nu(s_t)]. \tag{2.1.19}$$

With the compatible function approximator, and the fact that $\nabla_a Q = \nabla_a A$, the estimated performance gradient for the actor-critic method may be defined:

Definition 2.1.8. Estimated Performance Gradient: The estimated performance gradient, as provided by the actor-critic method, is an estimate of the true performance gradient defined in 2.1.6, and is given as

$$\nabla_\theta \hat{J}(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_a \hat{Q}_{\pi_\theta}(s, a)] \tag{2.1.20}$$

$$= \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \omega. \tag{2.1.21}$$

It is shown in [5], that if the actions are chosen according to the exploratory behaviour policy β, that is $\beta : e \sim \mathcal{N}(0, \sigma\mathbb{I})$, then this estimated performance gradient is an exact estimate of the true performance gradient,

$$\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \hat{J}(\pi_\theta)] = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta J(\pi_\theta)]. \tag{2.1.22}$$

2.1.6 Least squares

The concluding part for the RL introduction is that of the least squares solution. The classical and straightforward approach towards the actor-critic method is visualised in a feedback loop style in Figure 2.2 and of which equations are presented in the previous subsection. This approach constitutes of initially getting a proper estimate of the critic, through the iterative update laws (2.1.18) and (2.1.19), before doing another pass and updating the policy parameters θ through the iterative policy gradient update law (2.1.12). However, the least squares solution allows simultaneously updating the critic and actor parameters, given that the function approximators are linear.

Inspection of the iterative equations (2.1.18) and (2.1.19) makes it clear that these updates are linear in the parameters ω and ν, given that $\hat{V}_\nu$ is a linear

function approximator. Also, these equations search for a stationary solution where $\omega_{k+1} = \omega_k$, which is equivalent to

$$0 = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[(r + \gamma\hat{Q}_{\pi_\theta}(s_{t+1}, \pi_\theta(s_{t+1}) - \hat{Q}_{\pi_\theta}(s_t, a_t))\nabla_\omega \hat{Q}_{\pi_\theta}(s_t, a_t)]. \tag{2.1.23}$$

Inserting the compatible function approximator $\hat{Q}_{\pi_\theta}$, using a linear function approximator $\hat{V}_\nu = \psi(s)^\top \nu$, and denoting $e = a - \pi_\theta$ yields

$$0 = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[(r + \gamma\psi(s_{t+1})^\top \nu - e^\top \nabla_\theta \pi_\theta^\top \omega - \psi(s_t)^\top \nu)\nabla_\theta \pi_\theta e], \tag{2.1.24}$$

which has the shape $0 = \mathbb{E}[A\omega] + \mathbb{E}[B]$. The least squares solution for ω may then be written as:

$$\omega = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e(r + \gamma\hat{V}_\nu(s_{t+1}) - \hat{V}_\nu(s_t))]. \tag{2.1.25}$$

A similar approach may be used in order to solve for ν. With this, the estimated performance gradient may be calculated as

$$\begin{aligned} \nabla_\theta \hat{J}(\pi_\theta) =& \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \\ & \cdot \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e(r + \gamma\hat{V}_\nu(s_{t+1}) - \hat{V}_\nu(s_t))], \end{aligned} \tag{2.1.26}$$

which makes for a more direct update law for the policy parameters θ.

2.2 Predictive Control Algorithms

Predictive control algorithms are algorithms that allow for the control of both simple and advanced systems. They are powerful tools that, when combined with proper model dynamics, may control the most complex and advanced systems. The key working principle behind these controllers is the controller's internal model. With a sufficiently accurate model, these controllers may predict any outcome of any input, and as such also choose an input that is in line with achieving a desired goal. This advantage is typically also one of the more considerable weaknesses of such controllers. An insufficient model results in a poor outcome. There are many methods that may negate issues caused by poor modelling and, as will be explored later, RL may become an additional method for this purpose.

This section will introduce some theory and concepts on the predictive controllers that will be used in this thesis. It will not explain these concepts extensively, and it is assumed that the reader has previous experience with the predictive control algorithms presented here. This section will present the two controllers LQR and NMPC in consecutive order. The focus of the section will be on the procedures and properties that will be used in chapter three.

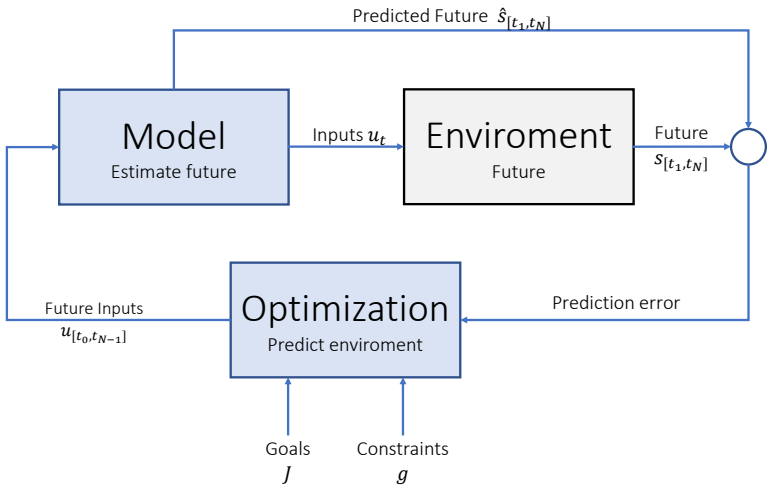


Figure 2.3: Feedback loop visualization of a generic predictive controller

2.2.1 Linear Quadratic Regulator

The LQR [9] is a simple controller that has a simple shape $u = -Ks$, where K is some feedback parameters. The LQR has an advantage above more regular controllers in the sense that it is possible to prioritise how fast the controller should work (size of the input u) versus how much deviations the states may have (size

of the error s). This prioritisation is performed by using a performance objective J, where J typically extends across a prediction horizon N. The LQR may be calculated merely from linear system dynamics and quadratic performance objective, thereby the name linear quadratic regulator.

$$x_{t+1} = Ax_t + Bu_t, \quad L(x,u) = \begin{bmatrix} x \\ u \end{bmatrix}^\top \begin{bmatrix} T & N \\ N^\top & R \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \tag{2.2.1}$$

$$J = \sum_{t=0}^{N} L(x_{t+1}, u_t). \tag{2.2.2}$$

For the above linear, quadratic system, the finite horizon LQR is simply the sequence of inputs u_N that minimises the performance objective $J = \sum_{t=0}^{N} L(x_{t+1}, u_t)$, in more mathematical terms:

$$\begin{aligned} u^* = \arg\min_{x,u} \quad & \sum_{t=0}^{N} L(x_{t+1}, u_t) \\ \text{s.t} \quad & \\ & x_{t+1} = Ax_t + bu_t, \quad t \in [0, N-1], \end{aligned} \tag{2.2.3}$$

where u^* is the optimal sequence of input for the system described by the dynamics, across the prediction horizon N. The performance objective is, as previously mentioned, quadratic and may be represented as a sum across the prediction horizon, $J(x,u) = \sum_{t=0}^{N} \frac{1}{2} x_{t+1}^\top Q_{t+1} x_{t+1} + \frac{1}{2} u_t^\top R_t u_t$. The time-dependent feedback parameters K_t may be calculated from the following recursive algebraic Ricatti equations [10]:

$$u_t = -K_t x_t \tag{2.2.4}$$

$$K_t = R_t^{-1} B_t^\top P_{t+1} (I + B_t R_t^{-1} B_t^\top P_{t+1})^{-1} A_t \tag{2.2.5}$$

$$P_t = Q_t + A_t^\top P_{t+1} (I + B_t R_t^{-1} B_t^\top P_{t+1})^{-1} A_t \tag{2.2.6}$$

$$P_N = Q_N, \tag{2.2.7}$$

where the sequence of inputs u^* are retrieved from the feedback law $u^* = -K_t u_t$. It is interesting that if $N \to \infty$, then the time-varying feedback gain K_t, and the time-varying Ricatti matrix P_t stabilises to a constant value. An LQR with $N \to \infty$ is typically called infinite horizon LQR and are one type of controller that is of interest later on. The advantage of the infinite horizon controller is that the sequence of equations (2.2.4) - (2.2.7) becomes time-independent, and simplifies to [10]

$$u_t = -Kx_t \tag{2.2.8}$$

$$K = R^{-1} B^\top P (I + BR^{-1} B^\top P)^{-1} A \tag{2.2.9}$$

$$P = Q + A^\top P (I + BR^{-1} B^\top P)^{-1} A \tag{2.2.10}$$

$$P = P^\top \geq 0 \tag{2.2.11}$$

where, interestingly, the Ricatti matrix P denotes the total cost, such that $\min_{x,u} \sum_{t=0}^{N} L(x_{t+1}, u_t) = x_0^\top P x_0$

2.2.2 Nonlinear Model Predictive Control

While the LQR is excellent for some problems, they are generally not applicable to systems which have significant constraints on states and dynamics. In addition, if the linearised system dynamics become too inaccurate, then the LQR will also struggle to provide consistent results. The Nonlinear Model Predictive Controller (NMPC) [11][12] is one solution to such problems. It is a controller that serves the same purpose, to find a sequence of inputs that minimise some performance objective J, however, it may also incorporate inequalities and nonlinearities. The sequence of inputs is derived based upon the performance objective and all of the constraints. In typical operations as a controller, the NMPC will calculate a sequence of outputs, where only the first input will be applied to the system. The state trajectory estimated in the optimisation is then fed back to the controller, and a new sequence of outputs are produced, visualised in Figure 2.3

In opposition to the LQR, the NMPC is able to cope with both nonlinear system dynamics and with system constraints. This flexibility makes them much more useful in settings where violations of certain state and input constraints are critical. NMPC's are a broad category of controllers and have many different shapes and forms. The type of NMPC that are of interest in this thesis are those on the following form,

$$\begin{aligned} u^* = \arg\min_{x,u} \quad & J(x,u) \\ \text{s.t} \quad & \\ & g_{eq}(x,u) = 0 \\ & g_{Ieq}(x,u) \leq 0. \end{aligned} \tag{2.2.12}$$

Where J may be a nonlinear performance objective, g_{eq} is a set of linear equalities, and g_{ieq} us a set of nonlinear inequalities. This type of problem is fundamental later in this thesis. In opposition to the LQR, this type of problem does not have a trivial solution, and, depending on the shapes and forms of the costs and constraints functions, the problem might be particularly hard to solve. As a result, there exists a multitude of different solvers, each specified towards differently posed problems. The solvers that will be used in this thesis is part of a class of solvers known as interior-point solvers [12]. These solvers search for solutions along the null space of the KKT conditions, given by

$$\mathcal{L} = \lambda^\intercal g_{\text{eq}} + \chi^\intercal g_{\text{Ieq}} + J(x,u) \tag{2.2.13}$$
$$\nabla_{x,u}\mathcal{L}(x,u,\lambda,\chi) = 0 \tag{2.2.14}$$
$$g_{eq} = 0 \tag{2.2.15}$$
$$g_{\text{Ieq}} \leq 0 \tag{2.2.16}$$
$$\chi \geq 0 \tag{2.2.17}$$
$$[\lambda,\chi] \circ [g_{\text{eq}}, g_{\text{Ieq}}] = 0. \tag{2.2.18}$$

The KKT conditions may furthermore be collected in a KKT-vector,

$$R(x,u,\Lambda) = \begin{bmatrix} \nabla_{x,u}\mathcal{L}(x,u,\Lambda) \\ g_{eq} \\ \chi \circ g_{\mathrm{Ieq}} \end{bmatrix} = 0, \tag{2.2.19}$$

where Λ is a vector of the Lagrangian multipliers. The KKT-vector becomes important later for the derivations of the sensitivities for the NMPC scheme.

2.3 Statistics

This section will focus on providing the mathematical tools that will be used when working with the research in this thesis. It is not an extensive list and provides only the necessary tools required for the research provided in the next chapter. The focus is upon variance and expected value operators. The concepts are initially presented in just the scalar case before a short extension to the vector case is provided. The section ends by briefly discussing Taylor expansion in the domain of statistics.

2.3.1 Variance and expected Value

In statistics, there are two important properties that is used to describe random variables. The first property is the mean, generally denoted μ, whereas the second property is the variance, denoted Σ. These properties are special cases of a more general concept that is known as moments. The n^{th} moment of a random variable is the expected value of that variable raised to the n^{th} power. From this, it should be clear that the mean μ of a random variable χ is also the first moment. However, the variance is not the second moment of a random variable, but the second centralised moment, centralised meaning that the expected value is taken of a random variable centred at zero. In general, the respective n^{th} moment and n^{th} centralised moment is given by

$$\mu'_n = \mathbb{E}[\chi^n], \tag{2.3.1}$$
$$\mu_n = \mathbb{E}[(\chi - \mathbb{E}[\chi])^n]. \tag{2.3.2}$$

The apostrophe indicates that the moment is not centred. The centralised and regular moments are very much related, and a general conversion formula between the two is given as:

$$\mu_n = \Sigma_{j=0}^{n} \binom{n}{j} (-1)^{n-j} \mu'_j \mu^{n-j}. \tag{2.3.3}$$

For the research in this thesis, the first, second, and third regular moments of random variables are of interest. The first moment has already been mentioned and is simply the mean. For the second moment, the conversion gives rise to the fairly known formula for the variance,

$$\mathrm{var}(\chi) = \mathbb{E}[\chi^2] - \mathbb{E}[\chi]^2 \quad \Leftrightarrow \tag{2.3.4}$$
$$\mu_2 = \mu'_2 - \mu^2, \tag{2.3.5}$$

which may be solved for the second moment as

$$\mu'_2 = \mu_2 + \mu^2. \tag{2.3.6}$$

Finally, the third moment μ_3'. Utilising the same conversion between central and regular moments on the third moment, the third moment may be expressed as

$$\mu_3 = \mu_3' - 3\mu\mu_2' + 2\mu^3, \tag{2.3.7}$$
$$\mu_3' = \mu_3 + 3\mu(\mu_2 + \mu^2) - 2\mu^3, \tag{2.3.8}$$
$$\mu_3' = \mu_3 + 3\mu\mu_2 + \mu^3. \tag{2.3.9}$$

The first three moments above are now expressed in terms of the random variable's mean, variance and skew (third centralised moment). The variance and mean are often known as they are used as design variables when constructing random variables of specific distributions. It is also important to note that the third and first centralised moment of a centred normal distribution is equal to zero. In fact, every odd centralised moment is zero for a centred normal distribution, but for the calculations in the next chapter, only the three first moments will be required.

2.3.2 Expansion to matrices and vectors

The equations above are only consistent with scalar random variables, and an extension to vectors and matrices are required. Consider $\chi \in \mathbb{R}^n$, a random variable with mean μ and standard deviation Σ, $A \in \mathbb{R}^{n\times n}$, a constant square matrix, and $B \in \mathbb{R}^n$, a constant vector. The following results for the expected value operator then hold true:

$$\mathbb{E}[\chi] = \mu \tag{2.3.10}$$
$$\mathbb{E}[\chi\chi^\top] = \mu\mu^\top + \Sigma\Sigma^\top \tag{2.3.11}$$
$$\mathbb{E}[\chi^\top A\chi] = \mu^\top A\mu + \mathrm{Tr}(\Sigma A\Sigma^\top) \tag{2.3.12}$$
$$\mathbb{E}[\chi B^\top A\chi] = (\mu\mu^\top + \Sigma\Sigma^\top)A^\top B. \tag{2.3.13}$$

The above is the extension of the first and second scalar moments of random variables to matrices. Typically, the third moment of a random variable is not known, but by utilising that the third centralised moment is zero and that a stochastic variable χ may be written as $\chi = \mu + Y$, where $Y \sim \mathcal{N}(0, \Sigma\Sigma^\top)$ (The distribution for Y must the same as the distribution for χ) then the third moment may be expressed through the first and second moment. This gives the third moment for matrices (in the form that will be used later) as

$$\mathbb{E}[\chi\chi^\top A\chi] = \mu\mu^\top A\mu + \Sigma\Sigma^\top(A + A^\top) + \mu\mathrm{Tr}(\Sigma A\Sigma^\top). \tag{2.3.14}$$

2.3.3 Taylor expansion of moments of functions

The final tool that it is necessary to introduce is the Taylor expansion for the moments of functions of random variables. This is essentially just a Taylor expansion around the mean of a random variable. Consider a random scalar variable

χ with mean μ and variance $\Sigma\Sigma^\top$. In addition, consider some unknown function $f(\chi)$. With the function f being unknown, it is difficult to evaluate any moments of the function $\mathbb{E}[f(\chi)]$. Assume now that the first and second derivative of f exists. The second order Taylor expansion of f around the mean μ becomes

$$f(\chi) \approx f(\mu) + (\chi - \mu)f'(\mu) + \frac{1}{2}(\chi - \mu)^2 f''(\mu) \tag{2.3.15}$$

Taking the expected value of the approximation yields

$$\mathbb{E}[f(\chi)] \approx \mathbb{E}[f(\mu)] + \mathbb{E}[(\chi - \mu)f'(\mu)] + \mathbb{E}[\frac{1}{2}(\chi - \mu)^2 f''(\mu)] \tag{2.3.16}$$

$$\approx \mathbb{E}[f(\mu)] + \mathbb{E}[\frac{1}{2}(\chi - \mu)^2 f''(\mu)], \tag{2.3.17}$$

where $\mathbb{E}[(\chi - \mu)f'(\mu)]$ constitutes the first central moment and is zero for all random variables. Due to f in this case being unknown, the second order term $\mathbb{E}[\frac{1}{2}(\chi - \mu)^2 f''(\mu)]$ is not calculable. Therefore, the first order Taylor approximation becomes

$$\mathbb{E}[f(\chi)] \approx f(\mu) + \frac{1}{2}\Sigma\Sigma^\top f''(\mu) \tag{2.3.18}$$

$$\mathbb{E}[f(\chi)] \approx f(\mu) + \epsilon. \tag{2.3.19}$$

with an error $\epsilon \propto \Sigma\Sigma^\top f''(\mu)$ proportional to the curvature of the function and to the variance of the variable. Furthermore, if χ satisfies the Central Limit Theorem (CLT) and $\Sigma\Sigma^\top \to 0$, then the delta method [13] may be applied to show that the error terms $\epsilon \to 0$ and that the expansion is an exact estimate.

III Research

This chapter encompasses four sections that investigate the research questions that were proposed in the introduction of this thesis. The first section considers the first research question and presents two different safe Reinforcement Learning (RL) schemes that incorporate Nonlinear Model Predictive Controller (NMPC). One of these schemes is later utilised for experiments. The second section is an investigation of anisotropic exploration, as per the second research question. In this section, brief arguments are initially presented that justifies how anisotropic exploration is inevitable in safe RL schemes. The section then proceeds by analytically evaluating the consequences of anisotropic exploration. The chapter finalises with the second to last section providing some experiments and complementary results, whereas the final section provides a discussion of the theory and experiments.

3.1 Integration of NMPC in RL

In this thesis, RL will be integrated with NMPC in two different ways. Either NMPC may serve as a function approximator for the action-value function in Q-learning, or it may serve as the policy in the policy gradient method.

This section begins by reasoning why an NMPC may serve as a function approximator and to shortly comment on the similarities between the NMPC and RL. Only brief arguments that display the key connections between NMPC and RL are presented. For further proof and rationalisation the reader should refer [2] and [8]. After that, the NMPC controller in the position as a function approximator in Q-learning is presented, before ending on how to use the NMPC scheme as a policy in the policy gradient method.

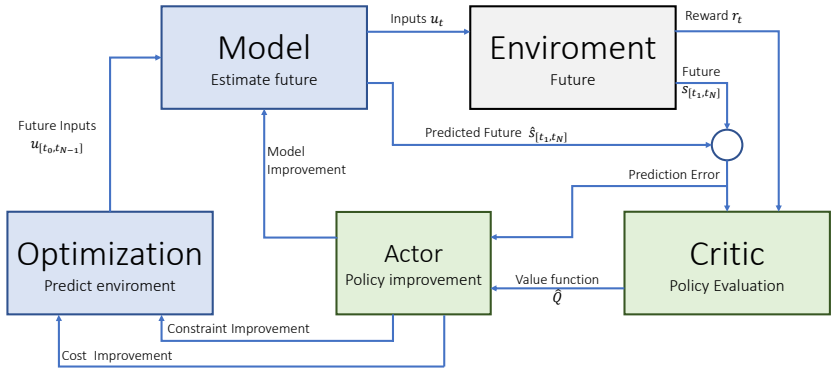


Figure 3.1: Feedback loop visualisation of a generic predictive controller

3.1.1 NMPC as function approximator

Consider a basic linear system in its state-space representation, together with a quadratic stage cost $L(s, u)$,

$$x_{t+1} = Ax_t + Bu_t \quad , \quad L(s,u) = \begin{bmatrix} x \\ u \end{bmatrix}^\top \begin{bmatrix} T & N \\ N^\top & R \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}. \tag{3.1.1}$$

Moreover, consider applying an RL algorithm that attempts to solve the posed problem. The goal of the agent would be to minimise the discounted accumulated cost gained from $L(s, u)$, i.e. to find a policy π such that

$$\pi_\theta = \arg\min_\theta \mathbb{E}[\sum_{t=0}^{N} \gamma^{t-1} L(s_t, u_t)]. \tag{3.1.2}$$

A traditional Q-learning algorithm would attempt to fit a function approximator $\hat{Q}_{\pi_\theta}$ to the true action-value function Q_{π_θ}, as a means to improve the policy by choosing the input that provides the lowest instantaneous value return. This technique for finding an optimal policy is named policy improvement, and eventually, this would possibly ensure that the policy becomes optimal. Traditionally, the function approximator takes the form of some neural network with internal weights θ that attempts indirectly to solve, or to find, the solution for the optimisation problem (3.1.2). However, for a simple linear quadratic system, as in (3.1.1), the value function may be gained directly from the system dynamics as the solution to the following problem:

$$\begin{aligned} V^*_{\pi_\theta}(s) = \min_{x,u} \quad & \sum_{t=0}^{N} \gamma^{t-1} L(x_t, u_t) \\ \text{s.t} \quad & \\ & x_{t+1} = Ax_t + Bu_t, \\ & x_0 = s_t, \end{aligned} \tag{3.1.3}$$

where the optimal policy in any state s_t would be the first input from the sequence $u = u_0, ..., u_{N-1}$ that is generated from the solution of the above scheme, $\pi^*_\theta(s) = u_0$. Furthermore, the optimal action-value function $Q^*_{\pi_\theta}$ for the same system could be generated by solving the same scheme, but with the added constraint of $u_0 = a_t$, giving

$$\begin{aligned} Q^*_{\pi_\theta}(s_t, a_t) = \min_{x,u} \quad & \sum_{t=0}^{N} \gamma^{t-1} L(x_t, u_t) \\ \text{s.t} \quad & \\ & x_{t+1} = Ax_t + Bu_t, \\ & x_0 = s_t, \\ & u_0 = a_t. \end{aligned} \tag{3.1.4}$$

Note that the bellman equations that are fundamental for the RL algorithms are still satisfied:

$$\pi_\theta(s) = \arg\min_a Q(s,a), \quad V(s) = \min_a Q(s,a) \tag{3.1.5}$$

There is a multitude of different solvers and methods that may solve such optimisation problems, and the advantage of being able to express the problem in such a form will soon become apparent. Consider now that some insight about the aforementioned system is known a priori, e.g. that the system is linear or perhaps the structure of some the matrices, to the degree that an estimate of the environment may be given as

$$x_{t+1} = \hat{A}x_t + \hat{B}u_t \quad , \quad \hat{L}(s,u) = \begin{bmatrix} x \\ u \end{bmatrix}^\top \begin{bmatrix} \hat{T} & \hat{N} \\ \hat{N}^\top & \hat{R} \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}, \tag{3.1.6}$$

where $\hat{A}, \hat{B}, \hat{T}, \hat{R}, \hat{N}$ are estimates of the parameters A, B, T, R, N. In traditional Q-learning, this a priori insight would be difficult to utilise efficiently, as it would not be clear how to properly initialise the parameters of the neural network in order to fully exploit the a priori knowledge. However, one may construct a similar optimisation problem, based upon the a priori insight:

$$\begin{aligned} \hat{Q}_{\pi_\theta}(s,a) = \min_{x,u} \quad & \sum_{t=1}^{N} \gamma^{t-1} \hat{L}(x_t, u_t) \\ \text{s.t} \quad & \\ & x_{t+1} = \hat{A}x_t + \hat{B}u_t \\ & x_0 = s \\ & u_0 = a \end{aligned} \qquad \begin{aligned} \hat{V}_{\pi_\theta}(s) = \min_{x,u} \quad & \sum_{t=1}^{N} \gamma^{t-1} \hat{L}(x_t, u_t) \\ \text{s.t} \quad & \\ & x_{t+1} = \hat{A}x_t + \hat{B}u_t, \\ & x_0 = s \end{aligned} \tag{3.1.7}$$

If the estimated parameters are fairly accurate, then a slight adjustment in the parameters may yield the optimal policy, and if it were possible to utilise these modified problems as function approximator in RL, then this would yield and advantage above traditional methods. This is contrary to neural networks, where any a priori insight typically must be derived from previously trained networks, a technique known as transfer learning [14]. Therefore, if the sensitivities necessitated by RL algorithms exist for the optimisation schemes (3.1.7), then there should be no hindrance for the utilisation of such function approximators above any other neural network. These sensitivities do exist and will be presented in next sections.

Note that the example parametrisation and systems provided above would make it possible to find the optimal policy exactly, due to the solution being given as a whole by the scheme (3.1.3). This would not be easy to guarantee in the case of neural networks, even in such a simple environment. However, in the case where the true system dynamics are stochastic and nonlinear, in addition to a complex reward function, then it becomes less trivial to fit an exact optimisation scheme with a limited parametrisation. A final note of interest is that it should be possible to find the parametrised optimal policy, regardless of the underlying model implemented in the NMPC scheme, by simply tuning the performance objective [2].

3.1.2 NMPC and Q-learning

In Q-learning, a proper function approximator is vital for the approximation of the true action-value function defined in 2.1.3, and as established in the previous subsection, an optimization problem may work as a function approximator. Therefore, consider a general environment and the following NMPC schemes as function approximators for the corresponding action-value function and value

function:

$$\begin{aligned} \hat{Q}_{\pi_\theta}(s,a) = \min_{x,u} \quad & \hat{J}_\theta(x,u) \\ \text{s.t} \quad & \\ & g_\theta(x,u) = 0 \\ & h_\theta(x,u) \leq 0 \\ & x_0 = s \\ & u_0 = a \end{aligned} \qquad \begin{aligned} \hat{V}_{\pi_\theta}(s) = \min_{x,u} \quad & \hat{J}_\theta(x,u) \\ \text{s.t} \quad & \\ & g_\theta(x,u) = 0, \\ & h_\theta(x,u) \leq 0 \\ & x_0 = s \end{aligned} \tag{3.1.8}$$

θ are the internal parameters of the policy that is tuned in order to improve the function approximation. Note that the only difference between $\hat{Q}_{\pi_\theta}$ and $\hat{V}_{\pi_\theta}$ is the lack of the constraint $u_0 = a$ in $\hat{V}_{\pi_\theta}$. The parameter update law, as given by the Q-learning algorithm, (2.1.10) -(2.1.11), is given as

$$\theta_{k+1} = \theta_k + \alpha \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [(r + \min_a \hat{Q}_{\pi_\theta}(s_{t+1},a) - \hat{Q}_{\pi_\theta}(s_t,a_t)) \nabla_\theta \hat{Q}_{\pi_\theta}(s_t,a_t)], \tag{3.1.9}$$

$$= \theta_k + \alpha \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [(r + \hat{V}_{\pi_\theta}(s_{t+1}) - \hat{Q}_{\pi_\theta}(s_t,a_t)) \nabla_\theta \hat{Q}_{\pi_\theta}(s_t,a_t)]. \tag{3.1.10}$$

Consequently, the sensitivities $\nabla_\theta \hat{Q}_{\pi_\theta}(s_t,a_t)$ of the NMPC scheme is required when updating the parameters θ. The Lagrangian defined in (2.2.13) turns the above constrained scheme with the primal variables (x,u) into an unconstrained optimisation scheme with the primal and dual variables (x,u,Λ) [12]. Hence, the necessary sensitivities of the NMPC may simply be extracted from the Lagrangian [2] as

$$\nabla_\theta \hat{Q}_{\pi_\theta}(s,a) = \nabla_\theta \mathcal{L}(x,u,\Lambda). \tag{3.1.11}$$

The prime reason for the introduction of NMPC in RL is the NMPC's ability to handle constraints, and as such, the consequences of constraints must be discussed. A violation of a constraint in the NMPC may be compared to an infinite penalty, and in fact, the NMPC is not solvable if the constraints are violated. However, RL algorithms do not cope well with infinite penalties, implicating that the scheme must be modified to an extent where infinite penalties are avoided. This may be performed through either a robust NMPC scheme or with the introduction of slack variables σ. A robust NMPC has built-in parametric uncertainties that enable it to guarantee that constraints are never violated, and an NMPC is as such advantageous for real-life implementation. The slack variables are a simpler method that heavily penalises constraint violations. Specifically, the slack variables are optimisation variables that penalise a unit violation of a constraint by the parameters ω. The RL algorithm will then learn not to violate the constraints. A general example of a parametrised NMPC with slack variables

may be written as

$$\hat{Q}_{\pi_\theta}(s,a) = \min_{x,u} \quad \gamma^N(\hat{V}_\theta^f + \omega_f^\top \sigma) + \sum_{t=0}^{N-1} \gamma^t(\hat{L}_\theta(x_t,u_t) + \omega^\top \sigma_t) \tag{3.1.12}$$

$$\begin{aligned} \text{s.t} \quad & \\ & x_{t+1} = f_\theta(x_t,u_t) \\ & h_\theta(x,u,\sigma) \leq 0 \\ & x_0 = s, \quad u_0 = a \\ & \sigma \geq 0, \end{aligned}$$

where $\hat{L}_\theta(x_t,u_t)$ is the estimated state reward, $\omega^\top \sigma_t$ is the penalty for violating the constraints, and $\hat{V}_\theta^f$ is a final cost that constitutes to the Linear Quadratic Regulator (LQR) infinite horizon controller where $N \leq \infty$ and may be approximated by the infinite Ricatti matrix. The policy is given by the Bellman equation $\pi_\theta(s) = \arg\min_a Q(s,a)$, which is the first input generated from solving the NMPC scheme in (3.1.12) without the constraint $u_0 = a$.

3.1.3 NMPC and policy gradient

In the policy gradient scheme, the NMPC will act directly as a policy. Its parametrization is however similar to previous parametrizations, giving the policy as

$$\pi_\theta(s) = \arg\min_{x,u} \quad \gamma^N(\hat{V}_\theta^f + \omega_f^\top \sigma) + \sum_{t=0}^{N-1} \gamma^t(\hat{L}_\theta(x_t,u_t) + \omega^\top \sigma_t) \tag{3.1.13}$$

$$\begin{aligned} \text{s.t} \quad & \\ & x_{t+1} = f_\theta(x_t,u_t) \\ & h_\theta(x,u) \leq 0 \\ & x_0 = s \\ & \sigma \geq 0, \end{aligned}$$

where the action provided by the policy is the first input acquired from the solution $u = u_0, ..., u_{N-1}$, $\pi_\theta(s) = u_0$. The parameter update law for the policy gradient, as given in definition 2.1.4 and 2.1.6, are

$$\theta_{k+1} = \theta + \alpha \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [\nabla_\theta \pi_\theta \nabla_a Q_{\pi_\theta}(s_t,a_t)|_{a=\pi_\theta(s_t)}]. \tag{3.1.14}$$

Through using the actor-critic method to establish an estimate of $\nabla_a Q_{\pi_\theta}$, the parameter update law for the policy may be written as

$$\theta_{k+1} = \theta + \alpha \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \omega. \tag{3.1.15}$$

Subsequently, the sensitivities of the policy $\nabla_\theta \pi_\theta$ are required. These may be acquired from the KKT-vector (2.2.19), where the first-order sensitivities for the

primal-dual variables $y = \begin{bmatrix} x & u & \Lambda \end{bmatrix}^\top$ stem from the linear equations provided by Implicit Value Theorem (IVT)

$$\frac{\partial R \partial y}{\partial y \partial \theta} + \frac{\partial R}{\partial \theta} = 0 \tag{3.1.16}$$

$$\frac{\partial y}{\partial \theta} = -\Big(\frac{\partial R}{\partial y}\Big)^{-1} \frac{\partial R}{\partial \theta}. \tag{3.1.17}$$

Only the sensitivities of the policy $\nabla_\theta \pi_\theta = (\frac{\partial u_0}{\partial \theta})^\top$ are of interest, and they may be extracted from the vector $\frac{\partial y}{\partial \theta}$ consisting of the primal-dual sensitives by utilising the partial derivative of the primal-dual variables, giving

$$\nabla_\theta \pi_\theta^\top = -\frac{\partial y}{\partial u_0}^\top \Big(\frac{\partial R}{\partial y}\Big)^{-1} \frac{\partial R}{\partial \theta} \tag{3.1.18}$$

$$\nabla_\theta \pi_\theta = -\nabla_\theta R (\nabla_y R)^{-1} \frac{\partial y}{\partial u_0}. \tag{3.1.19}$$

The second part of the actor-critic method is the critic, typically parametrised by ω. It is described through the compatible function approximator defined in definition 2.1.7, and is reiterated here:

$$\hat{Q}_w(s,a) = (a - \pi_\theta(s))^\top \nabla_\theta \pi_\theta^\top \omega + \hat{V}_\nu(s). \tag{3.1.20}$$

The parameters ω, together with the baseline parameters (discussed in section 2.1.5), has the update laws as given in (2.1.18) - (2.1.19). Utilising the least squares method yields the complete parameter batch update laws for the actor-critic policy gradient method as:

$$\nu_k = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [\nabla_\nu \hat{V}_\nu(s_t)(\nabla_\nu \hat{V}_\nu(s_t) - \gamma \nabla_\nu \hat{V}_\nu(s_{t+1}))^\top]^{-1} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [r \nabla_\nu \hat{V}_\nu(s_t)] \tag{3.1.21}$$

$$\omega_k = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}[\nabla_\theta \pi_\theta e (r + \gamma \hat{V}_\nu(s_{t+1}) - \hat{V}_\nu(s_t))] \tag{3.1.22}$$

$$\theta_{k+1} = \theta + \alpha \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \omega_k. \tag{3.1.23}$$

Note that ν_k and ω_k are estimations of parameters that are fitted to the dataset k, and that in order to necessitate only one data sweep, they must be calculated in order. If e.g. θ_{k+1} were calculated before ω and ν, then the update would be erroneous, and based upon a combination of the previous dataset and therefore also a combination of the previous policy parameters θ_k and θ_{k-1}.

3.1.4 Brief summary of section

This section reasoned why an NMPC scheme could be used as a function approximator in the domain of RL. Furthermore, it presented two common RL algorithms that utilised an NMPC as a function approximator. The Q-learning algorithm was primarily presented here as a steppingstone to the policy gradient with NMPC, making it more intuitive. For the experiments presented later in this thesis, only the policy gradient with NMPC will be used. The experiments

consider primarily an infinite horizon LQR policy, but the sensitivities of the LQR are not apparent. The NMPC is a perfect tool that, with a proper prediction horizon, may approximate the LQR accurately, and of which the sensitivities are simple to calculate.

3.2 Anisotropic Exploration with NMPC and RL

This section concerns itself with the second research goal "*What are the consequences of using anisotropic exploration in the context of NMPC and RL?*". It starts by presenting why anisotropic exploration might be a problem and why it might occur. Thereafter, the policy gradient with the actor-critic method is evaluated analytically in a general system in order to observe the effects of anisotropic exploration. The last part of this section establishes some convergence bounds on the estimated performance gradient.

3.2.1 Why anisotropic exploration may occur

The deterministic policy gradient derived in [5] have the assumption that the stochastic exploration policy β is a centred normal distribution with isotropic variance $\Sigma\Sigma^\top = \sigma\mathbb{I}$. It is not given that this assumption extends to an exploration policy with an anisotropic variance; similarly, it also not given that the results extend with β being not centred. Although mentioned in [8], the consequences of breaking these assumptions have not been thoroughly investigated in previous literature. With the introduction of safe RL, exploration schemes that isotropic and centred are no longer always possible, which warrants an investigation. Previously in this thesis, constraints on the states in the NMPC have been tackled by utilising penalising slack variables. This enables the scheme to learn not to violate these constraints, as it is allowed to break the constraints in order to observe and learn the consequences. If one were to utilise RL in a real-time setting, then suddenly unbreakable constraints become essential. Anisotropic exploration arises with the introduction of unbreakable constraints.

Motivating example

Consider an example robot arm, Figure 3.2, that is controlled by an NMPC scheme. Consider also some obstacles that are in the reach of the robot arm. The arm is required to operate around the obstacles and the arms safe state domain $\mathcal{S}_{\text{safe}}$ becomes a subset of the complete state domain $\mathcal{S}$, $\mathcal{S}_{\text{safe}} \leq \mathcal{S}$. It is desirable to improve the performance of the arm in real-time by applying an RL algorithm to the NMPC scheme that controls the arm. However, the exploration policy is not permitted to explore outside the safe state domain. The exploration is therefore limited when approaching the edges of the safe state domain, and if the state of the arm is at the edge on the safe state domain, an isotropic exploration algorithm is no longer possible as it may not explore in certain directions.

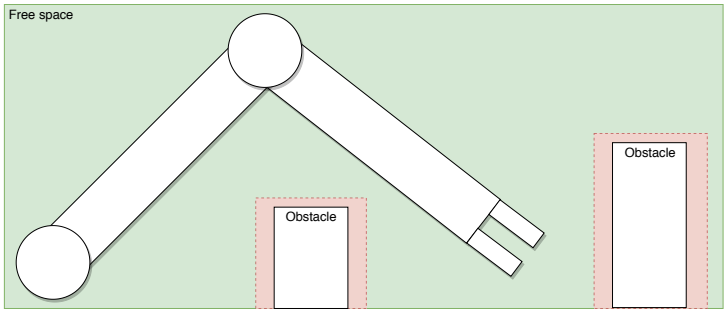


Figure 3.2: In this example, the arm has a theoretical reach of the entire green area. However, a couple of obstacles are within the arms operating area, and the arm may not approach these. The operating state space for the arm is limited to a subset state-space $\mathcal{S}_{\text{safe}} < \mathcal{S}$

Implementing safe exploration

A typical stochastic exploration policy $\beta : e = a - \pi_\theta(s)$ may be generated by simply choosing the exploratory move a as a normally distributed stochastic variable centred at the policy, $a \sim \mathcal{N}(\pi_\theta(s), \Sigma\Sigma^\top)$. However, this method of generating exploration does not take any considerations of constraints on neither the input a nor the state s, and is not properly applicable in safe RL schemes. One method that allows for the generation of safe exploratory moves is an implementation of small perturbations to the policy variables u_0 in the NMPC. The addition of the term $d^\top u_0$, where $d \sim \mathcal{N}(0, \Sigma\Sigma^\top)$, to the performance objective is sufficient to generate a safe exploratory move:

$$
\begin{aligned}
a_t = \arg\min_{x,u} \quad & \gamma^N \hat{V}_\theta^f + d^\top u_0 + \sum_{t=0}^{N-1} \gamma^t (\hat{L}_\theta(x_t, u_t)) \\
\text{s.t} \quad & \\
& x_{t+1} = f_\theta(x_t, u_t) \\
& h_\theta(x, u) \leq 0 \\
& x_0 = s
\end{aligned}
\tag{3.2.1}
$$

When $d \neq 0$, the NMPC scheme will generate a distribution of exploratory actions centred around the desired policy π_θ, while simultaneously respecting any constraints implemented in the NMPC scheme.

When utilising perturbations in the performance objective to generate exploratory moves, the covariance of the exploration may no longer be guaranteed to be isotropic. A simple example is presented that visualises the problem. Con-

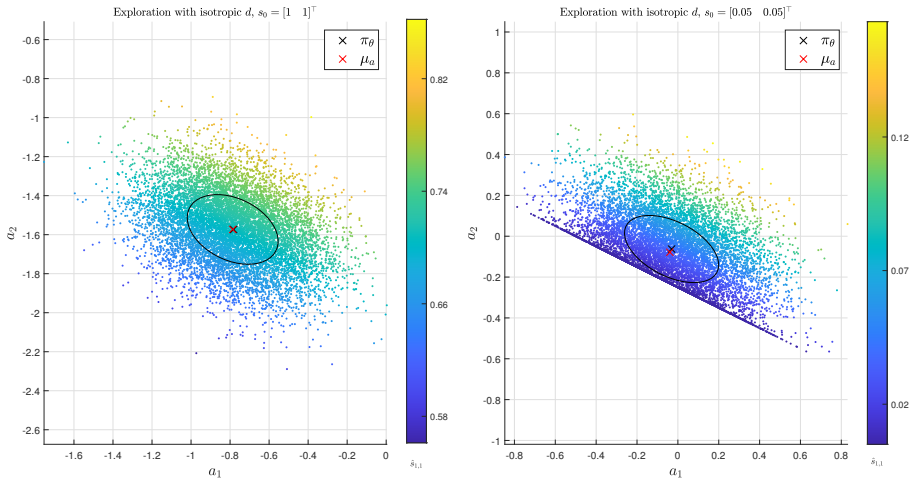


Figure 3.3: Visualisation of anisotropic exploration generated from an NMPC scheme with isotropic perturbation. The colour is the value of the predicted state $\hat{s}_{1,1}$ that approaches the constraint $s \geq 0$

sider the following linear quadratic to generate safe exploratory moves:

$$
\begin{aligned}
a_t = \arg\min_{x,u} \quad & \gamma^N x_n^\top P x_n + d^\top u_0 + \sum_{t=0}^{N-1} \gamma^t \left(\begin{bmatrix} x_t \\ u_t \end{bmatrix}^\top \begin{bmatrix} T & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} x_t \\ u_t \end{bmatrix} \right) \\
\text{s.t} \quad & \\
& x_{t+1} = Ax_t + Bu_t \\
& x \geq 0 \\
& x_0 = s.
\end{aligned} \tag{3.2.2}
$$

Choosing d as a centred isotropic normal distribution, $d \sim \mathcal{N}(0, \mathbb{I})$, the goal would be to generate isotropic exploration. In Figure 3.3, it is clear that this was not possible, regardless of the constraints. On the left-hand side, with the state being $s_0 = [1 \quad 1]$, neither of the states are close to any hard constraints. Still, the safe exploration generated becomes anisotropic, with its covariance given as $\Sigma_1\Sigma_1$ in (3.2.3). On the right-hand side, the initial states are given as $s_0 = [.05 \quad .05]$, close to the hard constraints $s \geq 0$. Here, the actions become limited due to the activation of the constraint $s_1 > 0$, which in turn increases the anisotropy of the exploration. The covariance of the second example is given as $\Sigma_2\Sigma_2$ in (3.2.3). Another consequence of the constraints is that the exploration mean deviates away from zero. This is also slightly visible on the right-hand side of Figure 3.3. However, this is not the primary focus of this thesis, and it will only be briefly mentioned later.

$$\Sigma_1\Sigma_1^\top = \begin{bmatrix} 0.0543 & -0.0141 \\ -0.0141 & 0.0319 \end{bmatrix} \qquad \Sigma_2\Sigma_2^\top = \begin{bmatrix} 0.0532 & -0.0180 \\ -0.0180 & 0.0270 \end{bmatrix} \tag{3.2.3}$$

3.2.2 Expected value of the estimated policy gradient

A brief recap of some definitions that will be of importance for the analysis to come is necessitated before proceeding. In chapter two, the true performance gradient $\nabla_\theta J(\pi_\theta)$, as used in the policy gradient method, was defined as

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_a A_{\pi_\theta}(s,a)]. \tag{3.2.4}$$

Moreover, the estimated performance gradient $\nabla_\theta \hat{J}(\pi_\theta)$, as derived from the actor-critic method, were also defined in chapter two as:

$$\nabla_\theta \hat{J}(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_a \hat{A}_{\pi_\theta}(s,a)]. \tag{3.2.5}$$

Finally, the exploration, definition 2.1.5, was given as some stochastic policy β:

$$\beta : e = a - \pi_\theta, \quad a \sim \mathcal{N}(\pi_\theta, \Sigma\Sigma^\top), \quad \Rightarrow \quad e \sim \mathcal{N}(0, \Sigma\Sigma^\top). \tag{3.2.6}$$

As previously established in (2.1.22), the estimated performance gradient has been shown to equal the true performance gradient when the exploration policy is isotropic and centred. The goal of this section will be to explore what happens when the exploration covariance $\Sigma\Sigma$ is not isotropic. This will be done by analytically evaluating the expected value of the estimated performance gradient for a general system. Therefore, consider some general deterministic state dynamics and reward function

$$s_{t+1} = f(s_t, a_t), \quad r = L(s_t, a_t), \tag{3.2.7}$$

and assuming that true advantage function may be written on a quadratic form:

$$A_{\pi_\theta} = \begin{bmatrix} s_t \\ a_t \end{bmatrix}^\top W \begin{bmatrix} s_t \\ a_t \end{bmatrix} + F^\top \begin{bmatrix} s_t \\ a_t \end{bmatrix} + C \tag{3.2.8}$$

$$= \begin{bmatrix} s_t \\ a_t \end{bmatrix}^\top \begin{bmatrix} W_{1,1} & W_{1,2} \\ W_{2,1} & W_{2,2} \end{bmatrix} \begin{bmatrix} s_t \\ a_t \end{bmatrix} + \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}^\top \begin{bmatrix} s_t \\ a_t \end{bmatrix} + C. \tag{3.2.9}$$

(Note that an advantage function on this form will only be exact for f linear and L quadratic. In any other cases, it will not be exact). The estimated performance gradient, while utilizing the actor-critic method, for such a system is given as

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top]\omega \tag{3.2.10}$$

$$\omega_k = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e (L + \gamma \hat{V}_\nu(s_{t+1}) - \hat{V}_\nu(s_t))], \tag{3.2.11}$$

where $\hat{V}_\nu$ is a baseline function. The parameters ν of the baseline function are chosen such that they minimize the mean squared error between the baseline function and the true value function, $\nu = \arg\min_\nu \mathbb{E}[(\hat{V}_\nu - V_{\pi_\theta})^2]$, of which its solution is given by solving the least-squares problem in (3.1.21). Given that $\hat{V}_\nu$ is sufficiently parametrized, then it may be assumed that its estimation becomes exact, such that $\hat{V}_\nu = V_{\pi_\theta}$. With this assumption, the parameter update law for ω may be reduced to

$$\omega_k = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e (L + \gamma V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t))] \quad (3.2.12)$$

$$\omega_k = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e A_{\pi_\theta}(s_t, a_t)], \quad (3.2.13)$$

where it has been used that

$$A_{\pi_\theta}(s_t, a_t) = Q_{\pi_\theta}(s_t, a_t) - V_{\pi_\theta}(s_t) \quad (3.2.14)$$

$$A_{\pi_\theta}(s_t, a_t) = r + \gamma V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t). \quad (3.2.15)$$

From this, the estimated performance gradient may be written as

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e A_{\pi_\theta}(s_t, a_t)]. \quad (3.2.16)$$

The expected value is taken across the two stochastic variables a and s. The state s relies on some unknown distribution ρ with an unknown mean μ_s and covariance $\Sigma_s \Sigma_s^\top$. The variable a stems from a distribution with a known variance and a known mean $a \sim \mathcal{N}(\mu, \Sigma\Sigma^\top)$. It is possible to evaluate the expected value of the estimated performance gradient with respect to a, which is performed in appendix A. The following analysis is based upon the aforementioned evaluation presented in (A.4) - (A.6).

Centred and Isotropic exploration

In (A.4) - (A.6), the only constraint on the exploration is that a is a normally distributed variable, it is possible to extend these results in order to investigate multiple cases. Initially, it is interesting to verify that, under the typical exploration scheme, the estimated policy gradient does indeed equal the true policy gradient. Therefore, the first case to be investigated is a centred exploration scheme with isotropic variance, $a \sim \mathcal{N}(\pi_\theta, \sigma^2 \mathbb{I})$. Setting $\mu = \pi_\theta$ and $\Sigma\Sigma^\top = \sigma\mathbb{I}$ in (A.4) - (A.6) gives the estimated performance gradient with isotropic exploration as

$$\nabla_\theta \hat{J}(\pi_\theta) = \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \sigma^2 \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \sigma^2 \nabla_a A_{\pi_\theta}(s_t, \pi_\theta)] \quad (3.2.17)$$

$$= \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \nabla_a A_{\pi_\theta}(s_t, \pi_\theta)] \quad (3.2.18)$$

$$= \nabla_\theta J(\pi_\theta). \quad (3.2.19)$$

The estimated performance gradient equalling the true gradient under an exploration scheme with isotropic exploration coincides with previous results presented in literature [5]. This method of verification (albeit not efficient) is however new.

Centred and Anisotropic exploration

An anisotropic exploration scheme, meaning that the covariance may not be written in terms of a scalar and the identity matrix, presents some different results. In this case, the exploration is given as $a \sim \mathcal{N}(\pi_\theta, \Sigma\Sigma^\top)$, which, when inserted into (A.4) - (A.6), gives the estimated performance gradient as

$$\begin{aligned}\nabla_\theta \hat{J}(\pi_\theta) =& \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_\theta \pi_\theta^\top]^{-1} \\ & \cdot \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_a A_{\pi_\theta}(s_t, \pi_\theta)].\end{aligned} \tag{3.2.20}$$

Due to the expected value operators in (3.2.20), no immediate further factorizations could be established at this point. Yet some interesting observations may still be attained; the dependency on the sensitivities for the advantage function in $\mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_a A_{\pi_\theta}(s_t, \pi_\theta)]$ ensures the estimated performance gradient is zero when θ is optimal, $\mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_a A_{\pi_\theta}(s_t, \pi_\theta^*)] = 0$. This entails that the estimated performance gradient has at the very least a common local minima with the true performance gradient.

Whereas it proved challenging to investigate the complete estimated performance gradient further, there may still be some additional insights in the estimated gradient granted through an investigation of the instantaneous estimated gradient. Therefore, consider a stationary state s, such that the expected value operators may be ignored. This yields the instantaneous estimated performance gradient as

$$\nabla_\theta \hat{J}(\pi_\theta) = \nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top [\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_\theta \pi_\theta^\top]^{-1} \nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_a A_{\pi_\theta}(s_t, \pi_\theta). \tag{3.2.21}$$

Assuming the following

$$\nabla_\theta \pi_\theta \in \mathbb{R}^{m\times n_a}, \quad \operatorname{rank}(\nabla_\theta \pi_\theta) = \min(m, n_a) \tag{3.2.22}$$

$$\Sigma \in \mathbb{R}^{n_a \times n_a}, \quad \operatorname{rank}(\Sigma) = n_a, \tag{3.2.23}$$

gives that

$$\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_\theta \pi_\theta^\top \in \mathbb{R}^{m\times m}, \quad \operatorname{rank}(.) = \min(m, n_a). \tag{3.2.24}$$

Typically, the dimension of θ is larger than the dimensions of the action a. This implies that $[\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_\theta \pi_\theta^\top]$ is rank insufficient and that its true inverse does not exist. However, its pseudo-inverse exists and may be found by utilizing the reduced Singular Value Decomposition (SVD) on $\nabla_\theta \pi_\theta$. This gives the Moore–Penrose pseudo-inverse of $\nabla_\theta \pi_\theta$ as

$$(\nabla_\theta \pi_\theta)^\dagger = (\nabla_\theta \pi_\theta^\top \nabla_\theta \pi_\theta)^{-1} \nabla_\theta \pi_\theta^\top. \tag{3.2.25}$$

The inverse of matrices follows a reverse order distributive property, giving

$$[\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_\theta \pi_\theta^\top]^{-1} = \nabla_\theta \pi_\theta^{\top^{-1}} \Sigma^{\top^{-1}} \Sigma^{-1} \nabla_\theta \pi_\theta^{-1}. \tag{3.2.26}$$

Inserting (3.2.26) in the estimated performance gradient (3.2.21), and utilizing the pseudo inverse for $\nabla_\theta \pi_\theta$, gives the instantaneous estimated performance gra-

dient as

$$\nabla_\theta \hat{J}(\pi_\theta) = \nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top [\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_\theta \pi_\theta^\top]^{-1} \nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla a A(s,a) \tag{3.2.27}$$

$$= \nabla_\theta \pi_\theta \underbrace{[\nabla_\theta \pi_\theta^\top \nabla_\theta \pi_\theta (\nabla_\theta \pi_\theta^\top \nabla_\theta \pi_\theta)^{-1}]}_{=I} \Sigma^{\top^{-1}} \Sigma^{-1} \tag{3.2.28}$$

$$\cdot \underbrace{[(\nabla_\theta \pi_\theta^\top \nabla_\theta \pi_\theta)^{-1} \nabla_\theta \pi_\theta^\top \nabla_\theta \pi_\theta]}_{=I} \Sigma\Sigma^\top \nabla a A(s,a)$$

$$\nabla_\theta \hat{J}(\pi_\theta) = \nabla_\theta J(\pi_\theta). \tag{3.2.29}$$

This does not imply that the expected value of the estimated performance gradient equal the true gradient. However, it shows that the immediate structure of the estimated gradient is similar to the true gradient, and indicates that any dissimilarities arise from the stochastic state trajectory $\mathcal{S}_n$. This property will be useful later in combination with a Taylor approximation.

Modified approximator, Centred and Anisotropic exploration

The primary hindrance in (3.2.20) is the presence of the exploration covariance $\Sigma\Sigma^\top$, which prevents any cancellations in the expected value terms. By looking at (3.2.20), it may be clear that the addition of the inverse exploration covariance to the compatible function approximator remove the problem. This addition gives rise to the modified compatible function approximator.

Definition 3.2.1. Modified Compatible Function Approximator: A Modified compatible function approximator is designed to cancel any distortions provided from utilizing anisotropic exploration

$$\bar{Q}_w(s,a) = (a - \pi_\theta(s))^\top (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top \omega + \hat{V}_\nu(s). \tag{3.2.30}$$

The modified compatible function approximator yields the *modified* performance gradient $\nabla_\theta \bar{J}(\pi_\theta)$, shown in appendix B, (B.5). The modified performance gradient may be shown to equal the true performance gradient:

$$\nabla_\theta \bar{J}(\pi_\theta) = \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top] \mathbb{E}[\nabla_\theta \pi_\theta (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top]^{-1} \tag{3.2.31}$$

$$\cdot \mathbb{E}[\nabla_\theta \pi_\theta \nabla_a A_{\pi_\theta}(s_t, \pi_\theta)]$$

$$= \mathbb{E}[\nabla_\theta \pi_\theta \nabla_a A_{\pi_\theta}(s_t, \pi_\theta)] \tag{3.2.32}$$

$$= \nabla_\theta J(\pi_\theta). \tag{3.2.33}$$

An important issue with the modified compatible function approximator is its dependency on the inverse exploration covariance. As shown in Figure 3.3, the exploration covariance in safe RL is not necessarily predefined nor constant, and if needed, must be estimated on the fly. In the case of a policy given by an NMPC as above, the inverse of the covariance matrix may be calculated from the second order sensitivities [8]. Thus, the modified compatible function approximator introduces additional complexity in computation, and it would be advantageous to have an algorithm that does not depend on the covariance.

3.2.3 Convergence analysis

While the modified compatible function approximators yield guaranteed convergence, it has not been shown that the estimated performance gradient is erroneous, nor has it been shown that it is correct. Should the estimated performance gradient be shown to converge sufficiently, then there may be no need for a modified compatible function approximator. Therefore, the following section presents further analysis into the convergence of the estimated performance gradient.

Two different methods will be utilised in order to limit the size of a possible error from the estimated performance gradient. The first method will investigate the error of the performance gradient by calculating the relative and absolute error of the estimate. The second approach will be to utilise a Taylor expansion for the estimation of the moments of functions [15]. From this, the error of the estimate may be bound do a degree n that is proportional with the nth moment.

For the following analysis, it is assumed that the expected value operator is taken across the distribution ρ, $\mathbb{E}_{s\sim\rho} = \mathbb{E}$

Relative error

The relative error may be used to quantified the precision of an estimate. The following will attempt to bound the estimated performance gradient by looking at the relative error between it and the true performance gradient. The relative error between the two gradients is given by

$$\frac{||\nabla_\theta J(\pi_\theta) - \nabla_\theta \hat{J}(\pi_\theta)||}{||\nabla_\theta J(\pi_\theta)||} \tag{3.2.34}$$

For simplicity, the absolute error will be initially calculated, and later transformed to the relative error. Due to norm operations on matrices, for the following results to be valid, the norm must be a submultiplicative induced p-norm. An upper bound on the error of the two vectors may be given by the triangle inequality, applying this and expanding gives

$$\begin{aligned} ||\nabla_\theta J - \nabla_\theta \hat{J}|| = &\Big|\Big| \mathbb{E}[\nabla_\theta \pi_\theta \nabla_a A(s,a)] \\ &- \mathbb{E}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \mathbb{E}[\nabla_\theta \pi_\theta M \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}[\nabla_\theta \pi_\theta M \nabla_a A(s,a)] \Big|\Big|. \end{aligned} \tag{3.2.35}$$

Using the triangle inequality and the submultiplicative property of the norm gives this as

$$\begin{aligned} \leq &||\mathbb{E}[\nabla_\theta \pi_\theta \nabla_a A(s,a)]|| \\ &+ ||\mathbb{E}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top]|| \cdot ||\mathbb{E}[\nabla_\theta \pi_\theta M \nabla_\theta \pi_\theta^\top]^{-1}|| \cdot ||\mathbb{E}[\nabla_\theta \pi_\theta M \nabla_a A(s,a)]||. \end{aligned} \tag{3.2.36}$$

By looking at the norm a nonlinear function transformation that is applied to the expected value operator, then Jensen's inequality, $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$, may

be used to move the norm inside the expected value

$$\begin{aligned} ||(.)|| \leq & \mathbb{E}[||\nabla_\theta \pi_\theta \nabla_a A(s,a)||] \\ & + \mathbb{E}[||\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top||] \cdot \mathbb{E}[||\nabla_\theta \pi_\theta M \nabla_\theta \pi_\theta^\top||]^{-1} \cdot \mathbb{E}[||\nabla_\theta \pi_\theta M \nabla_a A(s,a)||]. \end{aligned} \tag{3.2.37}$$

Similarly, the transformation $\mathbb{E}[f(x)g(x)] \leq \mathbb{E}[f(x)]\mathbb{E}[g(x)]$ may be used to extract the covariance matrix.

$$\begin{aligned} ||(.)|| \leq & \mathbb{E}[||\nabla_\theta \pi_\theta|| \cdot ||\nabla_a A(s,a)||] \\ & \cdot \mathbb{E}[||\nabla_\theta \pi_\theta|| \cdot ||\nabla_\theta \pi_\theta^\top||]^{-1} \cdot \mathbb{E}[||M||]^{-1} \cdot \mathbb{E}[||M||]\mathbb{E}[||\nabla_\theta \pi_\theta|| \cdot ||\nabla_a A(s,a)||], \end{aligned} \tag{3.2.38}$$

which yields

$$||\nabla_\theta J - \nabla_\theta \hat{J}|| \leq \mathbb{E}[||\nabla_\theta \pi_\theta|| \cdot ||\nabla_a A(s,a)||] + \mathbb{E}[||\nabla_\theta \pi_\theta|| \cdot ||\nabla_a A(s,a)||]. \tag{3.2.39}$$

Giving the relative error as

$$\frac{||\nabla_\theta J - \nabla_\theta \hat{J}||}{||\nabla_\theta J||} \leq \frac{2\mathbb{E}[||\nabla_\theta \pi_\theta|| \cdot ||\nabla_a A(s,a)||]}{\mathbb{E}[||\nabla_\theta \pi_\theta|| \cdot ||\nabla_a A(s,a)||]} \tag{3.2.40}$$

$$= 2. \tag{3.2.41}$$

Which provides an upper bound on the error. The factor of 2 stems from utilising the triangle inequality and are expected if the vectors are either identical or opposite. A lower bound may be found in a similar fashion, where the reverse triangle inequality is used instead, giving

$$\Big|\,||\nabla_\theta J|| - ||\nabla_\theta \hat{J}||\,\Big| = 0, \tag{3.2.42}$$

which bounds the relative error between the estimated and modified vector between $[0, 2]$. It is difficult to improve the upper bound any further as long as the triangle inequality is being used, and without any methods that allow for the calculation of the dot product $< \nabla_\theta J, \nabla_\theta \hat{J} >$.

Taylor expansion

As seen in (2.3.16), the expected value of a function may be approximated by a Taylor expansion around its mean. It may be possible to quantify the maximum size of an error in the estimated performance gradient through its Taylor expansion. Consider yet again the estimated performance gradient,

$$\begin{aligned} \nabla_\theta \hat{J}(\pi_\theta) = & \mathbb{E}[\nabla_\theta \pi_\theta(s) \nabla_\theta \pi_\theta(s)^\top] \mathbb{E}[\nabla_\theta \pi_\theta(s) M \nabla_\theta \pi_\theta(s)]^{-1} \\ & \cdot \mathbb{E}[\nabla_\theta \pi_\theta(s) M \nabla_a A(s, \pi_\theta)], \end{aligned} \tag{3.2.43}$$

where s is a stochastic variable with an unknown density $\rho(s)$. Furthermore, $\nabla_\theta \pi_\theta(s)$ is an unknown function with full rank. Applying the Taylor expansion

to (3.2.43) yields

$$\begin{aligned}\nabla_\theta \hat{J}(\pi_\theta) =& \mathbb{E}[\nabla_\theta \pi_\theta(\mu_s) \nabla_\theta \pi_\theta(\mu_s)^\top + \epsilon_1] && (3.2.44)\\ & \cdot \mathbb{E}[(\nabla_\theta \pi_\theta(\mu_s) M \nabla_\theta \pi_\theta(\mu_s))^{-1} + \epsilon_2] \\ & \cdot \mathbb{E}[\nabla_\theta \pi_\theta(\mu_s) M \nabla_a A(\mu_s, \pi_\theta) + \epsilon_3] \\ =& \nabla_\theta \pi_\theta(\mu_s) \nabla_\theta \pi_\theta(\mu_s)^\top \left(\nabla_\theta \pi_\theta(\mu_s) M \nabla_\theta \pi_\theta(\mu_s)\right)^{-1} && (3.2.45)\\ & \cdot \nabla_\theta \pi_\theta(\mu_s) M \nabla_a A(s, \pi_\theta) + \epsilon_{\nabla_\theta \hat{J}} \\ =& \nabla_\theta \pi_\theta(\mu_s) \nabla_a A(\mu_s, \pi_\theta) + \epsilon_{\nabla_\theta \hat{J}}, && (3.2.46)\end{aligned}$$

where $\epsilon_{\nabla_\theta \hat{J}}$ is a second order error term proportional to both the combined curvature of the three expected value terms, and to the covariance of the state trajectory $\mathcal{S}_n$. In addition, the transition from (3.2.45) to (3.2.46) is possible by similarly utilising the SVD as in(3.2.27).

The Taylor approximation of the true gradient is given by

$$\begin{aligned}\nabla_\theta J(\pi_\theta) &= \mathbb{E}[\nabla_\theta \pi_\theta(s) \nabla_a A(s, \pi_\theta)] && (3.2.47)\\ &= \nabla_\theta \pi_\theta(\mu_s) \nabla_a A(\mu_s, \pi_\theta) + \epsilon_{\nabla_\theta J}, && (3.2.48)\end{aligned}$$

where $\epsilon_{\nabla_\theta J}$ is proportional to the covariance of the state trajectory $\mathcal{S}_n$, and to the curvature of $f(s) = \nabla_\theta \pi_\theta(s) \nabla_a A(s, \pi_\theta)$. With the Taylor approximations of both the estimated and true performance gradient, the error from the estimated performance gradient may be written as:

$$\nabla_\theta J(\pi_\theta) - \nabla_\theta \hat{J}(\pi_\theta) = \epsilon_{\nabla_\theta \hat{J}} - \epsilon_{\nabla_\theta J}, \tag{3.2.49}$$

with the error being proportional to both the covariance of the state trajectory and the curvature of the functions, $\epsilon_{\nabla_\theta \hat{J}}, \epsilon_{\nabla_\theta J} \propto \Sigma_s \Sigma_s^\top$. Should the state trajectory and its distribution ρ satisfy $\lim_{n\to\infty} \sqrt{n}(\mathcal{S}_n - \mu_s) \to \mathcal{N}(0, \Sigma\Sigma^\top)$, that is the Central Limit Theorem (CLT), where $\mathcal{S}_n$ denotes state trajectory, and if $\Sigma\Sigma^\top \to 0$, then, via arguments from the delta method, it may be shown that the error terms are reduced to zero as the sample size n increases, as shown in section 2.3.3, giving

$$\nabla_\theta \hat{J}(\pi_\theta) - \nabla_\theta J(\pi_\theta) \xrightarrow[\Sigma_s \Sigma_s^\top \to 0]{s \to \text{i.i.d} \wedge \nabla_a A(\mu_s) \neq 0} 0. \tag{3.2.50}$$

More generally, from the Taylor expansion, it may be guaranteed that any possible errors in the estimated performance gradient are small in systems with a small state trajectory covariance and/or curvature in the advantage function and policy gradient.

3.3 Experiments

Two different experiments will be performed in order to test the statements presented in sections 3.2.3. In this section, the setup of the experiment; environment, system, controller, and RL algorithms, are initially presented. In addition, the results from the experiments are also presented in the end.

3.3.1 System dynamics

In order to illustrate the results, a simple and completely known system must be utilised. Consider the following true linear state dynamics,

$$s_{k+1} = As_t + Ba_t, \quad A = \begin{bmatrix} 0.7 & 0.3 \\ 0 & 1.1 \end{bmatrix}, \quad B = \begin{bmatrix} 0.1 & 0.2 \\ 0.1 & 0.2 \end{bmatrix}, \tag{3.3.1}$$

with a stage penalty defined as

$$L(s_k, a_k) = \begin{bmatrix} s \\ a \end{bmatrix}^\top \begin{bmatrix} T & N \\ N^\top & R \end{bmatrix} \begin{bmatrix} s \\ a \end{bmatrix}, \quad T = \begin{bmatrix} 8 & 0 \\ 0 & 8 \end{bmatrix}, \quad R = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}. \tag{3.3.2}$$

The policy may be written on the form

$$\pi_\theta(s) = -Ks. \tag{3.3.3}$$

In a linear quadratic system, the value functions become quadratic as well and are given as:

$$V_{\pi_\theta}(s_t) = s_t P s_t, \tag{3.3.4}$$

$$Q_{\pi_\theta}(s_t, a_t) = \begin{bmatrix} s \\ a \end{bmatrix}^\top \begin{bmatrix} T + \gamma A^\top PA & N + \gamma A^\top PB \\ N^\top + \gamma B^\top PA & R + \gamma B^\top PB \end{bmatrix} \begin{bmatrix} s \\ a \end{bmatrix}, \tag{3.3.5}$$

giving an advantage function as

$$A_{\pi_\theta}(s, a) = Q_{\pi_\theta}(s, a) - V_{\pi_\theta}(s) \tag{3.3.6}$$

$$= \begin{bmatrix} s \\ a \end{bmatrix}^\top \begin{bmatrix} T - P + \gamma A^\top PA & N + \gamma A^\top PB \\ N^\top + \gamma B^\top PA & R + \gamma B^\top PB \end{bmatrix} \begin{bmatrix} s \\ a \end{bmatrix}, \tag{3.3.7}$$

where P is found by utilising that

$$V_{\pi_\theta}(s_t) = L(s_t, \pi_\theta) + \gamma V_{\pi_\theta} V(s_{t+1}) \tag{3.3.8}$$

$$s_t^\top P s_t = \begin{bmatrix} s_t \\ -Ks_t \end{bmatrix}^\top \begin{bmatrix} T & N \\ N^\top & R \end{bmatrix} \begin{bmatrix} s_t \\ -Ks_t \end{bmatrix} + (As_t + B\pi_\theta(s_t))^\top P(As_t + B\pi_\theta(s_t)) \tag{3.3.9}$$

$$\Rightarrow P = T - 2NK + K^\top RK + (A - BK)^\top P(A - BK). \tag{3.3.10}$$

The derivative of the advantage function becomes

$$\nabla_a A(s, a) = 2(N + \gamma A^\top PB)s + (R + 2\gamma B^\top PB + R^\top)a, \tag{3.3.11}$$

giving the true performance gradient as

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_a A_{\pi_\theta}(s,a)]. \tag{3.3.12}$$

The policy that will be used is given as the solution to the following MPC problem

$$\begin{aligned} \pi(s) = \min_{x,u} \quad & x_N^\top \hat{P}_\theta x_n + \sum_{k=0}^{N-1} x_k^\top \hat{T}_\theta x_k + u_k \hat{R}_\theta u_k \\ \text{s.t} \quad & \\ & x_{k+1} = Ax_k + Bu_k \\ & x_0 = s \end{aligned} \tag{3.3.13}$$

where the matrices A and B are defined as above, the cross-matrix N is a zero matrix, and the matrices $\hat{T}_\theta$ and $\hat{R}_\theta$ are defined as

$$\hat{T}_\theta = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}, \quad \hat{R}_\theta = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}. \tag{3.3.14}$$

Additionally, P_θ was initiated as the resulting Ricatti-matrix to the pairs $\{A, B, \hat{T}_\theta, \hat{R}_\theta\}$. The horizon limit N is set to $N = 10$. The exploration is created through a normal distribution centred at π_θ with a standard-deviation given as a non-symmetric non-diagonal matrix, such that the exploration is anisotropic.

$$\Sigma_1 = \begin{bmatrix} 0.3 & 0.05 \\ 0 & 0.1 \end{bmatrix} \tag{3.3.15}$$

$$\Sigma_1 \Sigma_1^\top = \begin{bmatrix} 0.0925 & 0.005 \\ 0.005 & 0.01 \end{bmatrix}. \tag{3.3.16}$$

The starting state s_0 for each episode is selected as a normal distribution around $\begin{bmatrix} 0.3 & 0.4 \end{bmatrix}^\top$

$$s_0 = \begin{bmatrix} 0.3 \\ 0.4 \end{bmatrix} + \mathcal{N}(0, 0.04 * \mathbb{I}). \tag{3.3.17}$$

From the actor critic method, the estimated performance gradient to be used in the experiments may then be written as

$$\begin{aligned} \nabla_\theta \hat{J}(\pi_\theta) = & \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \\ & \cdot \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e A_{\pi_\theta}(s_t, a_t)]. \end{aligned} \tag{3.3.18}$$

The modified performance gradient is calculated as

$$\begin{aligned} \nabla_\theta \bar{J}(\pi_\theta) = & \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top] \\ & \cdot \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta (\Sigma\Sigma^\top)^{-1} e e^\top (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top]^{-1} \\ & \cdot \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e (\Sigma\Sigma^\top)^{-1} A_{\pi_\theta}(s_t, a_t)]. \end{aligned} \tag{3.3.19}$$

Finally, it is important to note that both $\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]$ and $\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta (\Sigma\Sigma^\top)^{-1} e e^\top (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top]^{-1}$ are lacking rank, as mentioned in (3.2.24). Some workarounds that enable the calculation of the inverse exists, such as the Sherman-Morrison formula; however, this adds complexity in computation. The method that was utilised in these experiments was a simple addition of a scaled identity matrix $c\mathbb{I}, c << 1$. This introduces some numerical inaccuracies that may affect the results.

3.3.2 Modified and estimated performance gradient

With the above system, the performance gradients, (3.3.12),(3.3.18),(3.3.19), were calculated from a batch consisting of 1000 episodes. The results are presented in Figure 3.4. Some interesting observations: After 1000 episodes, the absolute difference between the estimated and true performance gradient is given as $||\nabla_\theta J(\pi_\theta) - \nabla_\theta \hat{J}(\pi_\theta)|| = 0.0006$. Furthermore, the modified and estimated performance gradient are almost identical during the complete 1000 episodes. An extended plot of the relative error between the modified and estimated performance gradient is given in Figure 3.5, where the relative error stabilises at around 0.001.

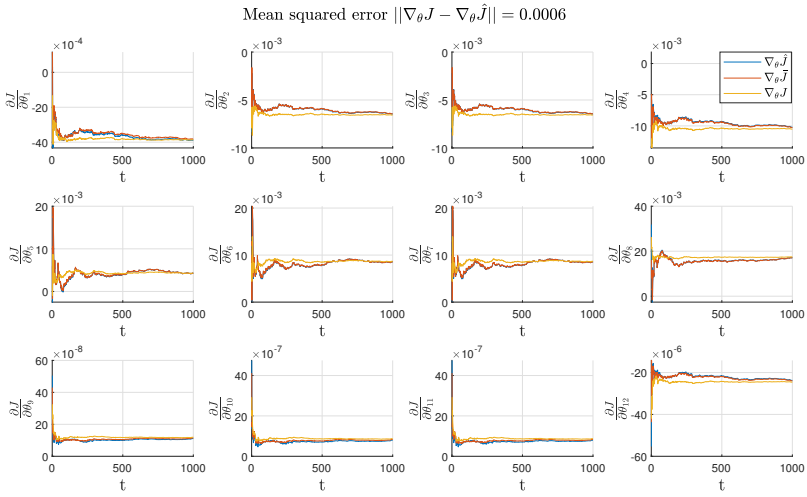


Figure 3.4: The evolution of the performance gradients across 1000 episodes

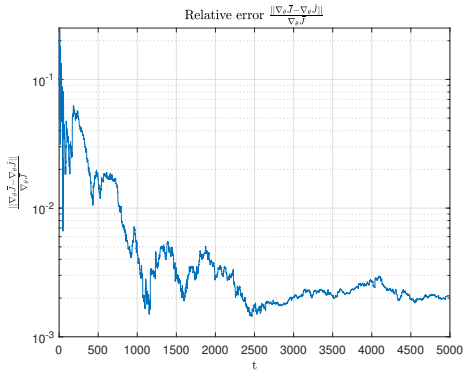


Figure 3.5: Extended batch across 5000 episodes showing the convergence of the relative error towards zero

3.3.3 Taylor approximation of the performance gradient

The Taylor approximation has an error proportional to both the covariance of the state trajectory, and to the curvature of the functions in the expected value terms. In order to verify that the error is proportional to the covariance of the state trajectory, the previously mentioned curvature may not be zero. This becomes an issue if the state trajectory has a mean $\mu_s = 0$, where the policy $\pi_\theta(0)$ delivers the same actions as the optimal policy π_θ^*. In the region around $s = 0$, the policy is therefore approximately optimal, which gives the advantage function and its derivatives as zero $\nabla_a A(0, \pi_\theta) = 0$ or very small $\nabla_a A(\delta, \pi_\theta) << 1$, where delta is a small deviation from $s = 0$.

If the experiments should be able to test proportionality with the covariance, the optimum point of the state must be shifted away from $s = 0$ in order to have a non-zero curvature. Any area where $s \neq 0$ suffice, however, a "large" shift is justified in order to avoid excessively small numbers in the results. A shift of optimum may be accomplished by introducing a reference in the policy and reward functions, giving the shifted policy and reward functions as:

$$\begin{aligned}\pi(s) = \min_{x,u} \quad & (x_N - s_{\text{ref}})^\top \hat{P}_\theta (x_N - s_{\text{ref}}) + \sum_{t=0}^{N-1} (x_t - s_{\text{ref}})^\top \hat{T}_\theta (x_t - s_{\text{ref}}) \\ & + u_k \hat{R}_\theta u_k \\ \text{s.t} \quad & \\ & x_{k+1} = A x_k + B u_k \\ & x_0 = s\end{aligned} \tag{3.3.20}$$

$$L(s_k, a_k) = \begin{bmatrix} s - s_{\text{ref}} \\ a \end{bmatrix}^\top \begin{bmatrix} T & N \\ N^\top & R \end{bmatrix} \begin{bmatrix} s - s_{\text{ref}} \\ a \end{bmatrix} \tag{3.3.21}$$

The reference point for the system is set to $s_{\text{ref}} = \begin{bmatrix} 0.4 & 0.4 \end{bmatrix}^\top$. The covari-

ance of the state trajectory is changed through increasing the covariance of the starting positions s_0. In the first experiment, $s_0 = \mathcal{N}(\begin{bmatrix}0.3 & 0.4\end{bmatrix}^\top, 0.05^2\mathbb{I})$, of which results are presented in Figure 3.7, whereas in the second simulation $s_0 = \mathcal{N}(\begin{bmatrix}0.3 & 0.4\end{bmatrix}^\top, 0.5^2\mathbb{I})$, of which the results are presented in Figure 3.8. Key metrics from the two experiments are given in Table 3.1. Finally, the exploration policy β may be seen in Figure 3.6

Table 3.1: Key numerical results from the experiments regarding Taylor expansion.

metric	Simulation 1	Simulation 2
$\|\|\epsilon_{\nabla_\theta J}\|\|$	0.0051	0.0301
$\|\|\epsilon_{\nabla_\theta \hat{J}}\|\|$	0.0046	0.0334
$\|\|\text{cov}(s)\|\|$	0.0051	0.0340
$\|\|\epsilon_{\nabla_\theta J} - \epsilon_{\nabla_\theta \hat{J}}\|\|$	0.0017	0.0120
μ_s	$\begin{bmatrix}0.289 & 0.423\end{bmatrix}^\top$	$\begin{bmatrix}0.291 & 0.424\end{bmatrix}^\top$

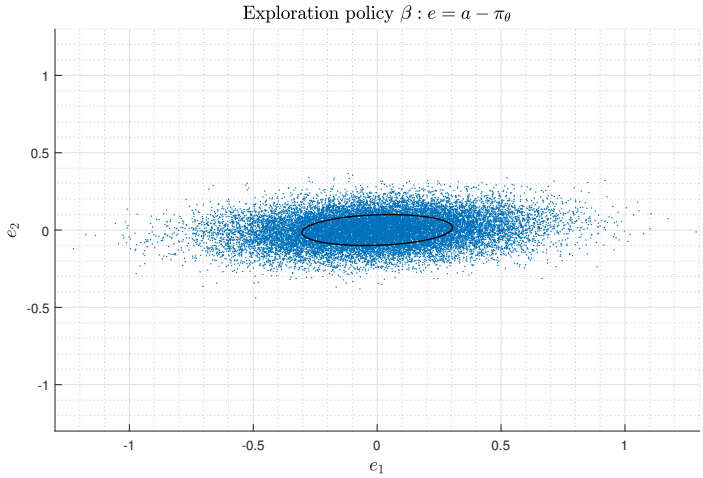
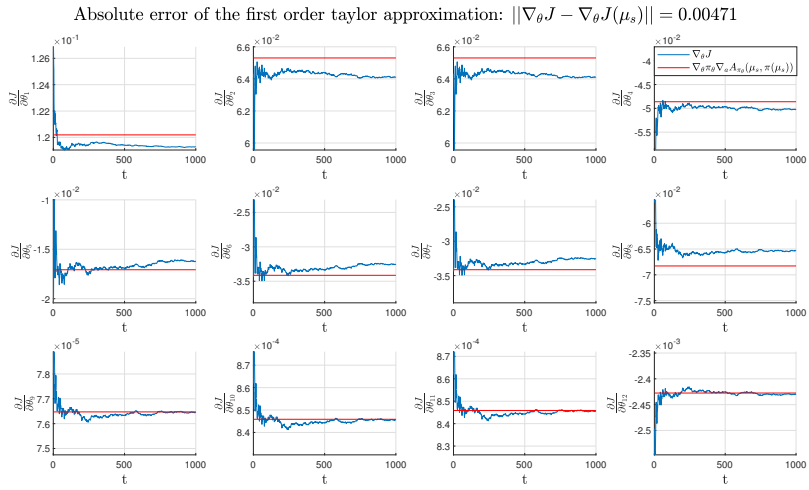
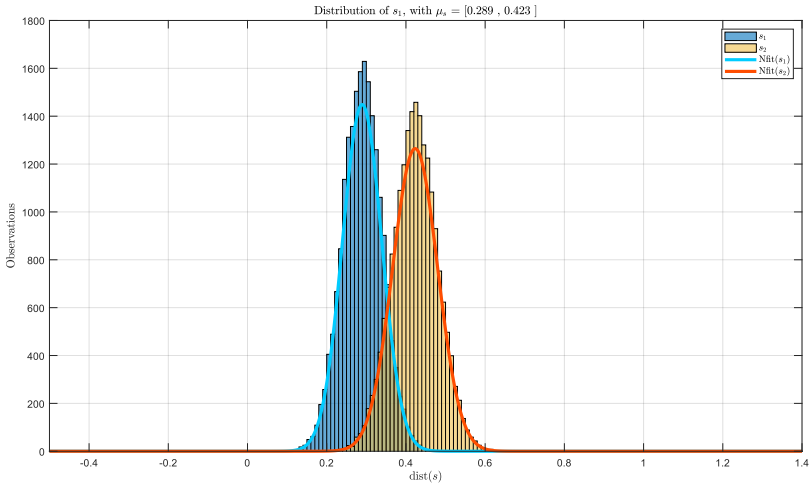


Figure 3.6: The exploration utilised in all experiments. The covariance is anisotropic and centred, with exploration prioritising the first action dimension a_1 above a_2. Sampled covariance gives $\Sigma\Sigma^\top = \begin{bmatrix}0.0925 & 0.0050\\ 0.0050 & 0.0100\end{bmatrix}$.

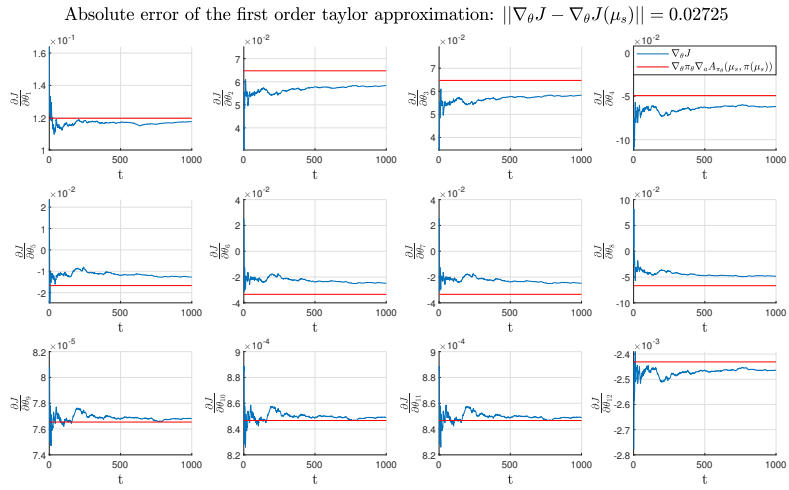


(a) The evolution of the performance gradients across 1000 episodes

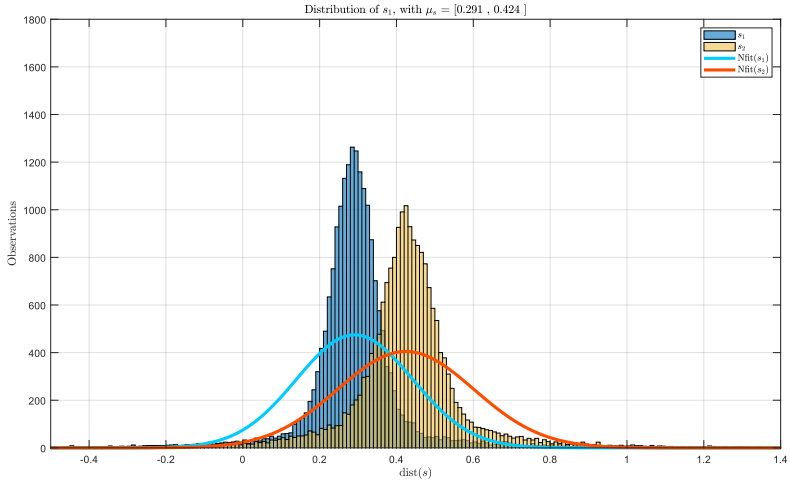


(b) The distribution of state s_1 across the batch

Figure 3.7: The Taylor approximation and true performance gradient, with the optimal position given as $s_{\text{ref}} = \begin{bmatrix}0.4 & 0.4\end{bmatrix}^\top$. Initial conditions were chosen as $s_0 = \mathcal{N}(\begin{bmatrix}0.3 & 0.4\end{bmatrix}^\top, 0.05^2\mathbb{I})$. Showing (a) the true gradient vs the predicted gradient from a first order Taylor expansion, and (b) the distribution of the state trajectories, with them having a covariance $\text{cov}(\mathcal{S}_n) = \begin{bmatrix}0.0025 & 0.0024\\ 0.0024 & 0.0032\end{bmatrix}$



(a) The evolution of the performance gradients across 1000 episodes



(b) The distribution of state s_1 across the batch

Figure 3.8: The Taylor approximation and true performance gradient, with optimal position $s_{\text{ref}} = \begin{bmatrix}0.4 & 0.4\end{bmatrix}^\top$. Initial conditions were chosen as $s_0 = \mathcal{N}(\begin{bmatrix}0.3 & 0.4\end{bmatrix}^\top, 0.5^2\mathbb{I})$. Showing (a) the true gradient vs the predicted gradient from a first order Taylor expansion, and (b) the distribution of the state trajectories, with them having a covariance $\text{cov}(\mathcal{S}_n) = \begin{bmatrix}0.0229 & -0.0003\\ -0.0003 & 0.0315\end{bmatrix}$

3.4 Discussion

This section presents a brief discussion about the experiments and their validity. In general, the most likely source of errors in the experiments would be the insufficient rank inverse. The experiments are also linear quadratic systems, and the results do not necessarily extend to more complex systems.

3.4.1 Modified and estimated performance gradient

The experiments support the convergence analysis that is presented in section 3.2.3. In Figure 3.4, the estimated and modified performance gradient are equal, and both approach the true performance gradient. The convergence of the modified performance gradient coincides with the theoretical results from (3.2.31) - (3.2.33).

The relative error between the estimated and modified performance gradient was shown in (3.2.41) and (3.2.42) to have an upper and lower bound of $[2, 0]$. In Figure 3.4, it is clear that the estimated and modified performance gradient is approximately equal. An extended plot in Figure 3.5 with an additional simulation of 5000 episodes shows that the relative error $\frac{||\nabla_\theta \bar{J} - \nabla_\theta \hat{J}||}{||\nabla_\theta \bar{J}||}$ approaches 0, stabilizing at ~ 0.002. One substantial error source may be due to $\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1}$ lacking rank, and may be a reason that the relative error is not smaller. A more proper method e.g. *Sherman-Morrisons* formula, may have gotten improved results. With this in mind, it is interesting to note that the estimated performance gradient was consistently a slightly better estimate than the modified gradient, with the absolute error in the experiment presented here being $||\nabla_\theta \hat{J} - \nabla_\theta J|| = 0.00090$, whereas $||\nabla_\theta \bar{J} - \nabla_\theta J|| = 0.0011$. This discrepancy is however most likely due to numerical errors.

It is important to note that in these experiments, the estimated performance gradient was an exact estimate, to the same degree as the modified performance gradient, of the true performance gradient. These results were gained with a fairly anisotropic exploration policy, as evident in Figure 3.6. It is also interesting that the estimated performance gradient had an almost identical convergence trajectory as the modified performance gradient, which is an indication that the addition of the inverse covariance has no apparent effect on the convergence. Therefore, the experiments suggest, that in linear quadratic systems, the addition of the inverse covariance to the compatible function approximator has little to no effect on the convergence of the estimated performance gradient.

3.4.2 Taylor approximation of the performance gradient

In the Taylor analysis, the error of the expansion was shown to be proportional to the covariance of the state trajectory. The experiments support this notion, as from the metrics in Table 3.1 it is clear that the error of the Taylor expansions

are almost directly proportional to the covariance of the state trajectory. Furthermore, the absolute error of the rest terms $||\epsilon_{\nabla_\theta J} - \epsilon_{\nabla_\theta \hat{J}}||$ is small, indicating that the rest terms $\epsilon_{\nabla_\theta \hat{J}}$ and $\epsilon_{\nabla_\theta J}$ are approximately equal. These results are also visualized in Figure 3.7 (a) and Figure 3.8 (a), where the first order Taylor approximation $\nabla_\theta \pi_\theta \nabla_a A_{\pi_\theta}(\mu_s, \pi_\theta(\mu_s))$ loses accuracy directly proportional to the covariance of the state trajectory $\mathcal{S}_n$.

These experiments support the notion that the estimated performance gradient is accurate to a degree proportional to the covariance of the state trajectory. The most probable error in these experiments may again be numerical inaccuracies due to the insufficient rank inverse of $\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1}$.

IV Summary

This chapter concludes the thesis. It starts with a review of the results and findings that have been presented throughout this thesis. Thereafter, precise conclusions for the research questions are made before the chapter finalises with a short discussion about future work.

4.1 Review

The research objective for this thesis was primarily to investigate the combination of predictive control algorithms and Reinforcement Learning (RL), in addition to furthering the research within this area. The focus for the research was based upon the estimated policy gradient proposed method in [5], and the goal was to provide some looser constraints on the possible methods for exploration such that it becomes more applicable in safe RL settings. More specifically, the effects of non-isotropic covariances in the exploration policy were investigated.

The research area combined two different disciplines and the second chapter provided some key foundations for both the disciplines RL and predictive control. These concepts served as the necessary basic level of theory that was required in order to work on the research objective. In addition, the second chapter also presented some statistical results that were necessary for some of the key results.

The first research question was to properly understand and investigate the implementation of Nonlinear Model Predictive Controller (NMPC) into RL, of which are based upon the work by [2] and was presented in section 3.1. This section started by reasoning briefly for why an NMPC scheme could be used, before proceeding to the details of the two algorithms. The second algorithm that was presented, policy gradient with NMPC, were later used in the next section for both experiments and results.

The second research question was to investigate how exploration affects the policy gradient algorithm. It started by discussing why the traditional constraints on the exploration policy are too restrictive in the sense of safe RL, making it clear that lesser restrictions are required. This discussion is aided by a brief example that visualises that an anisotropic exploration scheme is inevitable in safe RL.

An analytical evaluation of the estimated performance gradient was performed with the purpose of exploring the exact consequences of any normally distributed exploration scheme. This yielded the equations (A.4) - (A.6), only dependent on the state trajectory.

Based upon (A.4) - (A.6), a general exploration scheme with isotropic exploration was analysed. It was shown that with such an exploration, the estimated performance gradient converges to the true performance gradient (3.2.19).

Furthermore, an exploration policy with anisotropic covariance was considered. In this case, any immediate conclusions could not be drawn; however, a possible modification of the compatible function approximator became apparent. With this modification, the performance gradient was shown to converge to the true performance gradient. These modifications have previously been discussed in [8], where the same conclusions have been drawn.

Further analysis of the estimated performance gradient was warranted, as

while the modified compatible function approximator was shown theoretically to converge, it still introduced additional complexity to the algorithm. The first approach was to explore the relative error between the estimated and true performance gradient. Through utilising the triangular identity, among others, it was shown that an upper and lower bound on the relative error of the estimated performance gradient were $[0, 2]$. This is not exact, and a gradient pointing in the complete opposite direction do comply with these results. Further analysis with a different method was justified.

With a Taylor expansion, it was shown, via arguments from the delta method, that the estimate converges to the true gradient, provided that the state covariance goes to zero (3.2.50). Furthermore, the Taylor expansion shows that the error of the estimated policy gradient is at most proportional to the covariance of the state trajectory. This was also confirmed via experiments.

Experiments with a linear quadratic system were performed in order to illustrate the results. In Figure 3.4, it was shown that both the modified and estimated performance gradient converges to the true performance gradient under an anisotropic exploration scheme. In Figure 3.4, it is also clear that the modified and estimated performance gradient are equal, further indicating the convergence of the estimated performance gradient.

A second experiment was performed where it is illustrated that the estimated performance gradient has an error at most proportional to the covariance of the state trajectory.

4.2 Conclusion

How are NMPC integrated with RL?

NMPC may be integrated as a function approximator in both the traditional RL algorithms Q-learning (3.1.7) and policy gradient (3.1.4). With the combination of slack variables in the NMPC, the RL algorithms may learn to respect safety-critical constraints. Furthermore, with strict constraints, the NMPC may generate safe exploration steps that guarantee the respect of the constraints. As such, the NMPC has proper potential in the field of safe RL.

Does the estimated policy gradient depend on the shape of the exploration?

The research provides theoretical ground that any error in the policy gradient from utilising anisotropic covariance in the exploration is guaranteed to be at most proportional with the covariance of the state trajectory and to the curvature of the estimated performance gradient. Furthermore, it is guaranteed that the norm of the estimated performance gradient is no larger than twice that of the norm of the true gradient. Finally, experiments strongly suggest that the policy gradient is independent of anisotropic exploration.

4.3 Further work

The experiments in this thesis provide strong evidence that the estimated performance gradient converges to the true performance gradient regardless of anisotropic exploration. This proved however to be challenging to show theoretically, and further analysis in possible errors from anisotropic exploration is of interest. An expansion of the Taylor analysis appears most appropriate. It is proposed that any further work may investigate the error from the first order Taylor expansion through means provided by the Jensen's gap: $f(\mathbb{E}[\chi]) - \mathbb{E}(f(\chi))$[16][17]. The Jensen gap may provide further bounds on the error.

Furthermore, safe RL also introduces off-centred exploration, which has not been discussed to a large degree in this thesis. However, analytical evaluation of the estimated performance gradient does provide some preliminary results that off-centred exploration does not converge to the true performance gradient. These results are provided in Appendix C. It is possible to counter off-centred exploration by approximating the exploration mean, as shown in [8]. However, this demands more computing power and a more complex algorithm. Therefore, it might be beneficial to further analyse the results provided in Appendix C.

The results in appendix C show that any bias will ensure that the estimated policy gradient is not exact; however, it may be possible to show that the error decreases as the parameters approach optimal values. Through line-search methods, it is shown that the gradient in the gradient descent method is reduced to zero if the estimated gradient is not orthogonal to the gradient, if also the step length is chosen as per line search methods. These results may be a starting point for an investigation of the consequences of off-centred exploration.

Bibliography

[1] Javier García, Fern, and o Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(42):1437–1480, 2015.

[2] Sebastien Gros and Mario Zanon. Data-driven economic nmpc using reinforcement learning. *IEEE Transactions on Automatic Control*, PP:1–1, 04 2019.

[3] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[4] A. Nowé, P. Vrancx, and Y-M. De Hauwere. *Reinforcement Learning: State-of-the-Art*, chapter Game Theory and Multi-agent Reinforcement Learning, pages 441–470. Springer, 2012.

[5] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. *31st International Conference on Machine Learning, ICML 2014*, 1, 06 2014.

[6] G. Rummery and Mahesan Niranjan. On-line q-learning using connectionist systems. *Technical Report CUED/F-INFENG/TR 166*, 11 1994.

[7] Richard Sutton, David Mcallester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Adv. Neural Inf. Process. Syst*, 12, 02 2000.

[8] Sebastien Gros and Mario Zanon. Towards safe reinforcement learning using NMPC and policy gradients: Part II - deterministic case. *CoRR*, abs/1906.04034, 2019.

[9] G.C. Chow. *Analysis and Control of Dynamic Economic Systems*. A Wiley publication in applied statistics. Wiley, 1975.

[10] F.L. Lewis, V.L. Syrmos, and V.L. Syrmos. *Optimal Control*. A Wiley-interscience publication. Wiley, 1995.

[11] Lars Grne and Jrgen Pannek. *Nonlinear Model Predictive Control: Theory and Algorithms*. Springer Publishing Company, Incorporated, 2013.

[12] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, second edition, 2006.

[13] Alex Papanicolaou. Taylor approximation and the delta method, 2009.

[14] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning, 2019.

[15] Gustaf Hendeby and Fredrik Gustafsson. On nonlinear transformations of gaussian distributions, 01 2003.

[16] Xiang Gao, Meera Sitharam, and Adrian E. Roitberg. Bounds on the jensen gap, and implications for mean-concentrated distributions, 2017.

[17] J. G. Liao and Arthur Berg. Sharpening jensen's inequality, 2017.

Appendices

A Expected value calculation

The estimated policy gradient is given as

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e A_{\pi_\theta}(s_t, a_t)] \tag{A.1}$$

where the expected value is taken across the two random variables s and a. s is sampled from the state trajectory with the complex distribution ρ. Due to the distribution being difficult to evaluate in general, it is not possible to analytically evaluate the expected value above in terms of the state trajectory $\mathcal{S}_n$. However, the actions a is a known distribution with expected value dependent on s. This allows for the possibility to analytically evaluate the expected value in terms of a, giving the end result as an expected value taken purely across the state trajectory. Assuming that a is normally distributed with some mean μ and variance $\Sigma\Sigma^\top$, $a \sim \mathcal{N}(\mu, \Sigma\Sigma^\top)$, and with an advantage function given as

$$A_{\pi_\theta} = \begin{bmatrix} s_t \\ a_t \end{bmatrix}^\top W \begin{bmatrix} s_t \\ a_t \end{bmatrix} + F^\top \begin{bmatrix} s_t \\ a_t \end{bmatrix} + C \tag{A.2}$$

$$= \begin{bmatrix} s_t \\ a_t \end{bmatrix}^\top \begin{bmatrix} W_{1,1} & W_{1,2} \\ W_{2,1} & W_{2,2} \end{bmatrix} \begin{bmatrix} s_t \\ a_t \end{bmatrix} + \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}^\top \begin{bmatrix} s_t \\ a_t \end{bmatrix} + C, \tag{A.3}$$

then the expected values may be evaluated. With $e = a - \pi_\theta$, there are three orders of moments that need to be evaluated, of which the procedure are shown in sections 2.3.1 and 2.3.2. The expected values are therefore reduced to:

$$\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] = \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \tag{A.4}$$

$$\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top] = \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \Sigma\Sigma^\top \nabla_\theta \pi_\theta^\top] \tag{A.5}$$

$$\begin{aligned} \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e A_{\pi_\theta}(s_t, a_t)] = \mathbb{E}_{s\sim\rho}\Big[\nabla_\theta \Big(& \mu s^\top W_{1,1} s + (\mu\mu^\top + \Sigma\Sigma^\top) W_{2,1} s \\ & + (\mu\mu^\top + \Sigma\Sigma^\top) W_{1,2}^\top s + \mu\mu^\top W_{2,2} \mu \\ & + \Sigma\Sigma^\top (W_{2,2} + W_{2,2}^\top)\mu + \mu \mathrm{Tr}(\Sigma W_{2,2} \Sigma^\top) + \mu s^\top F_1 \\ & + (\Sigma\Sigma^\top + \mu\mu^\top) F_2 + \mu C \Big) - \nabla_\theta \pi_\theta \pi_\theta \Big(s^\top W_{1,1} s \\ & + \mu^\top W_{2,1} s + s^\top W_{1,2} \mu + \mu^\top W_{2,2} \mu + \mathrm{Tr}(\Sigma W_{2,2} \Sigma^\top) \\ & + s^\top F_1 + \pi_\theta^\top f_2 \ + C \Big)] \end{aligned} \tag{A.6}$$

where the identities presented in sections 2.3.1 and 2.3.2 have been used to calculate the higher order moments. In addition, it has been used that a normally distributed variable $\chi \sim \mathcal{N}(\mu, \Sigma\Sigma^\top)$ may be written as $\chi = \mu + \mathcal{N}(0, \Sigma\Sigma^\top)$. This is useful due to the skew of a centred normal distribution are zero, and allows for the expression to be derived purely in terms of the mean and variance. A detailed calculation is not provided, as it is extensive. However, all the methods and tools used to derive the results are presented in section 2.3.1 and 2.3.2.

B Modified compatible function approximator

With the modified compatible function approximator given as

$$\bar{Q}_w(s,a) = (a - \pi_\theta(s))^\top (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top \omega + \hat{V}_\nu(s), \tag{B.1}$$

then the least squares solution for critics parameters ω, with $e = a - \pi_\theta(s)$, is given by

$$0 = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [(r + \gamma \hat{Q}_\omega(s_{t+1}, \pi_\theta(s_{t+1})) - \hat{Q}_\omega(s_t, a_t)) \nabla_\omega \hat{Q}_\omega(s_t, a_t)] \tag{B.2}$$

$$0 = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [(r + \gamma \hat{V}_\nu(s_{t+1}) - e^\top (\Sigma\Sigma^\top)^{-1} \nabla_\theta \pi_\theta^\top \omega + \hat{V}_\nu(s)) \nabla_\theta \pi_\theta M^{-1} e] \tag{B.3}$$

Using $M = (\Sigma\Sigma^\top)^{-1}$, then the least squares solution for ω becomes

$$\omega = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [\nabla_\theta \pi_\theta M e e^\top M \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}[\nabla_\theta \pi_\theta M^{-1} e A(s,a)] \tag{B.4}$$

The addition of the constant matrix M does not affect the calculations provided in appendix A. This gives the modified performance gradient as

$$\nabla_\theta \bar{J}(\pi_\theta) = \mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}} [\nabla_\theta \pi_\theta M \nabla_\theta \pi_\theta^\top] \mathbb{E}[\nabla_\theta \pi_\theta M e e^\top M \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}[\nabla_\theta \pi_\theta M e A(s,a)] \tag{B.5}$$

$$= \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta M \nabla_\theta \pi_\theta^\top] \mathbb{E}[\nabla_\theta \pi_\theta M M^{-1} M \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}[\nabla_\theta \pi_\theta M M^{-1} \nabla_a A(s,a)] \tag{B.6}$$

$$= \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta M \nabla_\theta \pi_\theta^\top] \mathbb{E}[\nabla_\theta \pi_\theta M \nabla_\theta \pi_\theta^\top]^{-1} \mathbb{E}[\nabla_\theta \pi_\theta \nabla_a A(s,a)] \tag{B.7}$$

C Off-centred exploration

The analytical evaluation of the estimated performance gradient also makes it possible to investigate what happens when utilizing an exploration policy that is not centred at π_θ. Therefore, consider now the following exploration policy

$$\beta : e = a - \pi_\theta, \quad a \sim \mathcal{N}(\pi_\theta + \delta, \Sigma\Sigma^\top), \quad \Rightarrow \quad e \sim \mathcal{N}(\delta, \Sigma\Sigma^\top), \tag{C.1}$$

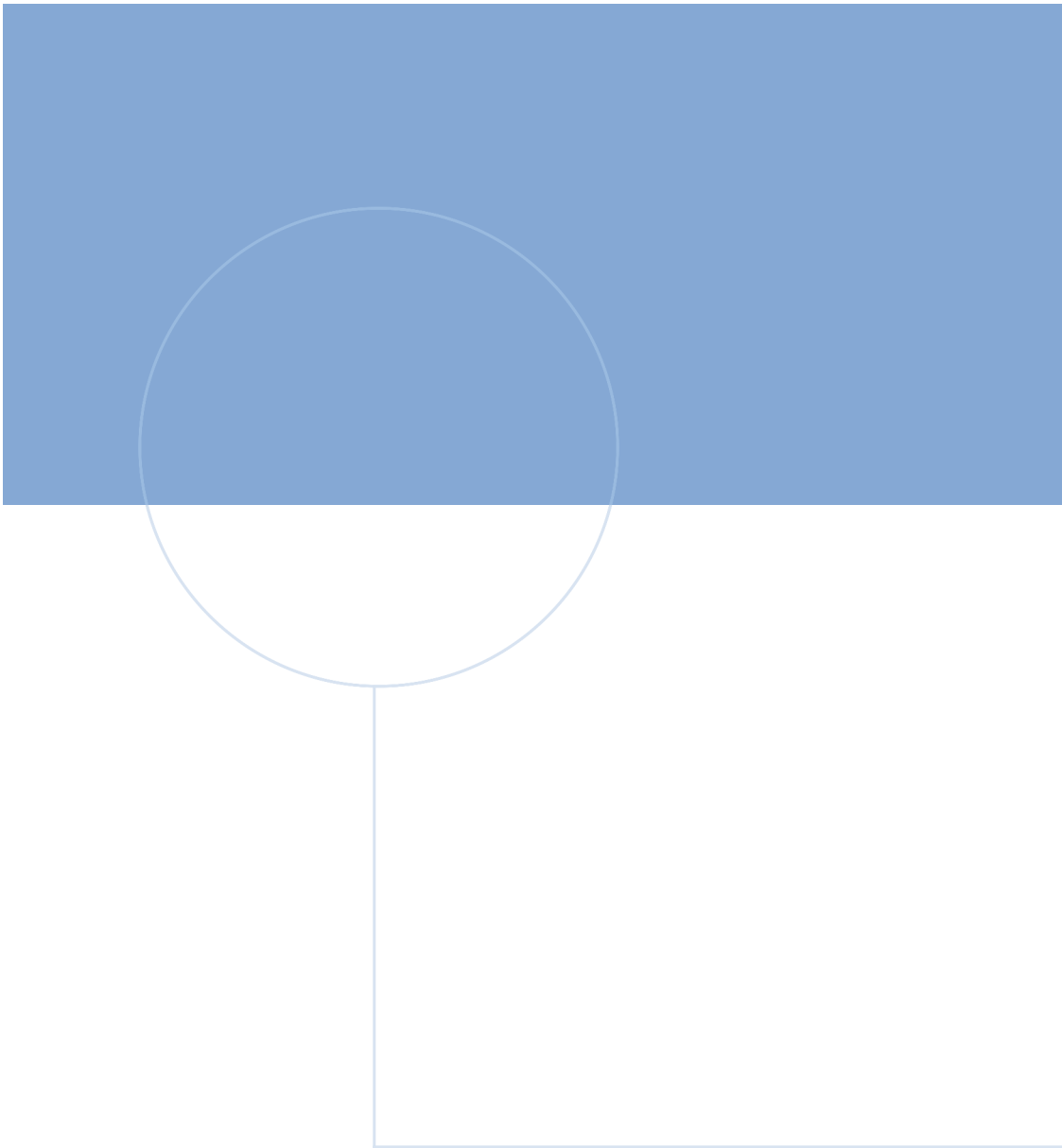
where δ represents a small deviation from the policy $\pi_\theta(s)$. From (A.6) - (A.6), the estimated performance gradient with the off-centred exploration scheme may be shown to be

$$\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] = \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta \nabla_\theta \pi_\theta^\top] \tag{C.2}$$

$$\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e e^\top \nabla_\theta \pi_\theta^\top] = \mathbb{E}_{s\sim\rho}[\nabla_\theta \pi_\theta (\delta\delta^\top + \Sigma\Sigma^\top) \nabla_\theta \pi_\theta^\top] \tag{C.3}$$

$$\begin{aligned}\mathbb{E}_{\substack{s\sim\rho \\ a\sim\beta}}[\nabla_\theta \pi_\theta e A_{\pi_\theta}(s_t, a_t)] = \nabla_\theta \pi_\theta \Big(& \Sigma\Sigma^\top \nabla_a A_{\pi_\theta}(s, \pi_\theta) \\ & + (\Sigma\Sigma^\top (W_{2,2} + W_{2,2}^\top) + A_{\pi_\theta}(s, \pi_\theta + \delta) + \mathrm{Tr}(\Sigma W_{2,2} \Sigma^\top))\delta \Big)\end{aligned} \tag{C.4}$$

Which by simple inspection, does not converge to the true performance gradient as long as $\delta \neq 0$.



NTNU

Norwegian University of
Science and Technology