Optimal and adaptive control frameworks using reinforcement learning for time-varying dynamical systems

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Ibrahim Eniola Sanusi

Department of Automatic Control and Systems Engineering

The University of Sheffield, UK

December 2019

To my family

Declaration

I, Ibrahim Eniola Sanusi, confirm that the work presented in this thesis is my own. All the material in the thesis which is not my own has been properly referenced and acknowledged.
Ibrahim Eniola Sanusi

Acknowledgements

I give all glory and thanks to Almighty Allah for His infinite mercies through my life and particularly through my academics.

My sincere thanks and gratitude go to Dr. Andrew R. Mills of the Rolls-Royce University Technology Center (UTC) for providing me with the opportunity to work on this doctoral research programme and for his continued support in scrutinising and guiding my work throughout the programme. I greatly appreciate your efforts and support.

I would also like to thank my supervisors, Professor Tony Dodd and Dr. George Konstantopoulos for their intellectual support, guidance and abundant feedback on my research works. I am grateful to Professor Visakan Kadirkamanathan, Director of the Rolls-Royce UTC for his continued academic and professional support.

I greatly acknowledge my senior colleagues Dr. Sayo Obajemu and Dr. Taofeeq Ibn-Mohammed for their push and encouragement both within and outside the academic environment. To all my colleagues, past and present at the Rolls-Royce UTC, all members of staff of Automatic Control and Systems Engineering, University of Sheffield, and all my PhD colleagues and friends, Thank you all!

I must at this point thank my family to which this research work is dedicated - my beloved parents, Mr. Najimdeen and Mrs. Monsurat Sanusi, my beloved brother, his wife and kids, Dr. Habeeb, Dr. Sherifat, Amaan and Aaliyah Sanusi, my beloved sister, her husband and kid, Mr. Jamiu, Mrs. Habibat and Khalil Adeniran, and my beloved partner, Dr. Moriam Oyebamiji. Thank you for your relentless prayers and support, I love you all.

Ibrahim Eniola Sanusi
December, 2019.

Abstract

Performance of complex propulsion and power systems are affected by a vast number of varying factors such as gradual system degradation, engine build differences and changing operating conditions. Owing to these variations, prior characterisation of the system performance metrics such as fuel efficiency function and constraints is infeasible. Existing model-based control approaches are therefore inherently conservative at the expense of the system performance as they are unable to fully characterise the system variations. The system performance characteristics affected by these variations are typically used for health monitoring and maintenance management, but the opportunities to complement the control design have received little attention. It is therefore increasingly important to use the information about the system performance characteristics in the control system design whilst considering the reliability of its implementation.

This thesis therefore considers the design of direct adaptive frameworks that exploit emerging diagnostic technologies and enable the direct use of complex performance metrics to deliver self-optimising control systems in the face of disturbances and system variations. These frameworks are termed condition-based control techniques and this thesis extends reinforcement learning (RL) theory which has achieved significant successes in the area of computing and artificial intelligence to the new frameworks and applications.

Consequently, an online RL framework was developed for the class of complex propulsion and power systems that make use of the performance metrics to directly learn and adapt the system control. The RL adaptations were further integrated into existing baseline controller structures whilst maintaining the safety and reliability of the underlying system. Furthermore, two online optimal RL tracking control frameworks were developed for time-varying dynamical systems that use a new augmented formulation with integral control. The proposed online RL frameworks advance the state-of-the-art for use in tracking control applications by not making restrictive assumptions on reference model dynamics or use of discounted tracking costs, and guaranteeing zero steady-state tracking error.

Finally, an online power management optimisation scheme for hybrid systems that uses a condition-based RL adaptation was developed. The proposed power management optimisation scheme is able to learn and compensate for the gradual system variations and learn online the optimal power management strategy be-

tween the hybrid power source given future load predictions. This way, improved system performance is delivered and providing a through-life adaptation strategy.

Contents

List of Figures

List of Tables

Acronyms

ACD adaptive critic design.

AD-DHP action-dependent dual heuristic dynamic programming.

AD-HDP action-dependent heuristic dynamic programming.

ADP approximate dynamic programming.

APR air pressure ratio.

ARE algebraic Riccati equation.

ARMA auto-regressive moving-average.

BLS batch least squares.

CBC condition-based control.

CT continuous-time.

DHP dual heuristic dynamic programming.

DOF degree-of-freedom.

DP dynamic programming.

DR demand response.

DT discrete-time.

EMS energy management system.

EPR engine pressure ratio.

ESC extremum seeking control.

ESS energy storage system.

GP gaussian process.

GTE gas turbine engine.

HDP heuristic dynamic programming.

HEV hybrid electric vehicle.

HJB Hamilton-Jabobi Bellman.

HTV hybrid tracked vehicle.

I-P integral-proportional.

IFT iterative feedback tuning.

ILC iterative learning control.

IRL integral reinforcement learning.

KF Kalman filter.

LQR linear quadratic regulator.

LQT linear quadratic tracker.

LTI linear time invariant.

MC Monte Carlo.

MDP Markov decision process.

MIMO multiple-input-multiple-output.

MPC model predictive control.

MRAC model reference adaptive control.

NCO necessary conditions for optimality.

NN neural networks.

OPFB output-feedback.

P.O. percentage overshoot.

PE persistence of excitation.

PI policy iteration.

PSC performance seeking control.

QFA Q-function approximation.

RL reinforcement learning.

RLS recursive least squares.

RTO real time optimisation.

SARSA state-action-reward-state-action.

SDP sequential decision problem.

SM surge margin.

SOC state of charge.

SPSA simultaneous perturbation stochastic approximation.

TD temporal difference.

TSFC thrust specific fuel consumption.

UAV unmanned aerial vehicle.

UC unfalsified control.

UUB uniformly ultimately bounded.

VBV variable bleed valve.

VFA value function approximation.

VGC variable geometry component.

VI value iteration.

VIGV variable inlet guide vane.

VRFT virtual reference feedback tuning.

VSV variable stator vane.

WFE water fuel emulsion.

Chapter 1

Introduction

The use of optimisation theory has become widespread in control owing to increasing needs to efficiently operate systems from economic, performance and safety perspectives. Particular performance considerations could be measures of utility such as the system operational costs, energy usage, fuel efficiency, system durability and life. It is often desirable to use suitable control techniques to optimise these complex performance metrics based on the system condition, whilst satisfying control specifications and system operational limits. In this thesis, such techniques are termed *condition-based* control techniques and encompass techniques which exploit emerging diagnostic technologies to deliver self-optimising control systems in the face of disturbances and system variations [3].

Well known control techniques such as the linear quadratic regulator (LQR) and standard model predictive control (MPC) are designed towards achieving set-point tracking and regulation of system disturbances, whose variations are either explicitly characterised or assumed to be within certain bounds [4], [5]. However, for some classes of systems, prior characterisation or determination of the system variations is infeasible. The inability to properly characterise these variations has led to the development of various alternate techniques over the years belonging to the class of adaptive control techniques and intelligent systems design [3], [6], [7], [8].

However, the conventional adaptive control techniques to mitigate against system variations are not usually designed to be optimal, in the sense of explicitly minimising a desired performance metric and are said to be indirect adaptive schemes [9]. Indirect adaptive schemes design an optimal controller against an identified system model that is assumed to characterise the desired performance metrics for all the possible system variations. In contrast, direct adaptive schemes explicitly adjust control actions to optimise a desired performance cost without the need to learn the system model or assume characterisation of the performance metrics. This thesis therefore considers the design of a class of direct adaptive controllers for time-varying dynamical systems, that is able to learn online, the optimal controller solutions to some desirable performance costs without the need to learn the system model and using only the measured system data. Furthermore, this thesis seeks extension of the class of direct optimal and adaptive controllers to complex propulsion and power systems such as the gas turbine engines, whose performances are affected by a vast number of varying factors [10]. These factors could include engine build differences, gradual engine degradation and changing operating conditions [11]. Optimising the system performance as a result of the varying factors pose a major challenge to the control of the complex systems, necessitating research into techniques that will achieve superior performance levels to conventional techniques. For the class of complex propulsion and power systems in consideration, the following characteristics are noted:

- *Increase in the system controllability.* Envisaged increase in the number of control variables for modern propulsion and power systems lead to increased prospect in achieving desired system optimality [3], [11]. However, this comes at an increased risk and cost in the design of the optimal controllers and modelling the interaction between the control variables, performance characteristics and the effects of system variations.

- *Performance variations arising from degradation and engine build differences.* The system behaviour, specifically the efficiency functions and constraints vary between different systems, and with different operating conditions. This means that the optimum values for the controllers cannot be designed in advance suggesting the need for techniques which enable controller adaptation that extract improved performance for the individual systems.

- *Mathematical models to characterise the system variations are difficult or infeasible to derive.* Lack of accurate analytical models to approximate all possible system variations means that techniques to directly compensate for their effects on performance cost such as minimum fuel consumption or minimum energy are unavailable.

- *Stringent performance and safety requirements.* Techniques by which to optimise the system performance must also ensure the integrity of the overall system throughout the operating envelope. The techniques must therefore provide practical strategies in implementation to guarantee the safety requirements for the systems.

1.1 Aims and objectives

The aim of this thesis is to develop (i.e. design, analyse and mature) condition-based control frameworks that optimise desired system performance by taking into account the effects of the system degradation and other system variations, whilst maintaining the system safety/reliability. Current candidate solutions for the condition-based control relies on the ability to accurately estimate the varying system states that affect the system performance, and therefore cannot fully compensate for all the possible system variations such as the gradual engine degradation and engine build differences. Objectives of this thesis are therefore to:

- Develop direct optimal and adaptive control algorithms for time-varying dynamical systems that do not require explicit mathematical models to characterise the varying system states due to degradation and other disturbances affecting the system performance.

- Develop direct optimal and adaptive control algorithms for time-varying dynamical systems that learn and optimise the desired system performance characteristics online such as fuel consumption, efficiency and life, using only the measured system data.

- Develop condition-based control frameworks that integrate the direct optimal and adaptive control algorithms into existing controller structures to learn and optimise the desired system performance characteristics whilst enabling the satisfaction of stringent safety requirements.

- Extend and show applications of the developed condition-based control frameworks to the class of propulsion and hybrid power systems.

1.2 Contributions

This thesis has explored the development of condition-based control algorithms and frameworks that enable the direct optimisation of desired performance characteristics for complex time-varying dynamical systems such as the class of propulsion and power systems that are subject to unknown variations and degradation. The main contributions of this thesis are listed as follows:

- The design of control architectures and algorithms that incorporate reinforcement learning approaches into existing controller structures for complex propulsion and power systems. The innovative architectures advance the state-of-the-art to allow direct optimisation of desired system performance measures whilst satisfying the system safety and stability constraints.
- The development of two new online optimal reinforcement learning tracking control frameworks for time-varying dynamical systems that guarantee zero steady-state tracking error and which unlike prior art do not make any restrictive assumptions on reference model dynamics or use of discounted tracking costs - the first framework uses state and input measurements, while the second uses only the input/output data for systems where full state measurements may be unavailable.
- The development of a new online power management optimisation scheme for hybrid systems that uses dynamic programming and an iterative Q-learning adaptation of the system performance function in a receding horizon manner to compensate for gradual system variations or uncontrolled system disturbances. The proposed power management optimisation scheme advances the state-of-the-art by compensating for gradual system variations, extracting improved system performance and iteratively learning online, the optimal power management strategy between the hybrid power sources given the future load predictions.

1.3 Publications

The proposed techniques in this thesis are based on the following author's publications:

- I. Sanusi, A. Mills, P. Trodden, V. Kadirkamanathan, and T. Dodd. Reinforcement learning for condition-based control of gas turbine engines. In *2019 18th European Control Conference (ECC)* pages 3928-3933. IEEE, 2019.
- I. Sanusi, A. Mills, T. Dodd, and G. Konstantopoulos. Online optimal and adaptive integral tracking control for varying discrete-time systems using reinforcement learning. *International Journal of Adaptive Control and Signal Processing*, 2020.
- I. Sanusi, A. Mills, and G. Konstantopoulos. Output-feedback tracking with integral control using reinforcement learning. 2020 (unpublished).

- I. Sanusi, A. Mills, G. Konstantopoulos, and T. Dodd. Power management optimisation for hybrid electric systems using reinforcement learning and adaptive dynamic programming. In *2019 American Control Conference (ACC)*, pages 2608-2613. IEEE, 2019.

1.4 Thesis Outline

In Chapter 2, a literature review for the various state-of-the-art optimal and adaptive control strategies that are used for system performance optimisation under uncertainties is provided and discusses the challenges and subsequent research efforts towards model-free adaptive approaches. The chapter then identifies reinforcement learning and approximate dynamic programming as a candidate model-free adaptive strategy that provides attractive features of iteratively learning optimising solutions to desired performance cost and the chapter concludes with a discussion on open research areas that are addressed in the rest of the thesis.

Chapter 3 provides the development of online reinforcement learning frameworks that are designed to be both adaptive and optimal for the control of time-varying dynamical systems. The reinforcement learning frameworks are first illustrated with a worked example for the linear quadratic regulation problem of discrete-time systems that converge to the optimum solutions subject to partially or completely unknown system dynamics. Subsequently, a condition-based framework for the reinforcement learning techniques is introduced here allowing three possible problem types to be solved - open-loop, closed-loop and supervisory control. This framework is compatible with legacy control architectures, aiding safety requirements to be met and was demonstrated on representative engine data sets.

Chapter 4 provides the development of online reinforcement learning frameworks for the optimal tracking control problem. The conventional model-based and reinforcement learning solutions to the online tracking problem for discrete-time systems are first provided. Limitations and restrictions of these existing solutions are thereafter highlighted and discussed. Consequently, an augmented formulation with integral control for the online tracking problem is proposed and extended to reinforcement learning frameworks that solve the limitations of the existing methods. The chapter concludes with a simulation of the proposed techniques on two representative case studies.

Chapter 5 extends the augmented formulation with integral control proposed in Chapter 4 and provides the development of online output-feedback reinforcement learning frameworks for the optimal tracking control problem. The output-feedback formulation uses only the measured input/output data for the optimal tracking control in systems where full state measurements are unavailable or the design of state estimators is difficult. A simulation example at the end of the chapter demonstrates the effectiveness of proposed online output-feedback optimal tracking control framework.

Chapter 6 provides the development of an online power management scheme for hybrid systems that compensates for gradual system variations or uncontrolled system disturbances given future load predictions. An overview of the current power management optimisation strategies are first discussed and a conventional dynamic programming solution to the power management problem is provided. Limitations of the conventional dynamic programming solution is discussed and consequently, a reinforcement learning and approximate dynamic programming strategy is proposed that overcomes the limitations of the existing strategies. The chapter concludes with a simulation case study of the proposed power management strategy on a representative autonomous hybrid system and shows improved system system as compared with the conventional dynamic programming solution.

Lastly, Chapter 7 provides concluding remarks on the proposed strategies and provides recommendations for future research directions. Figure 1.1 shows the outline of the thesis.

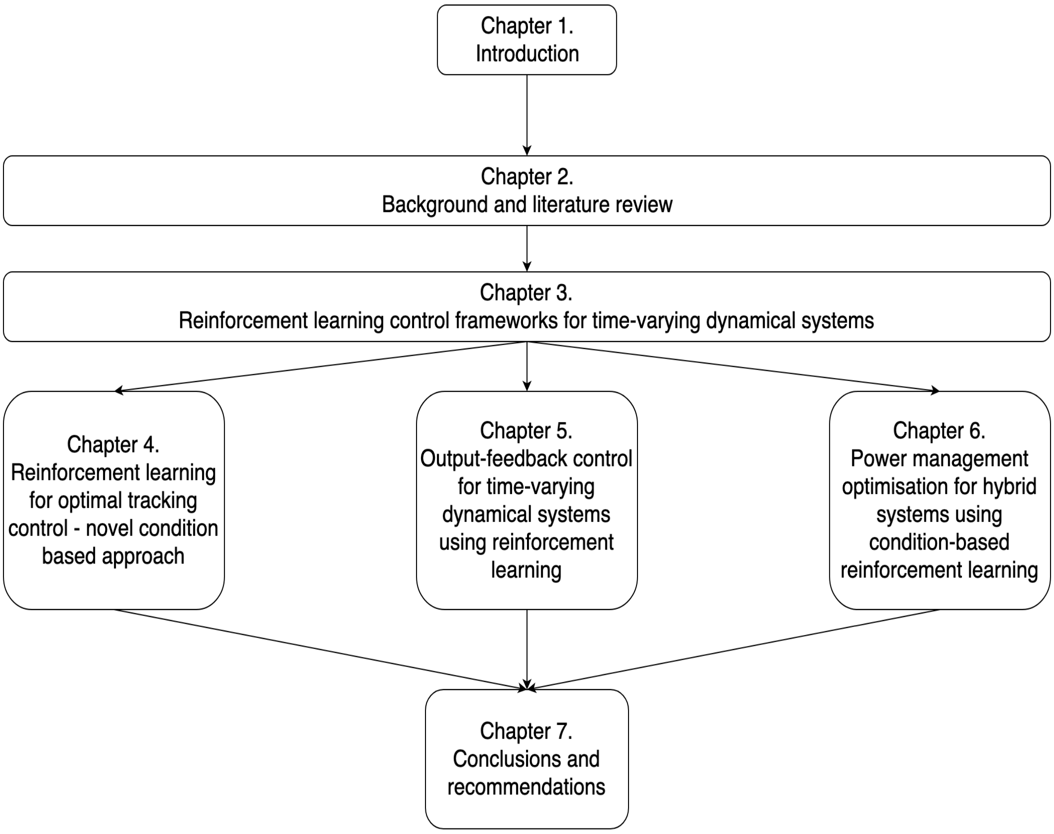

Chapter 1.
Introduction
Chapter 2.
Background and literature review
Chapter 3.
Reinforcement learning control frameworks for time-varying dynamical systems
Chapter 4.
Reinforcement learning for optimal tracking control - novel condition based approach
Chapter 5.
Output-feedback control for time-varying dynamical systems using reinforcement learning
Chapter 6.
Power management optimisation for hybrid systems using condition-based reinforcement learning
Chapter 7.
Conclusions and recommendations

Figure 1.1: Outline of the thesis.

Chapter 2

Background and literature review

This chapter provides a literature review for the various state-of-the-art optimal and adaptive control strategies that are used for system performance optimisation under uncertainties. The majority of these techniques are model-based, requiring considerable effort in the generation of high fidelity models that capture the varying systems and operating conditions. For complex systems, the model-based approaches become increasingly limited in their ability to compensate for system variations. An overview of the advantages and disadvantages of these techniques along with the subsequent research efforts towards data-based adaptive strategies and reinforcement learning (RL) are presented. This is followed by a study of the central topics and technical prerequisites of RL discussed throughout this thesis. A section is devoted to the review of current research trends and application of RL from historically significant works and the chapter concludes with a discussion on open research issues that are addressed in this thesis.

2.1 Optimal and adaptive control strategies

The design of optimal controllers is made possible by using complete system information and assuming bounds on possible disturbances. An example is the well known linear quadratic regulator (LQR) which is designed offline by solving the Hamilton-Jabobi Bellman (HJB) equations using full knowledge of the system dynamics [4]. Adaptive controllers on the other hand, are designed to use system measurements to learn and modify the behaviour of the controller in response to changes in the system dynamics and operating conditions. The ways and manners in which the controller learns from measurements and modifies its behaviour define different adaptive algorithms some of which are discussed in this chapter.

Despite the obvious benefits of using adaptive algorithms to compensate for unmodelled system dynamics and disturbances, their widespread applications have been limited in practice. For example, it was reported in [12] and [13] that conventional adaptive schemes such as the MIT rule are affected by inherent stability issues resulting from phenomenon such as bursting and non-separation of underlying adaptation time scales, leading to poor performance in applications. As researchers began to understand these problems better, newer adaptive algorithms have focused on ensuring persistence of excitation and separation of time scales associated with adaptation and dynamics of the systems. Evidence of applications of these newer adaptive schemes have since been reported in the literature [14], [15], [16], [17], [18] motivated by the need to operate systems at some varying optimum set-points that yield desired performance improvements.

This thesis focuses on adaptive algorithms that enable the possibility of realising minimising solutions to user prescribed performance characteristics (i.e. both adaptive and optimal) whilst ensuring stability and convergence of the underlying adaptation scheme for practical implementations. An overview of these adaptive strategies of interest is now presented and broadly classified into model-based and model-free strategies.

2.1.1 Model-based strategies

Model-based adaptive control techniques are conventionally classified as either indirect or direct adaptive schemes [8]. Indirect schemes make use of the system measurements to learn new system models via system identification techniques in closed loop [19]. The identified models are then used to adapt the system control law or modify its sensitivity. In contrast, direct model-based adaptive schemes make no efforts to identify new system models but instead use the system measurements to directly adapt parameterised system of controllers in the feedforward or feedback path [20]. A popular direct adaptive technique is the model reference adaptive control (MRAC) that makes use of a reference model with desired performance characteristics running in tandem with the actual system. An adjustment mechanism compares the output of the reference model with that of the actual system and uses the generated error statistics to adapt the system controller.

Figure 2.1 shows the schematics of the two model-based adaptive control schemes. Depending on the use of model, the model-based adaptive schemes can be further

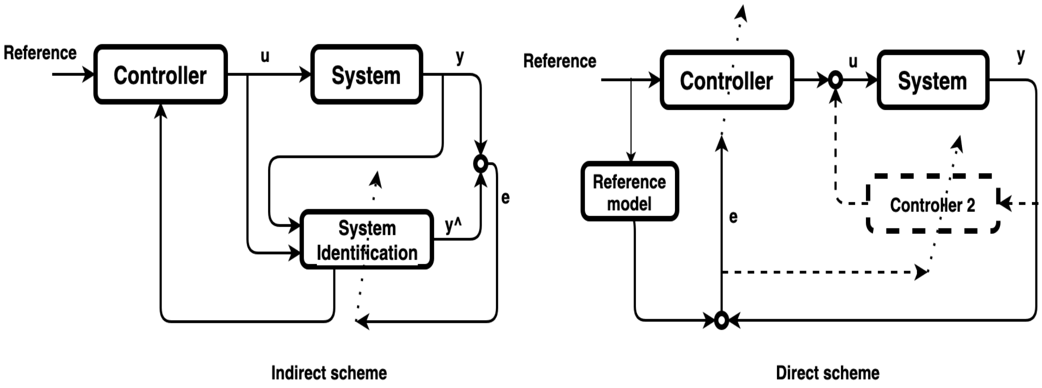


Figure 2.1: Schematic of model-based direct and indirect adaptive control schemes. Indirect adaptive control schemes adapt a model of the system or desired characteristics by using the error statistics generated from the actual system output and the model. In contrast, direct adaptive schemes use the error statistics to directly adapt a parameterised system of controllers.

classified into those that either use offline (fully model-based) or online models [16]. Classical examples of those that use offline models include the gain scheduling, multiple model adaptive and self-optimising control schemes [7], [21], [22]. Other variants of the model-based adaptive schemes involve the use of online models - of note are the performance seeking control (PSC) and real time optimisation (RTO) control schemes that have been widely reported respectively in the aerospace and process industries [23], [24], [25], [26]. In all of these schemes, a considerable effort is needed to build a high fidelity model to be used in the adaptations.

For example, the PSC schemes in aerospace applications make use of high fidelity on-board models which typically consist of a linear steady-state perturbation model of the engine and empirically derived steady-state trim tables with follow-on nonlinear engine calculations. By using flight measurements from the actual system and a Kalman filter (KF) framework, the on-board models are adapted and matched to the actual system conditions as shown in Figure 2.2 and are able to compensate for engine build differences, deterioration and changing operating conditions [23], [24], [27], [28]. But the significant benefits of the PSC schemes come at a cost, as reported by Gilyard and Orme [23] that the technology was only made possible by using models from 15+ years of experience with the F100 class of engines and accurate nonlinear simulation of the engines. Similarly, a recent work by General Electric (GE) aviation [28] which makes use of a tracking filter to estimate engine deviation parameters (EDPs) to account for engine deterioration and variations has also relied on the use of expensive high fidelity engine models.

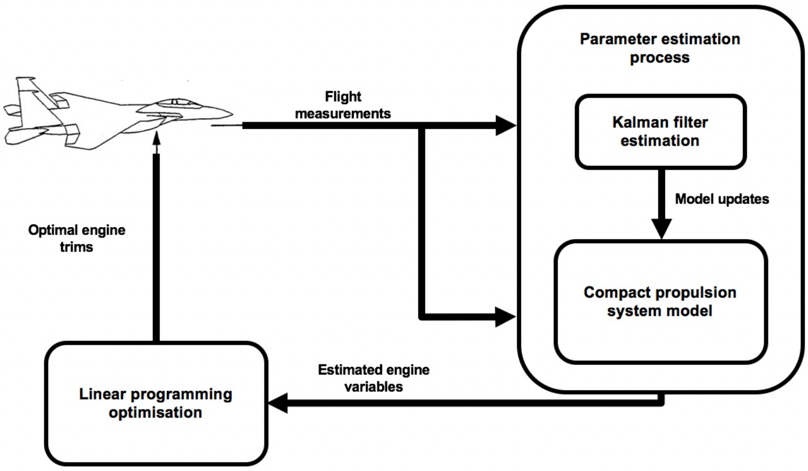

Parameter estimation process
Kalman filter estimation
Model updates
Compact propulsion system model
Flight measurements
Optimal engine trims
Estimated engine variables
Linear programming optimisation

Figure 2.2: Schematic of the performance seeking control scheme. The scheme uses a high fidelity on-board model which is matched to actual engine condition using a Kalman filter framework.

Whilst the model-based adaptive schemes are considered matured judging by their long history of applications, their performance is limited to the known dynamics of the specific models used. This can be restrictive as the mathematical models to fully approximate all the possible variations affecting the system performance is infeasible. Perhaps, alternatives to these schemes are those that do not rely on explicit mathematical models of the system, but systematically adapt and control the system using obtained measurements. These alternative schemes fall under the model-free strategies and are discussed next.

2.1.2 Model-free strategies

Model-free adaptive schemes do not rely on any explicit mathematical model or knowledge of the system. In principle, they are flexible in dealing with any uncertainties or variations by using the system measurements to directly synthesize or adapt the controllers. Also known as data-driven or data-based control, these schemes are typically not affected by many of the limitations of their model-based counterparts such as:

- the need for an expensive model or knowledge of the system.
- system identification problems in simultaneously achieving steady-state control and identifying the dynamics of the system.

- complicated manual tuning and stability issues associated with fixed model-based controller designs [13], [16].

Furthermore, the model-free strategies provide the possibility of developing truly optimal controllers that are equally adaptive by optimising to desired performance cost [9], [29]. Popular model-free strategies of interest include the unfalsified control (UC), simultaneous perturbation stochastic approximation (SPSA), iterative feedback tuning (IFT), virtual reference feedback tuning (VRFT), iterative learning control (ILC), extremum seeking control (ESC) and the class of RL and approximate dynamic programming (ADP) [18].

In UC control, sets of candidate controllers are iteratively falsified (discounted) against desired performance objectives using input/output data, and then selecting the controller with the best inferred performance [30]. Performance of the approach is however limited to the set of initial candidate controllers, thus providing little flexibility in accounting for unmodelled variations and achieving the desired optimality. SPSA control on the other hand uses a tunable function approximator with a fixed structure as its controller, whose parameters are adapted by minimising the desired performance objectives using the input/output data [31]. Two simultaneous recursions are performed for the parameter tuning to estimate both the cost gradient information and the subsequent controller parameters. For any theoretical guarantees on the approach, specific gradient forms are assumed for the choice of the fixed controller structures, equally limiting the flexibility of the approach to unmodelled system variations.

Other model-free approaches like the IFT iteratively adapt the parameters of their controller by using gradient information obtained from offline closed-loop experiments. Under mild assumptions, the IFT approach has been shown to converge to a local minimum of the objective function and has been successfully applied in many industrial applications [32]. The use of offline gradient information however limits the possibility of adapting the system controllers to match the actual system conditions and compensate for system variations. Consequently, achieving desired performance characteristics in a noisy setting has been proposed using the VRFT method through the use of a desired reference model. Given output data from the desired reference model, VRFT introduces a virtual reference signal to generate error statistics and transforms the control design into a controller parameter identification problem using the input/output measurements [33]. Similar performance limitations of its equivalent model-based MRAC is also present as a considerable effort is required in obtaining the desired reference model and variation characteristics.

A popular model-free adaptive scheme that circumvents the need for either a fixed controller structure or the use of reference models is the class of ESC schemes [34], [35]. These schemes do not use any identification mechanism or closed form knowledge of the system or reference dynamics but proceed by optimising a desired cost function from the system measurements and gradually adapting the control variables to optimal operating points. This is however achieved at the expense of perturbing the system by adding excitation signals to obtain information about the system and driving the search through gradient descent techniques. Furthermore, the basic model-free ESC scheme assumes the use of a smooth cost function with a unique optimum thereby limiting its possible extension to complex systems involving non-smooth cost function with input and state constraints [16].

ILC introduces a different approach to the adaptive control problem by providing an adaptation mechanism that achieves better performance for systems with repetitive tasks in finite time using input/output data with a memory. Based on contraction mapping theory, ILC scheme converges as the number of iterations approach infinity by using a simple fixed controller structure to minimise a learning error between a target trajectory and the actual system output [14]. The target trajectory is however assumed known and identical for all the iterations limiting its extensions to other non-repetitive adaptive control tasks. A similar approach that is based on contraction mapping but not limited to repetitive tasks is the class of RL and ADP. RL is classed as a model-free adaptive scheme that is also optimal by directly optimising user-prescribed performance characteristics and has achieved significant successes in the area of computing and artificial intelligence in solving complex optimisation problems [36]. With roots in psychology and recently in machine learning, RL incrementally improves desired control behaviour by simply interacting with the system and learning how to map states to actions using reward signals (positive or negative reinforcements) from the system [37].

RL schemes provide attractive features of learning 'optimality over time' using only the observed system measurements and are able to overcome the limitations of the other adaptive schemes. For example, the assumption that a model of the desired performance characteristics is known (either offline or online) in the popular adaptive schemes such as the MRAC, IFT, VRFT and ILC, limits the flexibility of the approaches in achieving optimality as they are constrained to the modelled desired characteristics. RL introduces a new design approach that is purely based on interaction with the actual system subject to the unknown system dynamics or variations. This extra flexibility in the design approach makes RL an attractive candidate for optimal and adaptive control and has been exploited in many complex applications spanning different fields [29], [37], [38].

RL however has its weakness as it fundamentally simplifies the control of complex systems by assuming a Markovian model i.e. the current state(s) is/are dependent only on the previous state(s) (more on this in the next section). Without a known model of the system, a direct consequence of the Markovian model is the dependence on a lot of data gathering to learn the optimising control policy. Consequently, this may lead to the use of complex approximation techniques whose robustness and stability guarantees may be intractable or hard to prove. This thesis focuses on extending the RL theory to new frameworks that provide a tractable design and control for complex dynamical systems such as in varying propulsion and power systems. The rest of this chapter discusses the key elements and central theories of RL that are used throughout the thesis.

2.2 Reinforcement learning and adaptive dynamic programming

General learning systems can be categorised based on the available learning feedback as *supervised*, *unsupervised* and *reinforcement learning* schemes. Supervised learning describes a framework that learns the mapping between available input and output data, while unsupervised learning learns hidden patterns in the output data without any input information. RL on the other hand describes a framework in which training information is initially unavailable but learns by interacting with the system and using the received data or measurements to enhance future control of the system [39]. This represents a common scenario in practical systems making RL an active area of research with applications from different fields including computer science and artificial intelligence, operations research, robotics and control.

In control, mathematical implementations of RL have been enabled through approximate/adaptive dynamic programming (ADP) which provides a framework to optimise desired performance cost and to learn optimal control policies using only measured data along the system trajectory [9]. RL and ADP are deeply rooted and based on the principles of dynamic programming. The next section provides the relationship of dynamic programming to optimal control which is fundamentally a backward-in-time problem, and later extended to RL-ADP adaptive control frameworks.

2.2.1 Dynamic Programming Optimisation

Introduced by Bellman [40], dynamic programming (DP) provides a systematic way of solving sequential decision problems (SDPs) in an optimum manner. By SDP, we refer to problems that involve a sequence of decision makings and observations, and occur in several fields including operations research and in control engineering. DP method is mainly recursive and has been applied to a variety of problems involving both continuous or discrete states and actions [41]. The problem addressed by DP is mostly studied under the Markov decision process (MDP) framework which is a tuple consisting of:

$$MDP := (\mathbb{X}, \mathbb{U}, \mathcal{P}_{SA}, \gamma, \mathcal{R}) \tag{2.1}$$

where $\mathbb{X}$ is the set of states, $\mathbb{U}$ is the policy or the set of actions, $\mathcal{P}_{SA}$ is the state transition probability (for stochastic system) or state dynamics (for deterministic systems), $\gamma \in [0, 1]$ is a discount factor, and $\mathcal{R} : x \times u \to \mathbb{R}$ is a reward function for taking action $u \in \mathbb{U}$ in state $x \in \mathbb{X}$. The MDP sequence proceeds as follows:

$$x_0 \underbrace{\rightarrow}_{u_0} x_1 \in \mathcal{P}_{x_1|x_0,u_0} \underbrace{\rightarrow}_{u_1} x_2 \in \mathcal{P}_{x_2|x_1,u_1} \underbrace{\rightarrow}_{u_2} \cdots \underbrace{\rightarrow}_{u_{N-1}} x_N \in \mathcal{P}_{x_N|x_{N-1},u_{N-1}} \tag{2.2}$$

The goal in DP is then to optimise some desired cost function that is additive over time as a result of visiting states x_0 to x_N and taking actions u_0 to u_{N-1} given as:

$$J(x_k) = \mathcal{R}(x_N) + \sum_{n=k}^{N-1} \gamma^{n-k} \mathcal{R}(x_n, u_n) \tag{2.3}$$

where the optimisation is over the control actions u_0 to u_{N-1} and $\mathcal{R}(\boldsymbol{x}_N)$ is the terminal cost. DP provides a solution to this optimisation problem by making use of the principle of optimality which states that "an optimal policy has the property that no matter what the previous decisions or actions have been, the remaining decisions must constitute an optimal policy with respect to the state resulting from those previous decisions" [40]. The principle of optimality therefore suggests that the optimal control sequence can be broken down into sub-stages, by first determining the optimal control decision for the last stage called the tail-subproblem, and then proceeding to the other sub-stages till the whole problem is solved [42]. Using the Bellman principle of optimality, a recursive formulation for the optimum value of the cost defined in (2.3) is given by the Bellman optimality equation as:

$$\begin{aligned}V^*(x_k) &= \min_{u_k}\Big\{\sum_{n=k}^{N}\gamma^{n-k}\mathcal{R}(x_n,u_n)\Big\}\\ &= \min_{u_k}\Big\{\mathcal{R}(x_k,u_k)+\gamma\sum_{n=k+1}^{N}\gamma^{n-(k+1)}\mathcal{R}(x_n,u_n)\Big\}\\ &= \min_{u_k}\Big\{\mathcal{R}(x_k,u_k)+\gamma V^*(x_{k+1})\Big\}\end{aligned} \tag{2.4}$$

DP therefore recursively determines the optimal cost for the problem starting from a terminal cost $V(x_N)=\mathcal{R}(\boldsymbol{x}_{\mathcal{N}})$ from which the optimal control sequence can be determined as follows:

Solve backwards from $N-1:-1:k$

$$V^*(x_k)\leftarrow\min_{u_k}\Big\{\mathcal{R}(x_k,u_k)+\gamma V^*(x_{k+1})\Big\} \tag{2.5}$$

Some key properties of the dynamic programming approach that will be further exploited in the later chapters are presented next.

Properties of dynamic programming: The introduced DP algorithm can be represented in a shorthand form introduced by Bertsekas [43] if the DP mapping for any cost function $J(\cdot)\in\mathbb{R}$ is considered as:

$$V(x_k)=(TJ)(x_k)=\min_{u_k}\Big\{\mathcal{R}(x_k,u_k)+\gamma V(x_{k+1})\Big\} \tag{2.6}$$

where $(TJ)(\cdot)\in\mathbb{R}$ is the DP cost function for the one-stage problem with instantaneous cost $\mathcal{R}(\cdot)$ and terminal cost $\gamma V(\cdot)$. Similarly, a second DP mapping for any cost function $J(\cdot)\in\mathbb{R}$ and any stationary policy $\mu(\cdot)\in\mathbb{U}$ is given as:

$$V_\mu(x_k)=(T_\mu J)(x_k)=\min_{u_k}\Big\{\mathcal{R}(x_k,u_k)+\gamma V_\mu(x_{k+1})\Big\} \tag{2.7}$$

where $(T_\mu J)(\cdot)\in\mathbb{R}$ is the DP cost function given $\mu(\cdot)$ for the one-stage problem with instantaneous cost $\mathcal{R}(\cdot)$ and terminal cost $\gamma V_\mu(\cdot)$. For the N stage optimisation problem, the mappings (2.6) and (2.7) become:

$$(T^N J)(x_k)=\big(T(T^{N-1}J)\big)(x_k) \tag{2.8}$$

$$(T_\mu^N J)(x_k)=\big(T_\mu(T_\mu^{N-1}J)\big)(x_k) \tag{2.9}$$

Using the shorthand form, the following are the properties of the DP approach with their respective proofs shown in [43]:

1 Monotonicity property:

$$(TJ)(x_k) \leq (TJ^{'})(x_k) \quad \forall\, x \in \mathbb{X},\, k = 1, 2, \cdots, \tag{2.10}$$

$$(T_\mu J)(x_k) \leq (T_\mu J^{'})(x_k) \quad \forall\, x \in \mathbb{X},\, k = 1, 2, \cdots, \tag{2.11}$$

for any $J(\cdot)$ and $J^{'}(\cdot)$ such that $J(x_k) \leq J^{'}(x_k)$.

2 Constant shift property:

$$\big(T(J + re)\big)(x_k) = (TJ)(x_k) + \gamma r \quad \forall\, x \in \mathbb{X},\, k = 1, 2, \cdots, \tag{2.12}$$

$$\big(T_\mu(J + re)\big)(x_k) = (T_\mu J)(x_k) + \gamma r \quad \forall\, x \in \mathbb{X},\, k = 1, 2, \cdots, \tag{2.13}$$

for any scalar r and where $[e(x) \;\equiv 1]$ is a unit function.

3 Contraction mapping property: For the case where $J(\cdot)$ and $J^{'}(\cdot)$ are bounded functions, then for any stationary policy $\mu(\cdot)$, we have:

$$\max_x |(TJ)(x_k) - (TJ^{'})(x_k)| \leq \gamma \max_x |J(x_k) - J^{'}(x_k)| \quad \forall\, x \in \mathbb{X},\, k = 1, 2, \cdots, \tag{2.14}$$

$$\max_x |(T_\mu J)(x_k) - (T_\mu J^{'})(x_k)| \leq \gamma \max_x |J(x_k) - J^{'}(x_k)| \quad \forall\, x \in \mathbb{X},\, k = 1, 2, \cdots, \tag{2.15}$$

Given the properties of the DP approach, the policy $\mu(\cdot)$ is therefore said to be optimal if and only if the minimum is achieved using the Bellman optimality equation for each $x \in \mathbb{X}$ such that:

$$(TJ^*)(x_k) = (T_\mu J^*)(x_k) \quad k = 1, 2, \cdots, \tag{2.16}$$

It should be noted that DP is an offline method for determining optimal control sequence backwards-in-time, and serves to limit the optimisation search to only the optimal trajectories. Two representative optimisation problems are presented to further illustrate the DP optimisation approach.

2.2.1.1 Shortest path problem

To illustrate the DP concept, consider the problem of finding the shortest path when travelling from node 'a' to node 'i' as shown in Figure 2.3. The states are given as the nodes $\mathbb{X} = \{a, b, c, d, e, f, g, h, i\}$ while the control inputs are given as $\mathbb{U} = \{up(+1), down(-1)\}$. The instantaneous rewards incurred from taking action $u \in \mathbb{U}$ in state $x \in \mathbb{X}$ are given as the numbers on the arrow head links joining each node as shown in the diagram.

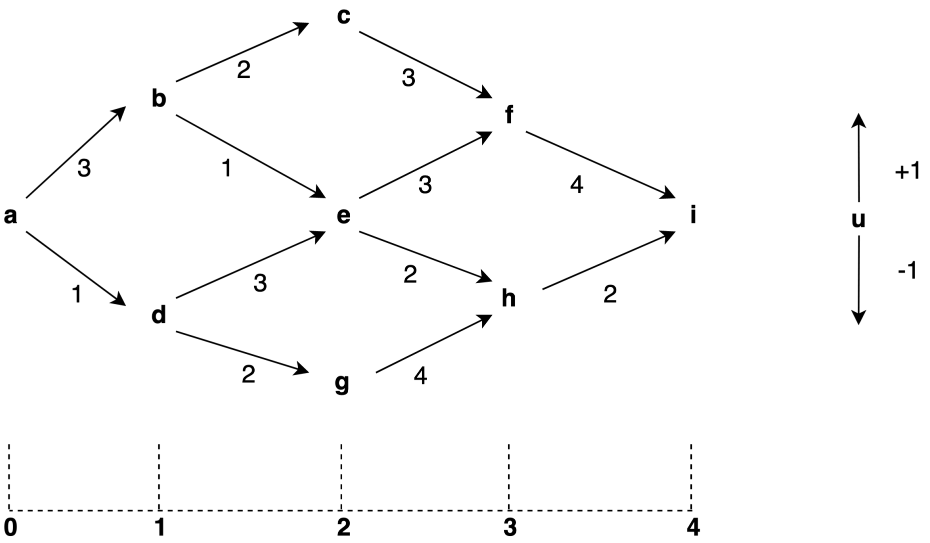


Figure 2.3: Schematic of a shortest path problem in traveling from node 'a' to 'i' given costs associated with each arrow head links joining the nodes.

For the undiscounted case (i.e $\gamma = 1$), a naive approach of solving the optimisation problem using an exhaustive search will easily lead to lots of evaluations and paths to optimise. In the given example with 9 states (nodes) and 4 stages of optimisation will result in $9^{4-1} = 729$ possible paths and a further $729 \times 2 = 1458$ evaluations for the 2 possible control actions. Using the backwards-in-time recursion of (2.5), DP solves the problem by starting from the terminal state (node 'i') and proceeding backwards as follows:

Assume terminal cost $V(i)^* = 0$

Stage 3:

$$\begin{aligned}
V^*(f) &= \min_u \Big\{4 + V^*(i)\Big\} = \min_u \Big\{4 + 0\Big\} = 4\\
V^*(h) &= \min_u \Big\{2 + V^*(i)\Big\} = \min_u \Big\{2 + 0\Big\} = 2
\end{aligned}$$

Stage 2:

$$\begin{aligned}
V^*(c) &= \min_u \Big\{3 + V^*(f)\Big\} = \min_u \Big\{3 + 4\Big\} = 7\\
V^*(e) &= \min_u \Big\{3 + V^*(f), 2 + V^*(h)\Big\} = \min_u \Big\{3 + 4, 2 + 2\Big\} = 4\\
V^*(g) &= \min_u \Big\{4 + V^*(h)\Big\} = \min_u \Big\{4 + 2\Big\} = 6
\end{aligned}$$

Stage 1:

$$\begin{aligned}
V^*(b) &= \min_u \Big\{2 + V^*(c), 1 + V^*(e)\Big\} = \min_u \Big\{2 + 7, 1 + 4\Big\} = 5\\
V^*(d) &= \min_u \Big\{3 + V^*(e), 2 + V^*(g)\Big\} = \min_u \Big\{3 + 4, 2 + 6\Big\} = 7
\end{aligned}$$

Stage 0:

$$V^*(a) = \min_u \Big\{3 + V^*(b), 1 + V^*(d)\Big\} = \min_u \Big\{3 + 5, 1 + 7\Big\} = 8 \tag{2.17}$$

DP thus limits the optimisation search to only the optimal trajectories and gives the optimal cost i.e the minimum cost of traveling from node 'a' to 'i' as $V^*(a) = 8$ and achieved in only 8 evaluations. The optimal paths and values identified from the DP recursion for starting at any node are shown in Figure 2.4.

2.2.1.2 Optimal control regulation problem

The DP solution to the example above provides a simple optimisation routine over the control inputs as both the cost and the state dynamics are given as discrete sequences. In general, the control optimisation approach is dependent on the nature of both the cost defined in (2.3) and the state dynamics function. For the optimal control problem, consider an affine-in-input discrete state dynamics given as:

$$\boldsymbol{x}_{k+1} = \mathcal{P}_{x_{k+1}|x_k,u_k} = f(\boldsymbol{x}_k) + g(\boldsymbol{x}_k)\boldsymbol{u}_k \tag{2.18}$$

where $f(\boldsymbol{x}) \in \mathbb{R}^n$ and $g(\boldsymbol{x}) \in \mathbb{R}^{n\times m}$ are respectively the drift and input system dynamics, $\boldsymbol{x} \subset \mathbb{X} \in \mathbb{R}^n$ are the system states and $\boldsymbol{u} = \mu(\boldsymbol{x}) \subset \mathbb{U} \in \mathbb{R}^m$ is the control input according to some policy $\mu(\cdot) : \mathbb{X} \to \mathbb{U}$. The optimal control problem is to

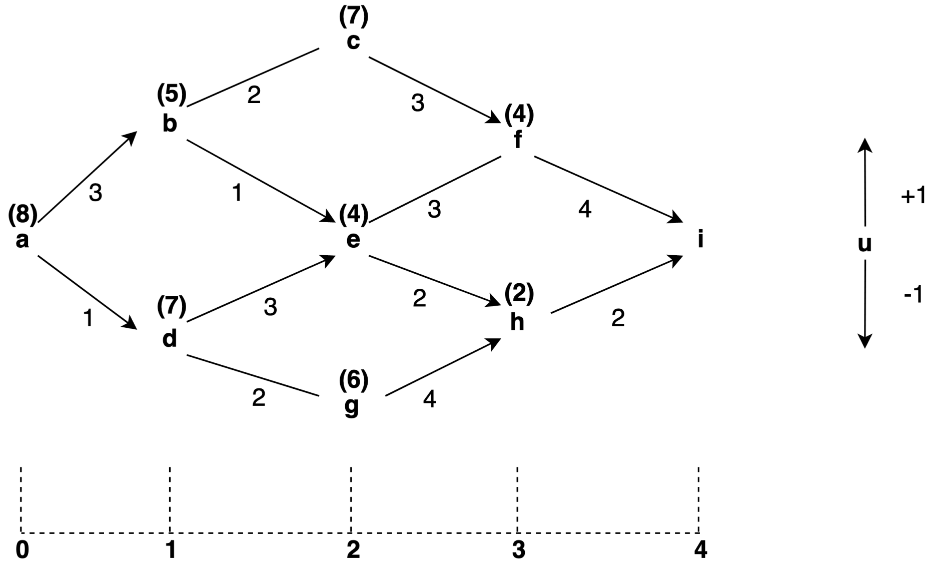


Figure 2.4: Schematic of the optimal paths and costs identified from the dynamic programming recursion for the shortest path problem.

find $\boldsymbol{u}_k^*, \forall k \in [n, N]$ that stabilises the closed loop system asymptotically in some set $\Omega \subset \mathbb{X}$ such that the cost (2.3) is minimised.

A baseline solution to the control optimisation problem is first provided using calculus of variation by defining a Hamiltonian function as:

$$H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1}) = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \boldsymbol{x}_{k+1} \tag{2.19}$$

where $\boldsymbol{\lambda}_{k+1} \in \mathbb{R}^n$ is a yet to be determined Lagrange multiplier. The Hamiltonian (2.19) forms a composite cost which includes the state dynamics. Corresponding necessary conditions for optimality (NCO) are given as:

1 State equation:

$$\boldsymbol{x}_{k+1} = \frac{\partial H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1})}{\partial \boldsymbol{\lambda}_{k+1}} = f(\boldsymbol{x}_k) + g(\boldsymbol{x}_k)\boldsymbol{u}_k \tag{2.20}$$

2 Co-state equation:

$$\boldsymbol{\lambda}_k = \frac{\partial H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1})}{\partial \boldsymbol{x}_k} = \frac{\partial \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k)}{\partial \boldsymbol{x}_k} + \left(\frac{\partial \boldsymbol{x}_{k+1}}{\partial \boldsymbol{x}_k}\right)^\top \boldsymbol{\lambda}_{k+1} \tag{2.21}$$

3 Stationary condition:

$$0 = \frac{\partial H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1})}{\partial \boldsymbol{u}_k} = \frac{\partial \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k)}{\partial \boldsymbol{u}_k} + \left(\frac{\partial \boldsymbol{x}_{k+1}}{\partial \boldsymbol{u}_k}\right)^\top \boldsymbol{\lambda}_{k+1}$$
$$= \frac{\partial \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k)}{\partial \boldsymbol{u}_k} + \big(g(\boldsymbol{x}_k)\big)^\top \boldsymbol{\lambda}_{k+1} \tag{2.22}$$

4 Boundary conditions $i < k < N$:

$$\left(\frac{\partial \mathcal{R}(\boldsymbol{x}_i, \boldsymbol{u}_i)}{\partial \boldsymbol{x}_i} + \left(\frac{\partial \boldsymbol{x}_{i+1}}{\partial \boldsymbol{x}_i}\right)^\top \boldsymbol{\lambda}_{i+1}\right)^\top d\boldsymbol{x}_i = 0;$$
$$\boldsymbol{\lambda}_N = \frac{\partial \mathcal{R}(\boldsymbol{x}_N)}{\partial \boldsymbol{x}_N} \tag{2.23}$$

In the following, consider the special case where (2.18) is modelled by the linear time invariant (LTI) system given as:

$$\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k \tag{2.24}$$

The pair (A, B) is assumed controllable and the reward is given as the standard quadratic energy function:

$$\mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{x}_k^\top \mathcal{Q}\boldsymbol{x}_k + \boldsymbol{u}_k^\top R\boldsymbol{u}_k \tag{2.25}$$

with $\mathcal{R}(\boldsymbol{x}_N) = \boldsymbol{x}_N^\top E\boldsymbol{x}_N$ and where $\mathcal{Q} \in \mathbb{R}^{n\times n}$, $E \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are symmetric positive semidefinite matrices. The NCO given by (2.20 to 2.23) becomes:

$$\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k \tag{2.26}$$
$$\boldsymbol{\lambda}_k = \mathcal{Q}\boldsymbol{x}_k + A^\top \boldsymbol{\lambda}_{k+1} \tag{2.27}$$
$$0 = R\boldsymbol{u}_k + B^\top \boldsymbol{\lambda}_{k+1} \tag{2.28}$$
$$\boldsymbol{x}_0; \quad \boldsymbol{\lambda}_N = E\boldsymbol{x}_N \tag{2.29}$$

From (2.28), the optimal control input is computed as:

$$\boldsymbol{u}_k^* = -R^{-1}B^\top \boldsymbol{\lambda}_{k+1} \tag{2.30}$$

Substituting for (2.30) in (2.26) and coupling the state and the costate equations yields the discrete Hamiltonian system given as:

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{\lambda}_k \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B^\top \\ \mathcal{Q} & A^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} \tag{2.31}$$

The Hamiltonian system (2.31) satisfies $JHJ = H^\top$ with $H \in \mathbb{R}^{2n\times 2n}$ and $J = \begin{bmatrix} 0 & -\mathcal{I}_n \\ \mathcal{I}_n & 0 \end{bmatrix}$. Using the boundary condition (2.29) on the co-state, the following linear relation can be assumed:

$$\boldsymbol{\lambda}_k = P_k \boldsymbol{x}_k \quad \forall k \leq N \tag{2.32}$$

for some intermediate kernel matrix $P \in \mathbb{R}^{n\times n}$ [4]. Substituting for (2.32) in (2.30) gives:

$$\begin{aligned} \boldsymbol{u}_k^* &= -R^{-1}B^\top P_{k+1}\boldsymbol{x}_{k+1} \\ &= -R^{-1}B^\top P_{k+1}(A\boldsymbol{x}_k + B\boldsymbol{u}_k) \end{aligned} \tag{2.33}$$

Pre-multiply both sides of (2.33) by R and simplify to yield the optimal control input in terms of the matrix P as:

$$\boldsymbol{u}_k^* = -(R + B^\top P_{k+1} B)^{-1} B^\top P_{k+1} A \boldsymbol{x}_k \tag{2.34}$$

To obtain consistent equations for the kernel matrix P, substitute (2.32) in the top part of the Hamiltonian system (2.31) to give:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A\boldsymbol{x}_k - BR^{-1}B^\top P_{k+1}\boldsymbol{x}_{k+1} \\ &= (I + BR^{-1}B^\top P_{k+1})^{-1} A\boldsymbol{x}_k \end{aligned} \tag{2.35}$$

Substituting (2.32) and (2.35) in the bottom part of the Hamiltonian system (2.31) gives:

$$\begin{aligned} P_k\boldsymbol{x}_k &= \mathcal{Q}\boldsymbol{x}_k + A^\top P_{k+1}\boldsymbol{x}_{k+1} \\ &= Q\boldsymbol{x}_k + A^\top P_{k+1}(I + BR^{-1}B^\top P_{k+1})^{-1} A\boldsymbol{x}_k \end{aligned} \tag{2.36}$$

Eliminating $\boldsymbol{x}_k$ from both sides of (2.36) and using the matrix inversion lemma:

$$(A_m + B_m D_m C_m)^{-1} = A_m^{-1} - A_m^{-1} B_m (D_m^{-1} + C_m A_m^{-1} B_m)^{-1} C_m A_m^{-1} \tag{2.37}$$

with $A_m = I$, $B_m = B$, $C_m = B^\top P_{k+1}$ and $D_m = R^{-1}$ gives:

$$\begin{aligned} P_k &= \mathcal{Q} + A^\top P_{k+1}\big(I - B(R + B^\top P_{k+1}B)^{-1}B^\top P_{k+1}\big)A \\ &= \mathcal{Q} + A^\top P_{k+1}A - A^\top P_{k+1}B(R + B^\top P_{k+1}B)^{-1}B^\top P_{k+1}A \end{aligned} \tag{2.38}$$

with boundary condition $P_N = E$. Sufficient condition for a solution is that the pair (A, B) is stabilisable on the set Ω [4]. Equation (2.38) is called the Riccati equation which in the case of the infinite horizon cost case i.e $\mathcal{R}(\boldsymbol{x}_N) \rightarrow 0$ as $N \rightarrow \infty$, becomes the discrete-time (DT) algebraic Riccati equation (ARE) given as:

$$P = \mathcal{Q} + A^T PA - A^\top PB(R + B^\top PB)^{-1}B^\top PA \tag{2.39}$$

DP can however be shown to provide the same results for the LTI system (2.24) with a quadratic cost (2.25) by using the Bellman principle of optimality. From the recursive form of (2.3), the value for the optimal control problem is computed as:

$$\begin{aligned} V(\boldsymbol{x}_k) &= \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma \sum_{n=k+1}^{N-1} \gamma^{n-(k+1)}\mathcal{R}(\boldsymbol{x}_n, \boldsymbol{u}_n) + \mathcal{R}(\boldsymbol{x}_N) \\ &= \boldsymbol{x}_k^\top \mathcal{Q}\boldsymbol{x}_k + \boldsymbol{u}_k^\top R\boldsymbol{u}_k + \gamma \sum_{n=k+1}^{N-1} \gamma^{n-(k+1)}\Big(\boldsymbol{x}_n^\top \mathcal{Q}\boldsymbol{x}_n + \boldsymbol{u}_n^\top R\boldsymbol{u}_n\Big) + \boldsymbol{x}_N^\top E\boldsymbol{x}_N \\ &= \boldsymbol{x}_k^\top \mathcal{Q}\boldsymbol{x}_k + \boldsymbol{u}_k^\top R\boldsymbol{u}_k + \gamma V(\boldsymbol{x}_{k+1}) \end{aligned} \tag{2.40}$$

given $V(0) = 0$. Equation (2.40) is the value function to the optimal control problem and it is assumed quadratic in the states in terms of a kernel matrix $P \in \mathbb{R}^{n\times n}$ given as:

$$V(\boldsymbol{x}_k) = \boldsymbol{x}_k^\top P_k\boldsymbol{x}_k \tag{2.41}$$

with $P_N = E$. Substituting (2.41) in (2.40) with $\gamma = 1$ gives:

$$\boldsymbol{x}_k^\top P_k\boldsymbol{x}_k = \boldsymbol{x}_k^\top \mathcal{Q}\boldsymbol{x}_k + \boldsymbol{u}_k^\top R\boldsymbol{u}_k + \boldsymbol{x}_{k+1}^\top P_{k+1}\boldsymbol{x}_{k+1} \tag{2.42}$$

A corresponding Hamiltonian function is defined as:

$$\begin{aligned} H\big(\boldsymbol{x}_k, \boldsymbol{u}_k, V(\boldsymbol{x}_k)\big) &= \boldsymbol{x}_k^\top \mathcal{Q}\boldsymbol{x}_k + \boldsymbol{u}_k^\top R\boldsymbol{u}_k + V(\boldsymbol{x}_{k+1}) - V(\boldsymbol{x}_k) \\ &= \boldsymbol{x}_k^\top \mathcal{Q}\boldsymbol{x}_k + \boldsymbol{u}_k^\top R\boldsymbol{u}_k + \boldsymbol{x}_{k+1}^\top P_{k+1}\boldsymbol{x}_{k+1} - \boldsymbol{x}_k^\top P_k\boldsymbol{x}_k \\ &= \boldsymbol{x}_k^\top \mathcal{Q}\boldsymbol{x}_k + \boldsymbol{u}_k^\top R\boldsymbol{u}_k + (A\boldsymbol{x}_k + B\boldsymbol{u}_k)^\top P_{k+1}(A\boldsymbol{x}_k + B\boldsymbol{u}_k) - \boldsymbol{x}_k^\top P_k\boldsymbol{x}_k \end{aligned} \tag{2.43}$$

Equation (2.43) is called the discrete-time Hamilton-Jacobi-Bellman (DT HJB) equation from which the optimal control input is obtained by differentiating with respect to $\boldsymbol{u}_k$ and equating to 0 as:

$$\begin{aligned} \frac{\partial H(\boldsymbol{x}_k, \boldsymbol{u}_k, V(\boldsymbol{x}_k))}{\partial \boldsymbol{u}_k} &= 2R\boldsymbol{u}_k + 2B^\top P_{k+1} A\boldsymbol{x}_k + 2B^\top P_{k+1} B\boldsymbol{u}_k = 0 \\ \therefore \boldsymbol{u}_k^* &= -(R + B^\top P_{k+1} B)^{-1} B^\top P_{k+1} A\boldsymbol{x}_k \end{aligned} \tag{2.44}$$

Substituting (2.44) in (2.42) and simplifying gives the same Riccati equation (2.38) as before, but using the DP approach. Therefore, for the LTI system (2.24) with quadratic cost (2.25), the optimal control policy $\mu(\cdot)$ is given by a linear feedback of the states as:

$$\boldsymbol{u}_k^* = -(R + B^\top P_{k+1} B)^{-1} B^\top P_{k+1} A\boldsymbol{x}_k = -K\boldsymbol{x}_k \tag{2.45}$$

where $K \in \mathbb{R}^{1\times n}$. For the general case where the system is modelled by nonlinear dynamics with non-quadratic cost, the control optimisation problem results in the nonlinear HJB equation which is known to be difficult or often impossible to solve analytically [9]. DP however provides a solution to these classes of problems through its systematic and recursive approach.

There are however challenges and limitations to the DP optimisation strategy one of which is widely known as the 'curse of dimensionality' of dynamic programming [41]. This stems from the discrete nature of the DP solution where each discrete state at each stage of the optimisation problem is associated with a discrete cost. In practical setting involving continuous states and actions, it is easy to see that the DP problem becomes non-trivial due to the infinite number of the possible states and actions. The curse of dimensionality therefore results from an explosion of either the state space $\mathbb{X} = \{x_1, x_2, \cdots, x_{N_x}\}$ which can take on D_x discrete values and thus $D_x^{N_x}$ possible state outcomes; or the action space $\mathbb{U} = \{u_1, u_2, \cdots, u_{N_u}\}$ which can take on D_u discrete values and thus $D_u^{N_u}$ feasible control outcomes. Moreover, there is an additional increase in memory usage associated with the increased state and action spaces and from storing the optimal values for each state at the different stages of the optimisation.

Furthermore, the DP approach is generally an offline method due to its backwards-in-time recursion and assumes that the system dynamics are well known. For stochastic systems, this involves evaluating the expectation over all the possible outcomes from the source of randomness. DP is therefore unable to cope with systems with unknown or varying dynamics commonly encountered in practice.

Approaches such as adaptive or approximate DP and RL are thus developed to cope with these challenges and limitations of the DP optimisation strategy.

2.2.2 Reinforcement learning frameworks

RL and ADP frameworks are widely employed to overcome the curse of dimensionality limitation of their DP counterpart. Formerly called adaptive critic designs (ACDs) by Werbos et al. [44], the approaches learn a network called the 'critic' that approximates the cost-to-go in the DP solution and have been known by different other labels including neuro-dynamic programming and critic global controller [45], [46], [47]. In contrast to the backwards-in-time iterative solution of the DP approach, ADP frameworks are enabled by iterative forward-in-time methods that utilise the Bellman optimality equation to develop value and policy update equations which are solved at each step of the iteration. Two of the popular iterative forward-in-time methods are the value iteration (VI) and policy iteration (PI) methods.

Value iteration methods: VI methods use the recursive Bellman optimality equation (2.4) as a value update equation that must be satisfied at each time step k from which a resulting control input can be computed forward-in-time as follows:

$$V_{k+1}(\boldsymbol{x}_k) = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma V_k(\boldsymbol{x}_{k+1}) \tag{2.46}$$

$$\boldsymbol{u}_{k+1} = \arg\min_{u_k} \Big(\mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma V_{k+1}(\boldsymbol{x}_{k+1}) \Big) \tag{2.47}$$

The VI method involves both a value update (2.46) and policy update (2.47) steps and are known to correspond to the contraction map (2.14) and (2.15) associated with the DP [9]. These iterative equations successively lead to improved policies and can be implemented online to determine the optimal control policies forward-in-time.

For convergence of the VI method, it is required that cost (2.3) is bounded and that the updates (2.46) and (2.47) are performed infinitely often for each state. Algorithm 2.1 gives the basic template for the VI method.

Policy iteration methods: In contrast to their VI counterpart, PI methods require an initially admissible policy (i.e. stabilising and with a finite cost $V(\cdot)$) and successively alternates between a policy evaluation and policy update steps

Algorithm 2.1 Value Iteration (VI) template

Initialise. For any initial policy $\mu_0(\boldsymbol{x})$, do till convergence:

Value update step. Update the value using (2.46) as:

$$V_{k+1}(\boldsymbol{x}_k) = \mathcal{R}(\boldsymbol{x}_k, \mu_k(\boldsymbol{x}_k)) + \gamma V_k(\boldsymbol{x}_{k+1})$$

Policy update step. Compute an improved policy using (2.47) as:

$$\mu_{k+1}(\boldsymbol{x}_k) = \arg\min_{\mu(\cdot)} \Big(\mathcal{R}(\boldsymbol{x}_k, \mu_k(\boldsymbol{x}_k)) + \gamma V_{k+1}(\boldsymbol{x}_{k+1})\Big)$$

as follows:

$$V_{k+1}(\boldsymbol{x}_k) = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma V_{k+1}(\boldsymbol{x}_{k+1}) \tag{2.48}$$

$$\boldsymbol{u}_{k+1} = \arg\min_{u_k} \Big(\mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma V_{k+1}(\boldsymbol{x}_{k+1})\Big) \tag{2.49}$$

Equations (2.48) and (2.49) serve as consistency equations from using the Bellman optimality equation and are solved at each time step k. Given a policy $\mu_k(\cdot)$, the value of the policy is evaluated by solving (2.48) till convergence and constitutes the policy evaluation step. For a system with finite state space, the policy evaluation step is equivalent to solving a linear system of equations. An improved policy is then computed using (2.49) and constitutes the policy update step.

The PI method is justified in [43] by showing that the improved policy μ_{k+1} ensures that $V_{k+1}(\boldsymbol{x}_k) \leq V_k(\boldsymbol{x}_k)$ and is associated with the monotonicity property (2.10) and (2.11) of the DP. This way the algorithm computes a strictly improved policy and convergence to the optimal policy and value under the assumption that the system is controllable has been shown in [43], [48]. Algorithm 2.2 gives the basic template for the PI method. In general, the VI methods are less computationally demanding than the PI methods as they require only a one step recursion in the value update step as opposed to solving the system of equations in the policy evaluation step. However, the PI methods are known to converge in fewer iterations [42], [49].

The iterative VI and PI methods are therefore online strategies as a result of their forward-in-time DP recursion using knowledge of the system dynamics and cost to obtain the optimum values. To enable model-free and forward-in-time online strategies, ADP uses function approximations to approximate the costs and

Algorithm 2.2 Policy Iteration (PI) template

Initialise. For any initial admissible policy $\mu_0(\boldsymbol{x})$, do till convergence:

Policy evaluation step. Evaluate the value of the current policy using (2.48) as:

$$V_{k+1}(\boldsymbol{x}_k) = \mathcal{R}(\boldsymbol{x}_k, \mu_k(\boldsymbol{x}_k)) + \gamma V_{k+1}(\boldsymbol{x}_{k+1})$$

Policy update step. Compute an improved policy using (2.49) as:

$$\mu_{k+1}(\boldsymbol{x}_k) = \arg\min_{\mu(\cdot)} \Big(\mathcal{R}(\boldsymbol{x}_k, \mu_k(\boldsymbol{x}_k)) + \gamma V_{k+1}(\boldsymbol{x}_{k+1})\Big)$$

solutions of the VI and PI methods. Based on the function that is been approximated, the ADP are broadly classified by Werbos [50] into heuristic dynamic programming (HDP), dual heuristic dynamic programming (DHP), action-dependent heuristic dynamic programming (AD-HDP) and action-dependent dual heuristic dynamic programming (AD-DHP). The HDP methods approximate the value function (i.e. $V(\boldsymbol{x})$) while their dual counterparts approximate both the cost and its gradient (i.e. $\frac{\partial V(x)}{\partial x}$). The action dependent variants further approximate the dependence of the control decisions on the value functions.

A variety of methods exist to solve the approximated costs such as the Monte Carlo (MC) and temporal difference (TD) methods [37]. MC methods are mainly implemented in simulation and learning occurs in an episodic manner i.e the approximated costs are updated using measurements obtained after a specified training period marked by system initialisation to some terminal conditions. In contrast, TD methods learn in an incremental fashion and can be implemented online using measurements obtained along the system trajectory. An hybrid approach called the TD(λ) allows for the combination of the incremental TD and the episodic MC methods [37]. This thesis considers the classification of the RL methods based on the control architecture using either the MC or TD approach and are presented next.

2.2.2.1 Actor only frameworks

Actor only frameworks are obtained by parameterising the control policy which is updated via some gradient descent tuning law in the direction of the cost function. These frameworks are closely related to the extremum seeking approach given in

[34], [35] and they proceed by approximating the control policy as:

$$\boldsymbol{u}_k = \mu(\boldsymbol{x}_k) \approx \theta_{a,k}^{\top} \Phi_a(\boldsymbol{x}_k) \tag{2.50}$$

where $\theta_a \in \mathbb{R}^{p_a}$ are the actor parameters and $\Phi_a(\boldsymbol{x}_k)$ is the basis function with p_a features. The actor update problem is to find the best parameters that minimise the desired cost function. It is assumed that the cost function (2.3) is differentiable with respect to the policy and also that the parameterised policy (2.50) is differentiable with respect to the parameters θ_a. A gradient descent tuning update law for the parameters can therefore be computed as:

$$\theta_{a,k+1} = \theta_{a,k} - l_a \nabla_{\theta_a} J \tag{2.51}$$

where $l_a > 0 \in \mathbb{R}$ is the learning rate or a tuning step size and $\nabla_{\theta_a} J = \frac{\partial J}{\partial \mu(\boldsymbol{x}_k)} \frac{\partial \mu(\boldsymbol{x}_k)}{\partial \theta_{a,k}}$. $\nabla_{\theta_a} J$ is directly estimated from the system or in simulation via a number of methods including finite differencing, likelihood ratio methods, REINFORCE method by Williams [51] and natural policy gradients [52], [53].

Convergence of the framework is inherited from the gradient descent tuning update law given unbiased gradient estimates and with the learning rate satisfying the following conditions [54]:

$$\sum_{k=0}^{\infty} l_{a,k} = \infty; \qquad \sum_{k=0}^{\infty} l_{a,k}^2 < \infty \tag{2.52}$$

A major drawback however is the problem of the large variance associated with the gradient estimates for the cost which may affect convergence of the approach.

2.2.2.2 Critic only frameworks

Critic only methods explicitly approximate the dependence of the control actions on the states and belong to the class of AD-HDP and AD-DHP algorithms. The approximated function is generally referred to as the *state-action* value function or Q-function and is given as:

$$Q^{\mu}(\boldsymbol{x}_k, \boldsymbol{u}_k) \approx \beta_k^{\top} \Psi(\boldsymbol{x}_k, \boldsymbol{u}_k) = \sum_{n=k}^{N} \gamma^{n-k} \mathcal{R}(\boldsymbol{x}_n, \boldsymbol{u}_n) \tag{2.53}$$

where $\beta \in \mathbb{R}^{p_q}$ are the Q-function parameters and $\Psi(\boldsymbol{x}_k, \boldsymbol{u}_k)$ is the basis function with p_q features. Equation (2.53) gives the Q-function approximation (QFA) which approximates the sum of the discounted reward signals $\mathcal{R}(\boldsymbol{x}, \boldsymbol{u})$ starting from state

$\boldsymbol{x}_k$ and taking action $\boldsymbol{u}_k$, then following policy $\mu(\boldsymbol{x})$ thereon. Using the Bellman optimality principle, the Q-function satisfies:

$$Q^*(\boldsymbol{x}_k, \boldsymbol{u}_k) = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma \min_{\boldsymbol{u}_k} Q^*(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}) \tag{2.54}$$

The state value function $V(\cdot)$ is related to the Q-function as follows:

$$V^*(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k} Q^*(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{2.55}$$

with the optimal control computed as:

$$\boldsymbol{u}_k^* = \arg\min_{\boldsymbol{u}_k} Q^*(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{2.56}$$

The critic network update problem is therefore to learn an approximate solution to the Bellman equation by minimising the Bellman error given in terms of the Q-function as:

$$e_{q,k} = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma Q^\mu(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}) - Q^\mu(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{2.57}$$

For the QFA approximation of (2.53), the Bellman error (2.57) is obtained based on either a VI or PI recursion as follows:

For VI recursion:

$$e_{q,k} = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma \beta_k^\top \Psi(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1}) - \beta_{k+1}^\top \Psi(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{2.58}$$

For PI recursion:

$$e_{q,k} = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) - \beta_{k+1}^\top \big(\Psi(\boldsymbol{x}_k, \boldsymbol{u}_k) - \gamma \Psi(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1})\big) \tag{2.59}$$

A suitable parameter estimation approach such as the batch least squares (BLS), recursive least squares (RLS), Kalman filter (KF) or gradient descent tuning can then be used to update the Q-function parameters till convergence. Following this, an optimal policy is computed by performing a greedy optimisation as:

$$\boldsymbol{u}_k^* = \arg\min_{u} \big(\beta_{k+1}^\top \Psi(\boldsymbol{x}_k, \boldsymbol{u}_k)\big) \tag{2.60}$$

Critic only methods are therefore classed as 'indirect' as they do not optimise directly over the policy space in comparison to their actor only counterpart. The methods generally provide good approximation to the value or Q-function, but may lack reliable guarantees in terms of near-optimality of the resulting policy [55], [56]. The described Q-learning approach is mainly implemented as an off-policy method, i.e. an independent exploratory or behavioural policy is used in the Q-function estimation during simulation [57]. A variant of the critic only method called state-action-reward-state-action (SARSA) is implemented as an on-policy TD method and uses the current policy with the state-action pair for its value estimation [37].

Convergence of the critic only method is guaranteed under the assumption that all the state-action pairs are visited infinitely often during learning, and that the monotonicity property associated with the Bellman optimality equation holds for (2.54) [43]. Consequently, the critic only approaches require more information for its approximations and convergence results in the literature are limited to simple systems with few states and actions [55].

2.2.2.3 Actor-critic frameworks

Actor-critic frameworks combine the strengths of both the actor and critic only frameworks by directly optimising over the policy space from the actor network, and combining with the low variance approximations of the value function from the critic network. The actor-critic frameworks are related to the class of HDP and DHP algorithms. Two networks are used for its approximations, the critic and actor networks, and respectively approximate the value function and control policy as follows:

$$V^{\mu}(\boldsymbol{x}_k) \approx \theta_{c,k}^{\top}\Phi_c(\boldsymbol{x}_k) = \sum_{n=k}^{N} \gamma^{n-k}\mathcal{R}(\boldsymbol{x}_n, \boldsymbol{u}_n) \tag{2.61}$$

where $\theta_c \in \mathbb{R}^{p_c}$ are the critic parameters and $\Phi_c(\boldsymbol{x}_k)$ is the basis function with p_c features. Equation 2.61 gives the value function approximation for the critic network which approximates the sum of the discounted reward signals $\mathcal{R}(\boldsymbol{x}, \boldsymbol{u})$ starting from state $\boldsymbol{x}_k$ under some fixed policy $\mu(\boldsymbol{x})$. Similar to the actor only framework, a second network approximates the control policy as:

$$\boldsymbol{u}_k = \mu(\boldsymbol{x}_k) \approx \theta_{a,k}^{\top}\Phi_a(\boldsymbol{x}_k) \tag{2.62}$$

with θ_a and Φ_a defined as before. The critic network aims to minimise the Bellman error which is given based on either a VI or PI recursion as:

VI recursion:

$$e_{c,k} = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \gamma\theta_{c,k}^{\top}\Phi(\boldsymbol{x}_{k+1}) - \theta_{c,k+1}^{\top}\Phi(\boldsymbol{x}_k) \tag{2.63}$$

PI recursion:

$$e_{c,k} = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) - \theta_{c,k+1}^{\top}\big(\Phi(\boldsymbol{x}_k) - \gamma\Phi(\boldsymbol{x}_{k+1})\big) \tag{2.64}$$

Following the update of the critic network, the actor network updates the control policy by minimising the critic estimates using gradient descent tuning as follows:

$$\begin{aligned} \theta_{a,k+1} &= \theta_{a,k} - l_a\nabla_{\theta_a}V(\boldsymbol{x}_k) \\ &= \theta_{a,k} - l_a\nabla_{\theta_a}\theta_{c,k+1}^{\top}\Phi(\boldsymbol{x}_k) \end{aligned} \tag{2.65}$$

with l_a defined as before and with $\nabla_{\theta_a}\theta_{c,k+1}^{\top}\Phi(\boldsymbol{x}_k) = \frac{\partial\theta_{c,k+1}^{\top}\Phi(\boldsymbol{x}_k)}{\partial\mu(\boldsymbol{x}_k)}\frac{\partial\mu(\boldsymbol{x}_k)}{\partial\theta_a}$. Evaluation of the critic gradient estimates is important to the actor-critic framework as it links the updates of the two initially separate networks together. The critic gradient estimates may be estimated during simulation or approximated using policy gradient theorem [56], [58]. In principle, the actor-critic update sequences lead to less oscillatory behaviour aiding its convergence as a change in the critic network is matched by a small variation in the policy determined by the learning rate l_a [59]. The actor-critic framework therefore represents one of the most commonly used RL methods and a schematic of the framework is shown in Figure 2.5.

Some applications of the RL ADP frameworks in the control literature include the control of discrete and continuous time dynamical systems, flight control systems, control of electrical power systems, dynamic energy management systems and the control of unmanned aerial vehicles to name a few. A review of the recent applications of RL in control which is the focus of this thesis warrants a further discussion and is presented in the next section.

2.3 Review of reinforcement learning in control applications

This section provides a review of recent advances of RL control applications to varying dynamical and complex systems of interest to this thesis. Control application of RL began with its investigation into the optimal control regulation problem of discrete-time systems [60], [61], [62]. These sets of results extended the

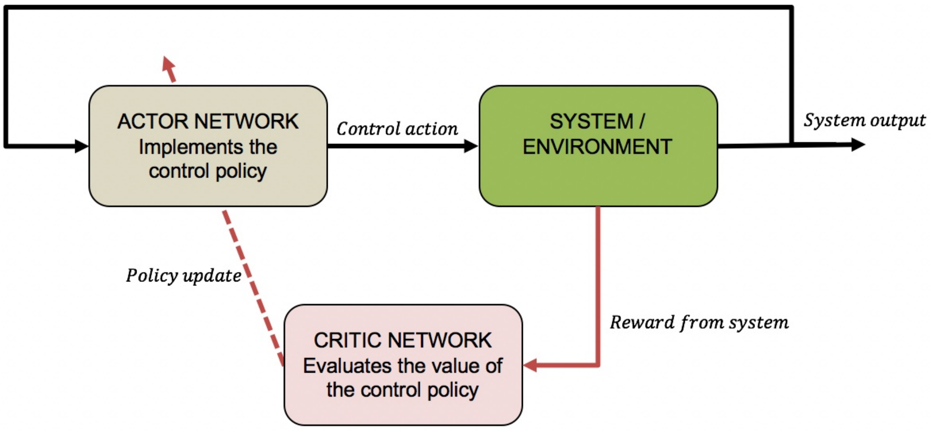


Figure 2.5: Schematic of the actor-critic reinforcement learning framework. The critic network learns and updates the value of the control action using rewards from the system while the actor network implements a learned optimal control action.

RL framework to the popular class of infinite horizon linear quadratic regulation (LQR) problem in which the dynamics are unknown or uncertain. Bradtke et al. [60] proposed a Q-learning framework using PI for the LQR problem and provided the persistence of excitation (PE) condition necessary for its convergence. A comparison of the RL framework with the conventional control theoretic solution to the LQR problem was carried out in [62] in the context of industrial manufacturing processes. Here, it was concluded that the performance of both approaches were very close, albeit the Q-learning RL approach required more probing noise for its approximations. Subsequent research works have therefore focused on more efficient realisations and extension of the RL frameworks to more practical control applications.

Extension of the other classes of the adaptive critic algorithms proposed by Werbos [50] (HDP, DHP, AD-HDP and AD-DHP frameworks) to the LQR control problem was given in [63] along with their convergence proofs. Landelius [63] concluded that the algorithms led to the convergence of the LQR control parameters to their optimal values using only the system measurements. Given terminal state penalties, a finite horizon control equivalent for the DT linear systems using the RL frameworks was shown in [64]. Applications in the continuous-time (CT) domain for the optimal state feedback control problem of linear systems using the adaptive critic designs have also been proposed in [65] and [66]. While Vrabie et al. [65] used a PI based framework that requires an initially stabilising policy, Bian and Jiang [66] proposed a VI based method that is not restricted to initially stabilising policies with both methods shown to converge to the optimal solutions.

For the DT nonlinear systems, Dierks and Jagannathan [67] proposed a time-based policy update RL framework using two neural networks (NN), the critic NN and the action NN to solve the infinite horizon control regulation problem. The approach assumes an initially stabilising policy, and tunes the two NN at regular predetermined intervals using a time history of performance measurements of the nonlinear system. Under the assumptions that the NN weight estimation errors are uniformly ultimately bounded (UUB) and that the NN approximation errors are negligible, it was shown that the estimated control policy asymptotically approaches the optimal values.

Similarly, for the regulation of CT nonlinear systems, Bhasin [49] proposed an actor-critic RL framework that uses system identification techniques to identify a model of the system dynamics online. A PE condition is given to ensure convergence of the framework and guarantee UUB stability of the closed-loop system. Likewise, Lv et al. [68] proposed an identifier-critic based RL framework for the optimal control of CT nonlinear systems that uses dual NN structure. The identifier NN learns a model of the system dynamics while the critic network learns an approximate solution to the nonlinear CT HJB equations from which the system policy can be derived.

Methods known as integral reinforcement learning (IRL) described in Vrabie et al. [65], [69] enables the development of RL frameworks for both optimal control of CT linear and nonlinear systems without the need for a model of the system dynamics required by other CT RL applications. The value function in the CT applications is expressed as an integral of the reward measurements, in contrast to the discrete summation in DT systems. Equivalent HJB equation in the CT domain therefore results in an expression that includes the full system dynamics making the CT RL applications more difficult to solve. Samples from the system measurements are collected at fixed time intervals to compute the integral reinforcement signals followed by a two-time scale asynchronous update process to sequentially update the weights of both the critic and actor networks.

A synchronous IRL equivalent that simultaneously updates both the critic and actor NN was shown in [70] along with a PE condition for its convergence. Extensions of the RL frameworks to include input-constrained CT nonlinear system applications have been proposed in [71], [72], [73] while Modares et al. [74] included the use of experience replay to relax the PE conditions needed for convergence. Other approximate approaches for CT domain applications have considered the Euler discretisation of the CT Bellman equations for which all the existing DT RL methods are subsequently applicable [9].

Other RL applications have considered the tracking control problem aimed at making the system outputs to follow desired reference trajectories. In [38], an infinite horizon LQT control for DT systems that makes use of an augmented RL state and reference dynamics formulation has been proposed. Application to practical systems is however limited as the approach assumes the use of reference dynamics that tend towards zero. An improved framework that relaxes the previous assumptions to the use of a discounted tracking cost for the DT linear system tracking application using RL was shown in [75] while an output-feedback (OPFB) equivalent was given in [76]. However, the use of a discounted tracking cost means that zero steady-state error cannot be guaranteed by the existing RL tracking frameworks. Further extensions of the tracking control problem to DT nonlinear systems to include the use of actor-critic RL structure with neural networks have been proposed in [77], [78], [80], [81], [83], [85], to multiple-input-multiple-output (MIMO) systems in [82], [84] and for a finite horizon case in [79].

Tracking control for the class of CT linear systems has been shown in [86] while Modares and Lewis [87] proposed an IRL framework for the tracking control in the CT systems using discounted cost, with an equivalent robust H_∞ approach shown in [88]. A model-free adaptive tracking algorithm using RL techniques has been proposed in [103] for the class of CT nonlinear systems while extensions to include input constraints have been shown in [89]. It is evident from the preceding paragraphs that there have been significant efforts in the development of novel RL algorithms for the control of both discrete and continuous-time systems. However, in all of the aforementioned algorithms, applications have been limited to simple systems for demonstration and tailored to the specific classes of systems in consideration. Other applications have considered extension of the RL algorithms to develop novel frameworks in complex flight control, power and energy management systems.

Ferrari and Stengel [46] proposed an adaptive control framework for the control of a six degree-of-freedom (DOF) simulated aircraft. The proposed framework employed a two-phase learning scheme, where the first phase initialises and matches parameters of a DHP network to chosen operating points using well established linear control theory, and the second phase adapts the DHP parameters to improve desired control response. This framework showed great potential in combining the advancements of the RL algorithms to real-life applications and foster the design of intelligent systems. Similar frameworks targeted at adaptive flight control applications using RL have been demonstrated by Ng et al. [90], [91] where a helicopter model was trained online to perform low speed manoeuvres, and by Abbeel et al. [92] for the aerobatic control of a real RC helicopter using differential dynamic programming.

In the power and energy management systems, notable advances using RL have been reported in [98], [99], [102] for applications in hybrid electric vehicles (HEVs). Motivated by the desire to achieve better fuel economy through efficient EMSs, Qi et al. [98] reported a 12% fuel saving by proposing a Q-learning RL method that takes into account the HEV's varying operating conditions to obtain optimal power-split control in real-time, in place of their traditional static rule-based strategy. Likewise, Zou et al. [99] integrated a RL framework with a power-request transition and control strategy for a hybrid tracked vehicle (HTV). The RL framework uses a model of the system running in tandem with a recursively updated power-request transition probability matrix to determine the best control strategy in real-time and was shown to achieve significant improvement in fuel efficiency. Xiong et al. [102] reported extensions of the RL strategy in [99] to take into account the health of the energy storage system (ESS) and different operating conditions, and achieved a 16.8% performance improvement as compared to a rule-based approach.

Further applications to the residential EMS has been reported in [93] where a novel RL algorithm was developed for the demand response (DR) control to optimise costs and risk of outages at peak periods by controlling the respective electrical load demands. The problem is formulated as a MDP and the approach uses a Q-learning framework which adapts to statistical changes in the residential energy systems to minimise the infinite horizon average financial cost and dis-utility to the consumer. Similar strategies using a NN based actor-critic framework to optimise the electricity cost for residential energy system consisting of combined power from the grid and ESS have been proposed in [95], [96], [97], [100] and extensions to multi-agent systems for smart microgrid solutions in [94], [101].

Table 2.1 provides a chronological chart of the discussed RL applications and advances. As we expect the continued relevance of RL in controls, there are however a number of open research issues in the aforementioned approaches and applications, and are discussed next.

2.4 Discussion

Application of RL has been shown in several applications where the full knowledge of the system dynamics is unavailable, and thus able to cope with varying or uncertain systems and achieve optimality. Whilst this motivates the RL approach, the problem characteristics of the different applications lead to different classes of RL frameworks. In addition, there are a number of open research problems that influence the type of RL framework for use in control applications, some of which are addressed in this thesis as follows:

1 **Use of direct measures of system performance metrics as reward signals:** RL is a goal-directed optimal control strategy that relies on the use of reward signals (positive or negative reinforcements) for its learning. Reward signals are therefore used to reflect desired performance objectives and to drive the RL search towards optimality. Typical reward signals used in RL control applications are analytically derived functions such as the quadratic state regulation function for the optimal control regulation and tracking [9], [72], [75]; the quadratic energy function for power and energy management systems to ensure regulation of the energy states to their desired reference values [95], [96], [98]; and the quadratic cost function on the spatial representation of states for flight control systems to train helicopter models to perform low speed manoeuvres [90], [91]. The performance of the resulting controllers is therefore dependent on its reward signals and defines different RL frameworks in the different application domains.

There are however a number of applications such as in degrading or varying systems where the conventional analytical reward functions prove inadequate. Designing an analytical reward function for such systems may be impossible to account for all the different variations and unknown degradation patterns. However, the use of direct measures of system performance such as fuel consumption, efficiency or life that reflect changes or variations in the systems as reward signals means that new ways are required for the control of the complex dynamical systems and to guarantee the system safety. Consequently, the design of a RL framework that makes use of the systems direct measurements as reward signals whilst providing a through-life optimal control and adaptive strategy remains an open research problem.

2 **Constraint handling in a RL framework:** The use of RL in constrained control applications remains an open research problem mainly because RL is based on dynamic programming which solves an unconstrained optimisation problem. Few applications in the literature have proposed input-constrained RL frameworks where bounded cost functions are used to limit the control signals to the constrained limits [72], [89], [104]. An example of such a bounded function that is typically used is the hyperbolic tangent function $\phi(\cdot) = tanh(\cdot)$ that satisfies $|\phi(\cdot)| \leq 1$. These methods are however limited to input constraints only. Further RL extensions to safety critical systems with constraints on both the states and inputs to ensure the safety/reliability of the systems are yet to be considered.

3 **Online optimal tracking control using reinforcement learning:** Existing RL techniques in the literature for the optimal tracking control problem either assume the use of a predetermined feedforward input for the tracking control, make restrictive assumptions on the reference model dynamics or use discounted tracking costs [38], [75], [77], [78], [79], [81], [87], [89]. By using discounted tracking costs, zero steady-state error cannot be guaranteed by the existing RL methods. The restrictive assumptions on the reference dynamics and discounted tracking costs makes the existing RL tracking methods less desirable for use in complex dynamical systems that may require precision tracking.

4 **Reinforcement learning controller integration:** Another important consideration in the development of RL frameworks in control applications is the overall system integration to enable the efficient and safe online learning and adaptation. Typical RL frameworks such as those discussed in the review have considered the RL controller as a standalone framework which completely neglects known information about the controlled system and learns the system control from scratch. For complex dynamical systems, these approaches may take a long time to learn and may require running the system to failure to learn the system's bounds and constraints. Whilst this is acceptable on some applications in robotics for exploration [105], other applications may require guarantees on the system performance both during learning and after adaptation. Techniques in the RL literature that ensure safe operation of the systems being controlled by reducing the risk and respecting the safety constraints are termed safe reinforcement learning frameworks [106], [107],

[108].

In [106], a safe RL approach that uses a gaussian process (GP) model to iteratively approximate a safe region of operation while learning the system's unknown dynamics has been proposed and demonstrated on a constrained cart-pole and quadrotor flight systems. In order to reduce the interference of the computed policy with the learning process, the approach further incorporates a safety factor in the RL performance metric. Likewise, Berkenkamp et al. [107] proposed a GP-based safe RL that starts from a pre-stabilised policy and proceeds by systematically using the system measurements to improve the policy alongside a GP model of the system used for safe exploration. However, the approaches are based on set-theoretic frameworks that assume that an estimated disturbance set used in the RL frameworks are known or can be computed. Consequently, Garcıa and Fernández [108] categorised the safe RL frameworks into - those that modify the optimality criterion through the use of safety factors, and those that modify the RL exploration either through the incorporation of external knowledge or through the use of a risk metric.

In this thesis, a potential strategy that considers the integration of the RL adaptations with a baseline controller structure suitable for practical implementation for the class of complex propulsion and power systems is explored. The frameworks proposed in this thesis therefore take into account these open research considerations and enable the efficient integration of the RL adaptations in practical systems, as well as considerations for the method of learning employed, choice of algorithm and the required PE condition needed for convergence of the overall scheme.

Table 2.1: Chronological chart of some novel reinforcement learning applications with the different proposed algorithms

Domain	Reinforcement learning frameworks										
DT linear systems	Q-learning with PI for LQR [60]	HDP, DHP, AD-HDP and AD-DHP with PI [63]	Q-learning LQR for industrial applications [62]				LQT [38]	LQT with discounted cost [75]	LQT using OPFB [76]	Finite horizon LQR [64]	
DT nonlinear systems				Greedy HDP tracker [77]	Tracker with online approximators [78]	Time-based policy update control [67] Finite-horizon tracker [79]	Dual-critic tracker [80]	Actor-critic tracker [81] Tracker with less learning parameters [82]	Tracker with NN approximation structure [83]	Fault-tolerant tracker [84]	Generalised PI tracker [85]
CT linear systems					PI based LQR [65]			LQT using ADP [86] LQT using IRL [87]	H_∞ LQT [88]	VI based LQR [66]	
Year	1994	1997	1998	2008	2009	2012	2013	2014	2015	2016	2017

continued from previous page

Domain	Reinforcement learning frameworks										
CT non-linear systems		Optimal control using NN HJB [71]		Asynchronous IRL control [69]		Model-based actor-critic control [49]	Input-constrained PI control [72]	Synchronous IRL control [70] Adaptive optimal RL control with input constraints [73] Input-constrained IRL control [74], [89]	Identifier-critic based control [68]		
Flight control systems	Adaptive critic controller [46]	Autonomous helicopter flight [90] Inverted helicopter flight [91]	Aerobatic helicopter flight [92]								
Power and energy management systems					Residential DR control [93]	Dynamic residential DR in smart microgrid [94]	Intelligent residential EMS with home battery connected to grid [95], [96]	Dual iterative Q-learning for residential EMS [97]	Optimal HEV power-split using Q-learning [98] Integrated RL framework for HTV [99] Dynamic load management for residential EMS [100]	EMS for integrated buildings and microgrid [101]	Improved Integrated RL framework for HTV [102]
Year	2002	2005	2007	2009	2010	2011	2013	2014	2016	2017	2018

Chapter 3

Reinforcement learning control frameworks for time-varying dynamical systems

This chapter presents the development of online reinforcement learning (RL) frameworks that are designed to be both adaptive and optimal for the control of time-varying dynamical systems. In contrast to the conventional adaptive controllers that are not designed to guarantee optimality by minimising desired performance objectives e.g. conventional adaptive schemes use system measurements to adapt models of the system or parameterised controllers and then use certainty of equivalence principle to synthesise new controls; the RL frameworks directly learn optimal control from minimising the desired performance objectives without prior knowledge of the system dynamics or the system variations.

The RL adaptation techniques are first shown for the linear quadratic regulation problem of discrete-time (DT) systems that converge to the optimum solutions subject to partially or completely unknown system dynamics. This is followed by the development of a candidate RL framework that advance the state-of-the-art to allow for the RL adaptations in existing baseline controller structures. An extension of the developed RL framework to the condition-based control of complex propulsion and power systems is then provided, where some of the open research problems highlighted in Section 2.4 are addressed. The strategies and results discussed in this chapter are based on the author's work in Sanusi et al. [109]. A summary of the main contributions presented in this chapter are as follows:

- A constraint handling scheme on both the system's inputs and outputs that solves a constrained optimisation problem in a RL framework is proposed.

This ensures the satisfaction of the system safety/reliability constraints whilst adapting the control inputs to optimal values.

- A dual-control loop structure in the implementation of the scheme that allows for the integration of the RL adaptations into an existing baseline controller and guarantees the system stability is developed. The overall framework maintains guarantees on the main system response whilst extracting improved performance by tuning extra degree-of-freedom (DOF) variables in a RL ADP control loop.

In the following, Section 3.1 develops the general RL frameworks applied to the optimal control regulation problem and presents a candidate RL framework that integrates the RL adaptations into existing controller structures. Section 3.2 presents the gas turbine condition-based control problem and the existing state-of-the-art baseline control architecture. This is followed by the extension of the proposed RL framework to the condition-based control problem in Section 3.3 along with the simulation results on representative engine test data.

3.1 Optimal control regulation of discrete-time systems using reinforcement learning

The general RL problem is concerned with determining policies that lead to improvement in a desired goal for systems with unknown or varying dynamics. RL uses the concept of reward (positive or negative reinforcements) observed from measurements to evaluate the performance of the current policy, and incrementally adapts the policy towards improving the desired goal [37]. For the optimal control regulation problem discussed in Section 2.2.1.2 for which baseline control solutions using both the calculus of variation and DP have been provided, we now wish to develop an online RL solution. Restated here, consider the optimal control regulation problem for the system described by the following discrete-time dynamics:

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k) + g(\boldsymbol{x}_k)\boldsymbol{u}_k \tag{3.1}$$

with state $\boldsymbol{x} \in \mathbb{R}^n$ and control input $\boldsymbol{u} = \mu(\boldsymbol{x}) \in \mathbb{R}^m$. The control input is governed by a deterministic feedback policy $\mu(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$ that maps the state space to the control space. A goal-directed optimal behaviour for the feedback policy may be given by the finite-horizon performance cost:

$$J(\boldsymbol{x}_k) = \mathcal{R}(\boldsymbol{x}_N) + \sum_{n=k}^{N-1} \gamma^{n-k} \mathcal{R}(\boldsymbol{x}_n, \boldsymbol{u}_n) \tag{3.2}$$

or in the infinite-horizon case where $\mathcal{R}(\boldsymbol{x}_N) \rightarrow 0$ as $N \rightarrow \infty$ as:

$$J(\boldsymbol{x}_k) = \sum_{n=k}^{\infty} \gamma^{n-k} \mathcal{R}(\boldsymbol{x}_n, \boldsymbol{u}_n) \tag{3.3}$$

where $\gamma \in [0, 1]$ is a discount factor and $\mathcal{R}(\boldsymbol{x}, \boldsymbol{u})$ is a scalar reward signal to measure the one-step cost of control under the feedback policy. $\mathcal{R}(\boldsymbol{x}, \boldsymbol{u})$ is evaluated using the standard quadratic energy function:

$$\mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{x}_k^\top \mathcal{Q} \boldsymbol{x}_k + \boldsymbol{u}_k^\top R \boldsymbol{u}_k \tag{3.4}$$

with $\mathcal{Q} \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ as positive definite weighting matrices. It is assumed that the system is both controllable and observable [4]. In addition to minimising and returning a finite cost, the optimal control regulation problem must also ensure that the feedback policy stabilises the closed loop system asymptotically on some set $\Omega \in \mathbb{R}^n$. Such a policy is said to belong to an admissible control set. As discussed in Section 2.2.2, RL frameworks make use of function approximations and iterative update equations of the forward-in-time methods (value iteration (VI) or policy iteration (PI)) to solve the optimal control problem online and without requiring *models* of the system. Based on the function that is being approximated and the iterative forward-in-time method employed, two RL frameworks are further developed in this chapter.

3.1.1 A value function approximation

A value function approximation (VFA) framework approximates the cost of Equation (3.3) for the infinite-horizon case as:

$$V(\boldsymbol{x}_k) \approx \theta_c^\top \Phi_c(\boldsymbol{x}_k) \tag{3.5}$$

where $\theta_c \in \mathbb{R}^{p_c}$ are the VFA parameters with basis function $\Phi_c(\boldsymbol{x})$. Update of the value parameters can be carried out using the temporal difference (TD) error generated by the Bellman recursion of the VI or PI methods. For the VI method, the TD error for the parameter updates is obtained at every time step k using measurements of the instantaneous reward signal $\mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k)$, the state $\boldsymbol{x}_k$ and the next state x_{k+1} as:

$$e_{c,k} = \mathcal{R}(x_k, u_k) + \gamma\theta_{c,k}^{\top}\Phi_c(x_{k+1}) - \theta_{c,k+1}\Phi_c(x_k) \tag{3.6}$$

with $\Phi_c(x_k)$ as the regressor vector. Conversely, the TD error for the parameter updates using the PI method is obtained as:

$$e_{c,k} = \mathcal{R}(x_k, u_k) - \theta_{c,k+1}^{\top}\big(\Phi_c(x_k) - \gamma\Phi_c(x_{k+1})\big) \tag{3.7}$$

with $\big(\Phi_c(x_k) - \gamma\Phi_c(x_{k+1})\big)$ as the regressor vector. Data from multiple time steps can be obtained for either iterative approach to determine the least squares solution for the VFA parameters and constitutes a batch least squares (BLS) procedure. Alternatively, standard recursive parameter estimation techniques such as the recursive least squares (RLS), Kalman filter (KF) or gradient descent tuning can be run till convergence to determine the best fit for the parameters that minimise the generated TD error.

Following the update of the VFA parameters, the policy for the control inputs is updated by equating to zero the derivative of the value function with respect to the control input and using the Bellman equation as follows:

$$\begin{aligned}\frac{\partial V(x_k)}{\partial u_k} &\approx \frac{\partial\theta_{c,k+1}^{\top}\Phi_c(x_k)}{\partial u_k} = 0\\ &= \frac{\partial\big(\mathcal{R}(x_k, u_k) + \gamma\theta_{c,k+1}^{\top}\Phi_c(x_{k+1})\big)}{\partial u_k} = 0\\ &= \frac{\partial\big(x_k^{\top}Qx_k + u_k^{\top}Ru_k + \gamma\theta_{c,k+1}^{\top}\Phi_c(x_{k+1})\big)}{\partial u_k} = 0\\ &= 2Ru_k + \gamma\frac{\partial\theta_{c,k+1}^{\top}\Phi_c(x_{k+1})}{\partial x_{k+1}}\cdot\frac{\partial x_{k+1}}{\partial u_k} = 0\end{aligned}$$

$$\therefore u_k^* = -\frac{\gamma}{2}R^{-1}g(x_k)^{\top}\nabla\theta_{c,k+1}^{\top}\Phi_c(x_{k+1}) \tag{3.8}$$

where $\nabla\theta_{c,k+1}^{\top}\Phi_c(x_{k+1}) = \frac{\partial\theta_{c,k+1}^{\top}\Phi_c(x_{k+1})}{\partial x_{k+1}}$. The update procedures therefore constitute both the value and policy update steps associated with the iterative forward-in-time methods and are repeated till convergence to the optimal control solutions.

An alternative approach for the policy update step is to introduce a second network to approximate the control inputs as:

$$\boldsymbol{u}_k^* = \mu^*(\boldsymbol{x}_k) \approx \theta_a^\top \Phi_a(\boldsymbol{x}_k) \tag{3.9}$$

where $\theta_a \in \mathbb{R}^{p_a}$ are the policy parameters with basis function $\Phi_a(\boldsymbol{x})$. A gradient tuning method can be used for the policy parameter updates with tuning index i as follows:

$$\begin{aligned}
\theta_a^{i+1} &= \theta_a^i - l_a \frac{\partial V(\boldsymbol{x}_i)}{\partial \theta_a} \\
&= \theta_a^i - l_a \frac{\partial V(\boldsymbol{x}_i)}{\partial \boldsymbol{u}_i} \times \frac{\partial \boldsymbol{u}_i}{\partial \theta_a} \\
&= \theta_a^i - l_a \Phi_a(\boldsymbol{x}_i)\big(2R\theta_a^{i\top}\Phi_a(\boldsymbol{x}_i) + \gamma \frac{\partial \theta_{c,k+1}^\top \Phi_c(\boldsymbol{x}_{i+1})}{\partial \boldsymbol{x}_{i+1}} \cdot \frac{\partial \boldsymbol{x}_{i+1}}{\partial \boldsymbol{u}_i}\big) \\
&= \theta_a^i - l_a \Phi_a(\boldsymbol{x}_i)\big(2R\theta_a^{i\top}\Phi_a(\boldsymbol{x}_i) + \gamma g(\boldsymbol{x}_i)^\top \nabla \theta_{c,k+1}^\top \Phi_c(\boldsymbol{x}_{i+1})\big)
\end{aligned} \tag{3.10}$$

where $l_a > 0$ is the tuning step size. Approximation of both the value and policy function with the use of two separate networks i.e. the critic and actor results in the general actor-critic RL framework. As discussed in Section 2.2.2.3, the actor-critic frameworks are preferred as they combine the strengths of both the actor and critic only frameworks. Algorithm 3.1 gives the template for the VFA based RL framework.

Remarks on Algorithm 3.1

- The gradient tuning update steps i can be chosen as the number of iteration steps j for the value function update.
- Knowledge of the input function $g(\boldsymbol{x})$ is required in the policy update step, thus, the VFA based RL algorithm is not completely model-free.
- To ensure convergence of the VFA parameter estimates, a persistence of excitation (PE) condition requires that the regressor vector satisfies [19]:

$$\alpha I \leq \sum_{i=k}^{k+M} \Gamma_i \Gamma_i^\top \leq bI \quad \forall i \tag{3.11}$$

where Γ is the regressor vector for the respective VI or PI method and with $M > 0$, $a > 0$, $b > 0$. This can be achieved in the implementation of the scheme by adding an exploration signal ϵ to the control inputs.

Algorithm 3.1 VFA based RL algorithm using PI

```
Initialise $V(x) \approx \theta_{c,k}^{\top}\Phi_c(x)$ at $k = 0$ for some stabilising policy $\mu(x) = \theta_{a,k}^{\top}\Phi_a(x)$, and do till convergence:
    **Value function update step**
for $j = 0$ till parameter convergence do
    At $x_j$, compute the control input $u_j$ with exploration signal $\epsilon$ as $u_j = \mu(x_j) + \epsilon$.
    Compute the least squares solution for $\theta_{c,j+1}$ using measurements $\mathcal{R}(x_j, u_j)$, $x_j$ and $x_{j+1}$ as:
        $\theta_{c,j+1}^{\top}\big(\Phi_c(x_j) - \gamma\Phi_c(x_{j+1})\big) = x_j^{\top} Q x_j + u_j^{\top} R u_j$
    $j = j + 1$.
end for
    **Policy update step**
Require: Set $\theta_{c,k+1} = \theta_{c,j+1}$
Update the policy parameters using the gradient descent tuning as:
    $\theta_{a,k}^{i+1} = \theta_{a,k}^{i} - l_a\Phi_a(x_i)\left(2R\theta_{a,k}^{i\top}\Phi_a(x_i) + \gamma g(x_i)^{\top}\nabla\theta_{c,k+1}^{\top}\Phi_c(x_{i+1})\right)$
At the end of the gradient tuning, set $\theta_{a,k+1} = \theta_{a,k}^{i+1}$ and update the policy as:
    $\mu(x) = \theta_{a,k+1}^{\top}\Phi_a(x)$
Increment time step $k = k + 1$.
```

3.1.2 A Q-function approximation

A Q-function approximation (QFA) framework explicitly approximates the dependence of the control inputs on the performance cost using a state-action value function or Q-function as:

$$Q(x_k, u_k) \approx \beta^{\top}\Psi(x_k, u_k) \tag{3.12}$$

where $\beta \in \mathbb{R}^{p_q}$ are the Q-function parameters with basis function $\Psi(x, u)$. Similar to the VFA framework, the Q-function parameters are updated using TD errors generated from the iterative VI or PI methods with measurements of the instantaneous reward signal $\mathcal{R}(x_k, u_k)$, the control input u_k, the state x_k and the next state x_{k+1}. This is defined for the VI method as:

$$e_{q,k} = \mathcal{R}(x_k, u_k) + \gamma\beta_k^{\top}\Psi(x_{k+1}, u_{k+1}) - \beta_{k+1}^{\top}\Psi(x_k, u_k) \tag{3.13}$$

with $\Psi(\boldsymbol{x}_k, \boldsymbol{u}_k)$ as the regressor vector or with the PI method as:

$$e_{q,k} = \mathcal{R}(\boldsymbol{x}_k, \boldsymbol{u}_k) - \beta_{k+1}^\top \big(\Psi(\boldsymbol{x}_k, \boldsymbol{u}_k) - \gamma \Psi(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1})\big) \tag{3.14}$$

with $\big(\Psi(\boldsymbol{x}_k, \boldsymbol{u}_k) - \gamma \Psi(\boldsymbol{x}_{k+1}, \boldsymbol{u}_{k+1})\big)$ as the regressor vector. Standard parameter estimation techniques such as the BLS, RLS, KF or gradient descent tuning can be equally used to determine the best fit for the QFA parameters that minimise the generated TD error.

In contrast to the VFA framework that requires knowledge of the input function in its policy update step, the QFA framework computes this without knowledge of the system dynamics as the updated Q-function contains the control inputs as arguments. A greedy optimisation is thus performed in the QFA policy update step as:

$$\boldsymbol{u}_{k+1} = \arg\min_{u} \big(\beta_{k+1}^\top \Psi(\boldsymbol{x}_k, \boldsymbol{u}_k)\big) \tag{3.15}$$

Both the Q-function parameter and policy update steps are repeated till convergence to the optimal control solutions and Algorithm 3.2 gives the template for the QFA based RL framework. The use of a single network for the QFA results in the general critic only RL framework.

Remarks on Algorithm 3.2

- For convergence, a PE condition requires that the regressor vector for the QFA satisfies Equation 3.11. This can be achieved by adding an exploration signal ϵ similar to the VFA method.

- Algorithm 3.2 assumes an on-policy implementation i.e. the policy that is being updated is also the same used during training or for exploration. The policy is kept fixed while generating samples, and only updated after convergence of the Q-function parameters.

Control regulation example using both the VFA and QFA based RL algorithms

To demonstrate both the VFA and QFA based RL frameworks, consider the infinite-horizon control regulation problem of a 2 state linear system with initially unstable

Algorithm 3.2 QFA based RL algorithm using PI

```
Initialise $Q(\boldsymbol{x},\boldsymbol{u}) \approx \beta_k^\top \Psi(\boldsymbol{x},\boldsymbol{u})$ at $k = 0$ for some stabilising policy $\mu(\boldsymbol{x}) = \arg\min_u \left(\beta_k^\top \Psi(\boldsymbol{x},\boldsymbol{u})\right)$, and do till convergence:
    Q-function update step
for $j = 0$ till parameter convergence do
    At $\boldsymbol{x}_j$, compute the control input $\boldsymbol{u}_j$ with exploration signal $\epsilon$ as $\boldsymbol{u}_j = \mu(\boldsymbol{x}_j) + \epsilon$.
    Compute the least squares solution for $\beta_{j+1}$ using measurements $\mathcal{R}(\boldsymbol{x}_j,\boldsymbol{u}_j)$, $\boldsymbol{u}_j$, $\boldsymbol{x}_j$ and $\boldsymbol{x}_{j+1}$ as:
        $$\beta_{j+1}^\top \left(\Psi(\boldsymbol{x}_j,\boldsymbol{u}_j) - \gamma\Psi(\boldsymbol{x}_{j+1},\boldsymbol{u}_{j+1})\right) = \boldsymbol{x}_j^\top Q\boldsymbol{x}_j + \boldsymbol{u}_j^\top R\boldsymbol{u}_j$$
    where $\boldsymbol{u}_{j+1} = \mu(\boldsymbol{x}_{j+1})$
    $j = j + 1$.
end for
    Policy update step
Require: Set $\beta_{k+1} = \beta_{j+1}$
Update the policy parameters using a greedy optimisation as:
        $$\mu(\boldsymbol{x}) = \arg\min_u \left(\beta_{k+1}^\top \Psi(\boldsymbol{x},\boldsymbol{u})\right)$$
Increment time step $k = k + 1$.
```

dynamics given as:

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} -1.0 & 2.0 \\ 2.2 & 1.7 \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} 2.0 \\ 1.5 \end{bmatrix} \boldsymbol{u}(t) \tag{3.16}$$

Using Euler's discretisation with a sampling time $t_s = 0.03s$, the corresponding DT system dynamics becomes:

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} 0.9724 & 0.0607 \\ 0.0668 & 1.0544 \end{bmatrix}}_{A} \boldsymbol{x}_k + \underbrace{\begin{bmatrix} 0.0605 \\ 0.0482 \end{bmatrix}}_{B} \boldsymbol{u}_k \tag{3.17}$$

The control regulation problem aims to regulate the states for system (3.16) to zero from any finite initial condition $\boldsymbol{x}_0$. The regulation cost parameters are given as $\gamma = 1$, $Q = \begin{bmatrix} 0.05 & 0 \\ 0 & 0.05 \end{bmatrix}$ and $R = 0.03$. Approaches such as the calculus of variation or DP discussed earlier can be used to compute the optimal state feedback gain and the corresponding Riccati matrix using full knowledge of the system dynamics (A,B). Optimal control solutions for the regulation problem can however be obtained using Algorithm 3.1 and 3.2 without knowledge of the system dynamics and using only measurements of the states and reward signals.

For the VFA framework, the critic network is approximated with a quadratic basis function since the value function for the general LQR problem is known to be quadratic as:

$$V(\boldsymbol{x}) \approx \theta_c^\top \Phi_c(\boldsymbol{x}) = \theta_c^\top \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix} \tag{3.18}$$

with $\theta_c \in \mathbb{R}^3$ while a linear basis function approximates the feedback control policy in the actor network as:

$$\mu(\boldsymbol{x}) \approx \theta_a^\top \Phi_a(\boldsymbol{x}) = \theta_a^\top \begin{bmatrix} x_1 & x_2 \end{bmatrix} \tag{3.19}$$

with $\theta_a \in \mathbb{R}^2$. The critic and actor network parameters respectively correspond to the Riccati matrix P and feedback gain K of the LQR from Section 2.2.1.2 as:

$$\begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} = \begin{bmatrix} \theta_c^{(1)} & 0.5\theta_c^{(2)} \\ 0.5\theta_c^{(2)} & \theta_c^{(3)} \end{bmatrix}$$
$$\begin{bmatrix} K_1 & K_2 \end{bmatrix} = \begin{bmatrix} \theta_a^{(1)} & \theta_a^{(2)} \end{bmatrix} \tag{3.20}$$

Figure 3.1 shows the online convergence of both the critic parameters $\theta_c^* = P^* = \begin{bmatrix} 0.5109 & 0.3677 \\ 0.3677 & 1.3994 \end{bmatrix}$ and the corresponding actor parameters $\theta_a^* = K^* = \begin{bmatrix} 1.4299 & 2.6169 \end{bmatrix}$ to the baseline optimal control solutions.

Similarly for the QFA framework, the Q-function is approximated with a quadratic basis function as:

$$Q(\boldsymbol{x}, \boldsymbol{u}) \approx \beta^\top \Psi(\boldsymbol{x}, \boldsymbol{u}) = \beta^\top \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 u \\ x_2^2 \\ x_2 u \\ u^2 \end{bmatrix} \tag{3.21}$$

with $\beta \in \mathbb{R}^6$. This gives the critic network approximation and the parameters correspond to the Q-function of the LQR in terms of the Riccati matrix P and reward signal $\mathcal{R}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{x}^\top \mathcal{Q}\boldsymbol{x} + \boldsymbol{u}^\top R\boldsymbol{x}$ as:

$$\begin{bmatrix} \mathcal{Q} + A^\top PA & A^\top PB \\ B^\top PA & R + B^\top PB \end{bmatrix} = \begin{bmatrix} \beta^{(1)} & 0.5\beta^{(2)} & 0.5\beta^{(3)} \\ 0.5\beta^{(2)} & \beta^{(4)} & 0.5\beta^{(5)} \\ 0.5\beta^{(3)} & 0.5\beta^{(5)} & \beta^{(6)} \end{bmatrix} \tag{3.22}$$

Figure 3.2 shows the online convergence of the critic parameters to the baseline optimal control solutions.

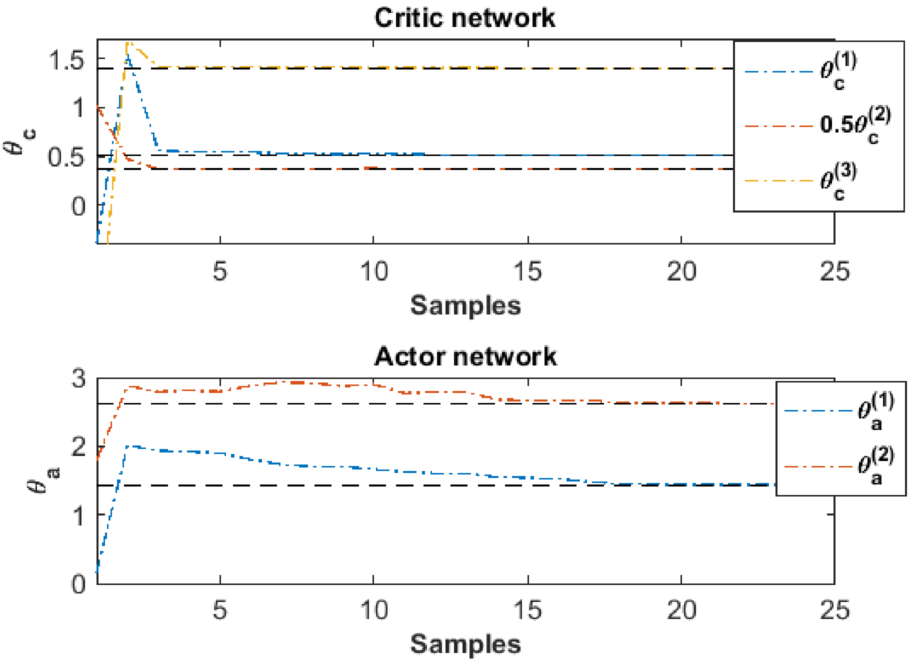


Figure 3.1: Convergence of the critic and actor network parameters using the value function approximation with policy iteration algorithm for the optimal control regulation problem.

Having shown the basic development and application of the two RL frameworks to the optimal control regulation problem, we now wish to provide an adaptation control architecture under which the frameworks can be extended to practical applications. Specifically, we consider integration with existing controller structures and using RL to continually adapt the system control subject to gradual variations in the system dynamics or performance. The overall frameworks move from a simple control of a system towards a through-life performance optimisation and adaptation strategy.

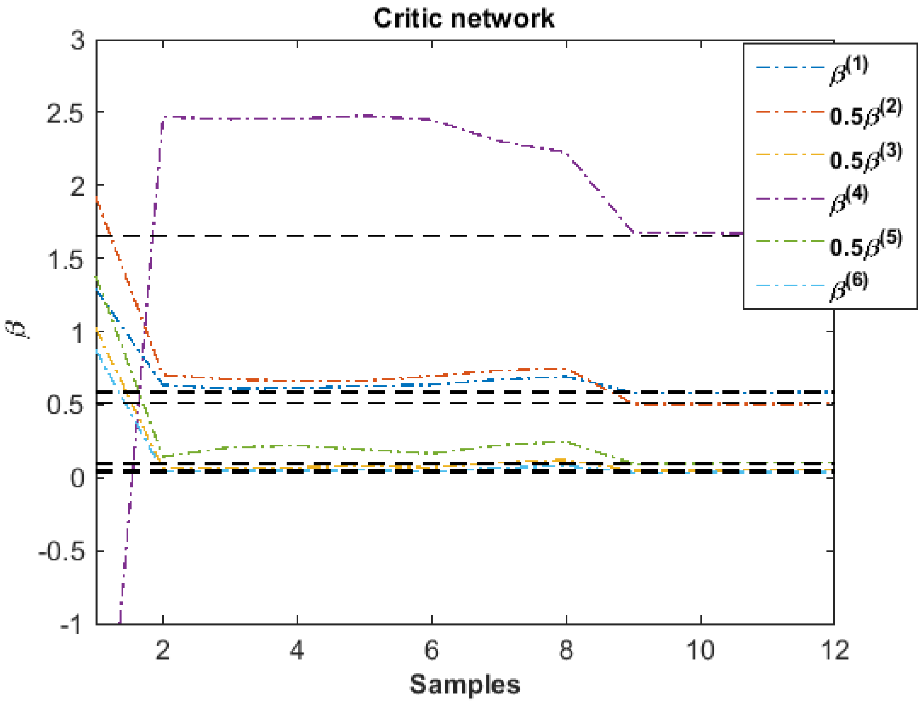


Figure 3.2: Convergence of the critic network parameters using the Q-function approximation with policy iteration algorithm for the optimal control regulation problem.

3.1.3 Control architecture for the reinforcement learning adaptations

RL provides techniques by which to learn optimal controllers over time by simply interacting with the system and without any guidance to the nature of the dynamics of the underlying system. Whilst this has been shown to work on simple systems, direct extension for use in dynamic and complex propulsion and power systems such as aircraft and space systems may be impractical for safety reasons. For these systems, the knowledge of the system dynamics and physics are fairly known - the baseline control system is therefore considered matured. To neglect all the information and treat the complex systems as a completely black-box model will be counter-intuitive. Furthermore, the class of complex systems in consideration are considered safety critical by requiring guarantees on the overall system performance, both during transient and at steady-state.

However, variations in the system due to degradation, engine build differences and changing operating conditions are not well understood and may be difficult to account for using mathematical models. As discussed in the previous chapters, RL provides techniques by which to compensate for the unknown variations and achieve desired optimality. Consequently, this thesis proposes a framework that integrates the RL adaptations into existing baseline controller structures to compensate for the system variations and maintain the desired level of system performance. These RL integrations may assume different topologies depending on either an open-loop or closed-loop integration, and are namely - feedforward and feedback RL adaptations.

Measurements of the desired performance quantities (reward signals) are used in the RL framework to continually tune or trim the baseline open-loop or closed-loop gains; with both RL adaptations using the additional reward signals in an implicit feedback loop. The proposed framework therefore assumes a hierarchical structure with the baseline controllers ensuring stability and inner loop regulation, while the RL adaptation loop continually adapts the controller gains to desired optimum values as the system varies. To allow for safe learning, the RL adaptations of the system controller gains are performed only at convergence of the RL algorithms - this way, the transient learning instability is minimised. Figure 3.3 shows a candidate framework that integrates both the feedforward and feedback RL adaptations into existing controller structures for use in complex propulsion and power systems. The next section introduces the first extension of the RL frameworks to the condition-based control of gas turbine engines.

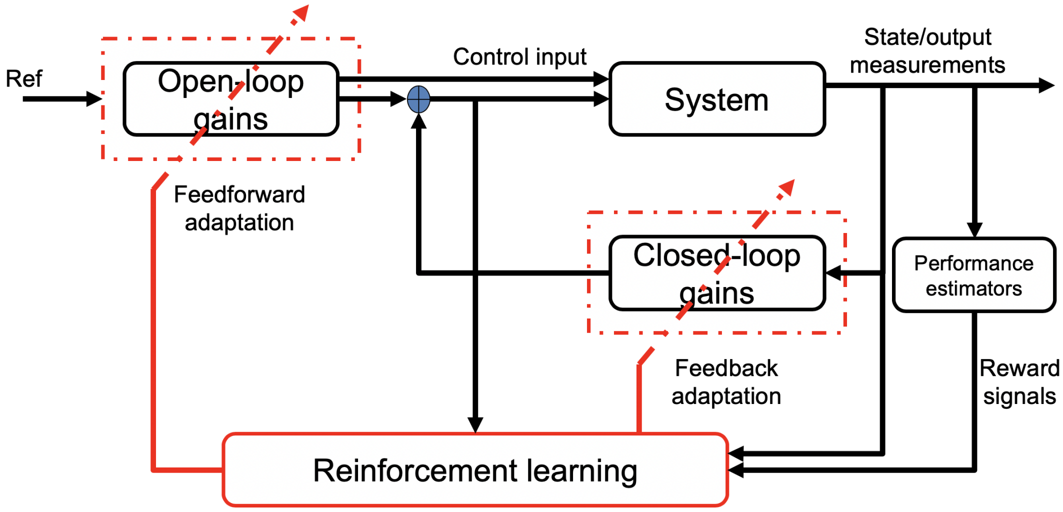


Figure 3.3: Block diagram of the candidate framework that integrates both the feedforward and feedback reinforcement learning adaptations into existing controller structures.

3.2 Reinforcement learning control framework for complex propulsion and power systems

Most engineering systems are subject to degradation, yet their control systems are not designed to explicitly account for it. While the dynamics that govern the operation of the systems are usually modelled or identified for the control design, the degradation dynamics are not; typically, these evolve over long timescales and in non-deterministic ways. This affects the states of the component health of the systems resulting in reduced performance and increased fuel consumption over time [3]. Opportunities to mitigate the effects of degradation therefore cannot be over emphasised as in the case of the civil gas turbine engine (GTE) where the cost of fuel accounts for about 15% to 25% of the total aircraft operating cost [11]. In addition to the gradual degradation, performance of gas turbine engines (GTEs) is also affected by fleet variations from engine build differences and changing operating conditions. Optimising the system performance as a result of these varying factors pose a major challenge to the GTE control.

The unknown degradation dynamics and variations affecting the GTE states are reflected as changes in the measured/estimated system performance characteristics such as the system efficiency index and life [11]. Whilst monitoring of these performance characteristics can help to reduce the cost of operation from economic and performance perspectives, the opportunities to complement the GTE control design have received little attention e.g. monitoring of the fuel consumption and component temperatures have been used to predict the system life necessary for maintenance scheduling but not for feedback control [110]. It is therefore increasingly important to use the information about the system performance characteristics in optimising the GTE control whilst considering the reliability of its implementation.

Techniques that enable such capabilities are generally termed *condition-based* and are aimed at maintaining the system's safety and reliability whilst optimising the system performance. Condition-based control (CBC) techniques can therefore be classed as types of adaptive control frameworks that focus on optimising to slow and varying changes in the system performance. This, combined with an appropriate adaptation strategy and architecture increases the feasibility of the framework to a fully intelligent control and health management technology for industrial applications. The GTE provides a good illustrative example to show case the condition-based control problem when considered as a power delivery system, but the techniques presented in this thesis are widely applicable.

3.2.1 Gas turbine engine system

Gas turbine engines (GTE) consist of a fan and compressor system to draw in and compress air; a combustor to mix and burn fuel with the compressed air; and a turbine to extract power or thrust from the generated hot stream of air or through a bypass system [2]. The system control is responsible for regulating thrust by setting measurable proxy parameters such as fan or shaft speeds, engine pressure ratio (EPR), pressure or temperature, and also providing engine limit protection. Limit protection in GTE includes compressor surge or stall and burner blowout protection [1]. Regulation of the thrust proxy parameters is then achieved via the control of fuel flow in closed-loop, with modern engines having other variable geometry components (VGCs) such as the variable stator vanes (VSVs), variable bleed valves (VBVs), variable inlet guide vanes (VIGVs) and the exhaust nozzle area in open-loop for compressor performance improvement [1], [11], [111].

Furthermore, the engine control is divided into both transient and steady-state control. Transient control enables system acceleration or deceleration with predetermined fuel flow schedules that provide limit protection while the steady-state control maintains engine operation along desired steady-state operating lines. The steady-state control is implemented in a feedback loop where the thrust parameter error between the desired reference and sensed values is fed into the controller to drive a fuel metering valve. This represents the main control loop and is the only closed-loop control in commercial GTE [1]. A schematic of the control elements in the main control loop is given in Figure 3.4.

Other VGCs are controlled in open-loop via fixed gain schedules designed for optimum operation at design points, which are usually acceleration, deceleration and cruise. These schedules are made dependent on engine shaft speeds, or some other measurable quantities that reflect engine operating conditions. Examples are the variable inlet guide vanes (VIGVs) and VSVs that operate between fixed low speed (closed) and high speed (open) positions to maintain the optimum angle of attack on the compressor blades, and maintain system stability. Figure 3.5 shows a simplified schematic of a typical fixed gain schedule for the VSVs.

The VGCs are however known to have a large effect on the GTE performance such as on fuel consumption [10], [112] and provide extra degrees of freedom

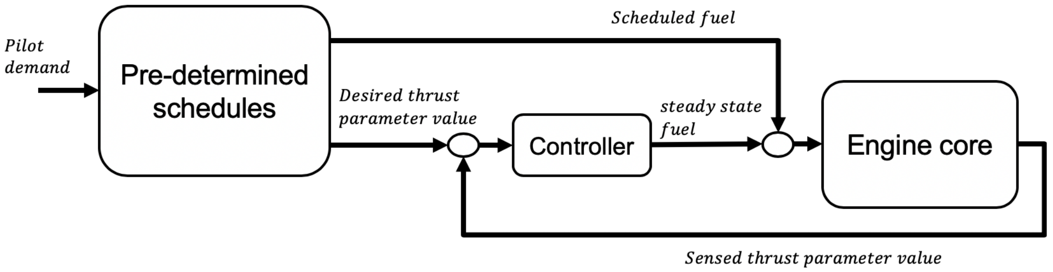


Figure 3.4: Block diagram of a typical main control loop of gas turbine engines showing fuel flow control at steady-state with acceleration and deceleration schedules that provide limit protection. (*adapted from [1]*).

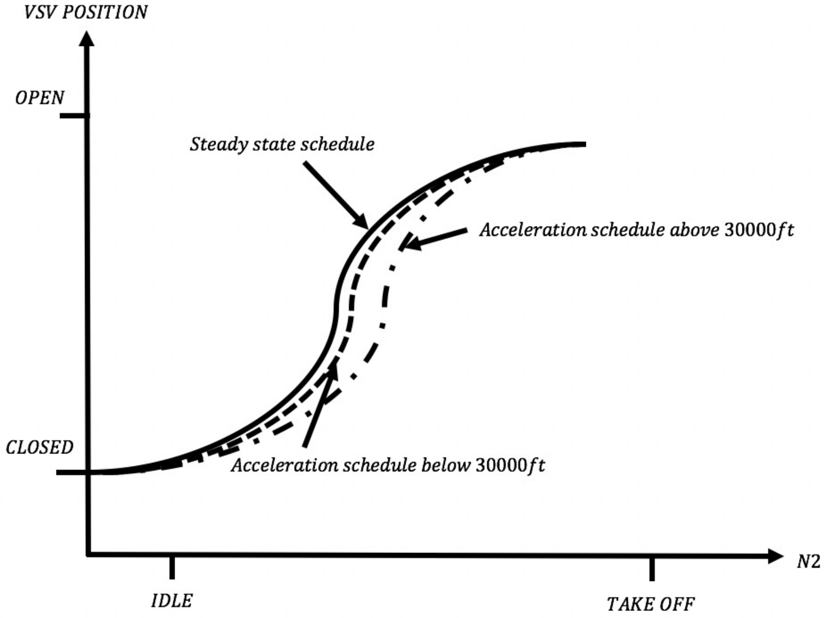


Figure 3.5: Simplified schedule for variable stator vanes dependent on shaft speed (N_2), at steady-state and acceleration operating conditions (*adapted from [2]*).

(DOF) to the GTE control. As a consequence of these fixed controller schedules, natural engine degradation, coupled with engine-to-engine variations cause shifts in optimal operating points resulting in reduced performance and increased fuel consumption. Degradation induced shifts also affect the pilot throttle-to-thrust parameter relationship leading to further loss in performance and increased overall system life cycle costs [113], [114]. An opportunity for performance improvement which involves a variety of measures including the periodic adjustment of the fixed gain schedules to system condition have been reported in Kurz and Brun

[10] and Bringhenti and Barbosa [112].

A CBC framework can therefore be developed for the GTE that continually optimises the desired system performance by taking into account the effects of system degradation, changing operating conditions and other system variations whilst maintaining the system safety and stability. A mitigating strategy is first considered in this chapter for the open-loop part of the system that tunes the VGC controller set-points in order to recover performance while the closed-loop part is considered in the subsequent chapters.

3.2.2 Condition-based control problem formulation

A conceptual mathematical model for the GTE system described in the previous section can be given as [109]:

$$\boldsymbol{x}_{k+1} = F(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{d}_k) \tag{3.23}$$

where $F(\cdot)$ represents the system dynamics and with $\boldsymbol{x} \in \mathbb{R}^n$ as the system thrust proxy states such as the shaft speeds (NH), EPR, pressure and temperature. The control inputs $\boldsymbol{u} = \begin{bmatrix} u^{main} \\ u^{aux} \end{bmatrix} \in \mathbb{R}^m$ consist of the main fuel flow input denoted by $u^{main} \in \mathbb{R}^{m_1} \subset \mathbb{R}^m$ and the additional DOF control parameters such as the VGCs employed in many GTE designs denoted by $\boldsymbol{u}^{aux} \in \mathbb{R}^{m_2} \subset \mathbb{R}^m$. The component health states $\boldsymbol{d} \in \mathbb{R}^d$ denote the system performance characteristics such as the compressor and turbine efficiencies that change slowly over time due to degradation [3]. Typically, $\boldsymbol{d}$ is difficult to estimate as it is governed by non-deterministic processes which vary across fleets and from engine to engine. Conventional thrust regulation is thus achieved by designing the control system at some identified nominal models of the system (i.e. at predetermined worst-case configuration of $\boldsymbol{d}$) [3], [2], [1].

Assumption 1. The main control loop is assumed to be pre-stabilised by a baseline controller of the form $\boldsymbol{u}^{main} = h(\boldsymbol{y})$. The controller regulates the thrust proxy state measurements $\boldsymbol{y} = c(\boldsymbol{x})$ to their desired reference values i.e $\boldsymbol{y} \rightarrow \boldsymbol{y}_{ref}$ such that the thrust response is guaranteed. This represents the conventional main control loop with $\boldsymbol{y}$ as the primary system measurements. The regulated states are kept within their prescribed limits using limit management controllers such as the min-max limiter logic [115].

The VGCs ($\boldsymbol{u}^{aux}$) on the other hand are typically set via fixed (open-loop) gain schedules against the system outputs or flight parameters σ (such as altitude, mach number (Mn) and temperature) [2] and designed for the worst case degradation condition.

Assumption 2. Secondary system measurements denoted as $\boldsymbol{y}^p$, that reflect changes in the system performance characteristics mainly due to degradation are assumed to be available. These measurements are normally used for engine health monitoring to schedule maintenance actions and are hitherto not used for control [3]. Additional measurements that provide limitations for the GTE safety and stability denoted as $\boldsymbol{g}^p$, are equally assumed to be available. These limits are calculated through a standard design practice to 'stack' uncertainties (actuation and sensing errors, operational uncertainties e.t.c.) into safety margins for the main control loop [116].

The CBC challenge within the current GTE control architecture is to use the secondary system measurements $\boldsymbol{y}^p$ in addition to the primary system measurements $\boldsymbol{y}$ for control decisions such that:

- The system maintains the desired thrust response control i.e. $\boldsymbol{y}_k \rightarrow \boldsymbol{y}_{ref}$ as $k \rightarrow \infty$.
- The system performance measurements are optimised subject to the gradual engine degradation i.e. $\min \sum_{n=k}^{\infty} \boldsymbol{y}_n^p$.
- The system safety/stability is guaranteed i.e. the measurements $\boldsymbol{g}_k^p \leq \boldsymbol{G}_{limits}$ $\forall k$ where $\boldsymbol{G}_{limits}$ are specified design limits.

A candidate solution approach to the CBC problem is therefore to devise a feedback tuning strategy for the $\boldsymbol{u}^{aux}$ in place of their conventional fixed gain scheduling by solving a performance optimisation problem as:

$$\begin{aligned} \boldsymbol{u}_k^{*aux} &= \arg\min_{u^{aux}} \sum_{n=k}^{\infty} \boldsymbol{y}_n^p \\ \textit{subject to: } \boldsymbol{x}_{k+1} &= F(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{d}_k) \\ \boldsymbol{y}_k &= c(\boldsymbol{x}_k) \\ \boldsymbol{u}_k^{main} &= h(\boldsymbol{y}_k) \\ \boldsymbol{g}_k^p &\leq \boldsymbol{G}_{limits} \end{aligned} \tag{3.24}$$

Solving (3.24) is difficult due to the unknown system degradation dynamics in $F(\cdot)$ from (3.23). Standard model-based control approaches may thus be impractical as discussed in Section 2.1.2. Furthermore, a standard system identification approach for the optimal control of the system will result in the nonlinear HJB equations which are often impossible to solve analytically [4]. RL solves the problem by not requiring models of the system but incrementally improves the desired control performance using the system measurements. The proposed solution approach is given in the next section.

3.3 Reinforcement learning for the condition-based control of gas turbine engines

RL problem is concerned with optimising the expected value of desired cost through a sequence of observations, actions and rewards over time [37]. Practical methods for solving the RL problems have been based on ADP and function approximations as discussed in Section 2.2.2. In comparison with the given optimal control regulation problem, the CBC poses a number of open RL research problems in practical applications as highlighted in Section 2.4 in the following ways:

- The optimal control regulation problem uses the conventional quadratic reward function $\mathcal{R}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{x}^\top Q\boldsymbol{x} + \boldsymbol{u}^\top R\boldsymbol{u}$. However, the strength of RL lies in its flexibility to use other 'crafted' or direct reward measurements towards achieving the desired goal. The CBC problem proposes to use the system's direct performance measurements that reflect changes due to gradual degradation as reward signals, and remains unexplored in RL control applications.

- RL frameworks for use in the CBC problem must consider the satisfaction of safety/reliability constraints whilst optimising the control inputs.

- The GTE is a complex system with numerous constraints for each component, resulting in a series of single-input-single-output controllers with fixed gain schedules. Careful considerations for the integration of the RL framework within the GTE architecture is therefore essential in providing a certifiable and working adaptation strategy.

3.3.1 RL-ADP condition-based control solution

Let the desired cost for the GTE CBC problem to optimise the VGC control parameters at discrete time steps k be given as:

$$Q(\boldsymbol{x}_k, \boldsymbol{u}^{aux}(\boldsymbol{x}_k)) = \sum_{n=k}^{N} \gamma^{n-k} \mathcal{R}(\boldsymbol{x}_n, \boldsymbol{u}_n) \tag{3.25}$$

where N is the discrete time interval considered for the optimisation, $0 \leq \gamma \leq 1$ is the discount factor and $\mathcal{R}(\boldsymbol{x}, \boldsymbol{u})$ is the observed scalar reward measurement or signal. Since RL is a goal-directed optimal strategy, the scalar reward signals are assumed to be the system performance measurements $\boldsymbol{y}^p$ to be optimised and are dependent on the system states $\boldsymbol{x}$ and VGC parameters $\boldsymbol{u}^{aux}$. Function approximation for the cost is then given as:

$$Q(\boldsymbol{x}_k, \boldsymbol{u}^{aux}(\boldsymbol{x}_k)) = \beta^\top \Psi(\boldsymbol{x}_k, \boldsymbol{u}^{aux}(\boldsymbol{x}_k)) \approx \sum_{n=k}^{N} \gamma^{n-k} \boldsymbol{y}_n^p \tag{3.26}$$

where $\beta \in \mathbb{R}^{p_q}$ are the approximated cost parameters with basis function $\Psi(\boldsymbol{x}, \boldsymbol{u}^{aux}(\boldsymbol{x}))$. Learning is achieved by minimising the TD error and using the recursive Bellman equation as:

$$\begin{aligned} \boldsymbol{e}_k &= \sum_{n=k}^{N} \gamma^{n-k} \boldsymbol{y}_n^p - \beta_k^\top \Psi(\boldsymbol{x}_k, \boldsymbol{u}^{aux}(\boldsymbol{x}_k)) \\ &= \boldsymbol{y}_k^p + \gamma \beta_{k+1}^\top \Psi(\boldsymbol{x}_{k+1}, \boldsymbol{u}^{aux}(\boldsymbol{x}_{k+1})) - \beta_k^\top \Psi(\boldsymbol{x}_k, \boldsymbol{u}^{aux}(\boldsymbol{x}_k)) \end{aligned} \tag{3.27}$$

A BLS or RLS solution is determined for the approximated cost parameters β at each time step using the TD error. This can also be cast in a KF problem with the additional advantage of optimally estimating the time varying parameters under assumed zero-mean parameter variations. Online approximation of the cost parameters using the system performance measurements corresponds to determining the desired operating points for the GTE subject to the gradual engine degradation and variations. As the RL cost explicitly approximates the dependence of the control inputs, the RL ADP strategy therefore belongs to the developed QFA RL method in Section 3.1.2 where the Q-function parameters are adapted to recursively solve the Bellman equation and thereafter used to prescribe a near-optimal policy.

On convergence of the Q-function parameters, an optimisation sub-problem is solved for the VGC parameters and constitutes a policy (set-points) update step. In contrast to the conventional Q-learning policy update, a constrained optimisation problem that guarantees the GTE safety/stability limitations is solved as:

$$u^{*aux}(x_k) = \arg\min_{u^{aux}} Q(x_k, u^{aux}(x_k); \beta)$$
$$\textit{subject to: } g_k^p \leq G_{limits} \tag{3.28}$$

In order for the RL ADP update framework to fit into the overall GTE control architecture presented in Section 3.2.1, a dual-loop control structure is proposed, where the conventional main loop regulates the fuel flow while a RL ADP loop continually updates the VGC set-points in a policy (set-point) optimisation subproblem. Figure 3.6 shows the block diagram of the proposed dual-loop RL framework which is essential to providing a potential route to certification of the overall condition-based control strategy. Transient interaction between the two control loops is minimised by triggering the RL ADP adaptation only at steady-state operating conditions where the most benefits in fuel savings is achievable [11]. Algorithm 3.3 gives the modified QFA template for the RL-ADP CBC framework.

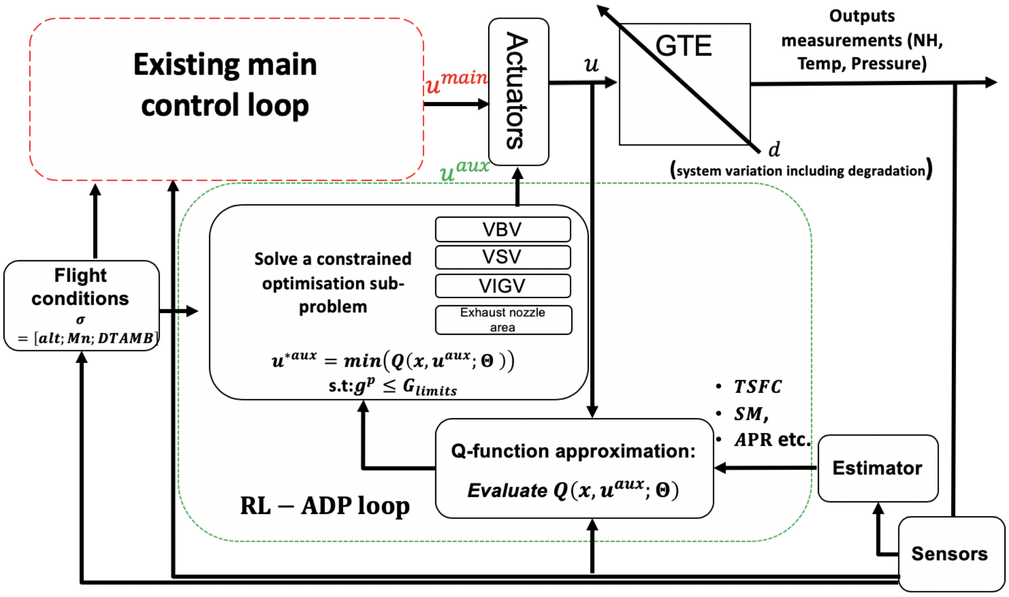

Existing main control loop
u^{main}
Actuators
u
GTE
Outputs measurements (NH, Temp, Pressure)
d
(system variation including degradation)
u^{aux}
Flight conditions
$\sigma = [alt; Mn; DTAMB]$
Solve a constrained optimisation sub-problem
VBV
VSV
VIGV
Exhaust nozzle area
$u^{*aux} = min(Q(x, u^{aux}; \Theta))$
s.t:$g^p \leq G_{limits}$
• TSFC
• SM,
• APR etc.
Q-function approximation:
Evaluate $Q(x, u^{aux}; \Theta)$
Estimator
RL – ADP loop
Sensors

Figure 3.6: Block diagram of the reinforcement learning and approximate dynamic programming (RL-ADP) dual control loop for the gas turbine engine condition-based control. The existing main control loop (in red) guarantees the thrust response control while the RL-ADP control loop (in green) continually adapts the variable geometry components' set-points.

Algorithm 3.3 QFA based RL-ADP framework for the GTE condition-based control

```
1: Initialise the Q-function model parameters $Q(\boldsymbol{x}, \boldsymbol{u}^{aux}(\boldsymbol{x}); \beta_k)$ at $k = 0$ for stabilising VGC gains $\boldsymbol{u}^{aux}(\boldsymbol{x})$
Main control loop: at discrete time steps $k$ during flight:
2: Existing controller computes $\boldsymbol{u}_k^{main} = h(\boldsymbol{y}_k)$ while the VGC set-points i.e. $\boldsymbol{u}^{aux}(\boldsymbol{x}_k)$ are kept fixed till the next update.
RL-ADP loop: triggered at steady-state intervals
        Q-function update step for $j = k$ till parameter convergence:
3: Compute the VGC control inputs with exploration signal $\epsilon$ as $\boldsymbol{u}^{aux}(\boldsymbol{x}_j) + \epsilon$ and obtain measurements for $\boldsymbol{y}_j^p$, $\boldsymbol{g}_j^p$, $\boldsymbol{x}_j$ and $\boldsymbol{x}_{j+1}$.
4: Solve the least squares solution for $\beta_{j+1}$ using the TD error:
        $$\begin{aligned} \boldsymbol{e}_j &= \sum_{n=j}^{N} \gamma^{n-j} \boldsymbol{y}_n^p - Q(\boldsymbol{x}_j, \boldsymbol{u}^{aux}(\boldsymbol{x}_j); \beta_{j+1}) \\ &= \boldsymbol{y}_j^p + \gamma Q(\boldsymbol{x}_{j+1}, \boldsymbol{u}^{aux}(\boldsymbol{x}_{j+1}); \beta_{j+1}) - Q(\boldsymbol{x}_j, \boldsymbol{u}^{aux}(\boldsymbol{x}_j); \beta_{j+1}) \end{aligned}$$
        VGC policy (set-points) update
5: Solve a constrained optimisation sub-problem using the updated steady-state Q-functions as:
        $$\begin{aligned} \boldsymbol{u}^{*aux}(\boldsymbol{x}) &= \arg\min_{u^{aux}} Q(\boldsymbol{x}, \boldsymbol{u}^{aux}(\boldsymbol{x}); \beta_{j+1}) \\ \textit{subject to: } & \boldsymbol{g}^p \leq \boldsymbol{G}_{limits} \end{aligned}$$
6: Repeat steps 2 to 5 till end of flight.
```

3.3.2 Simulation of the RL-ADP condition-based control framework

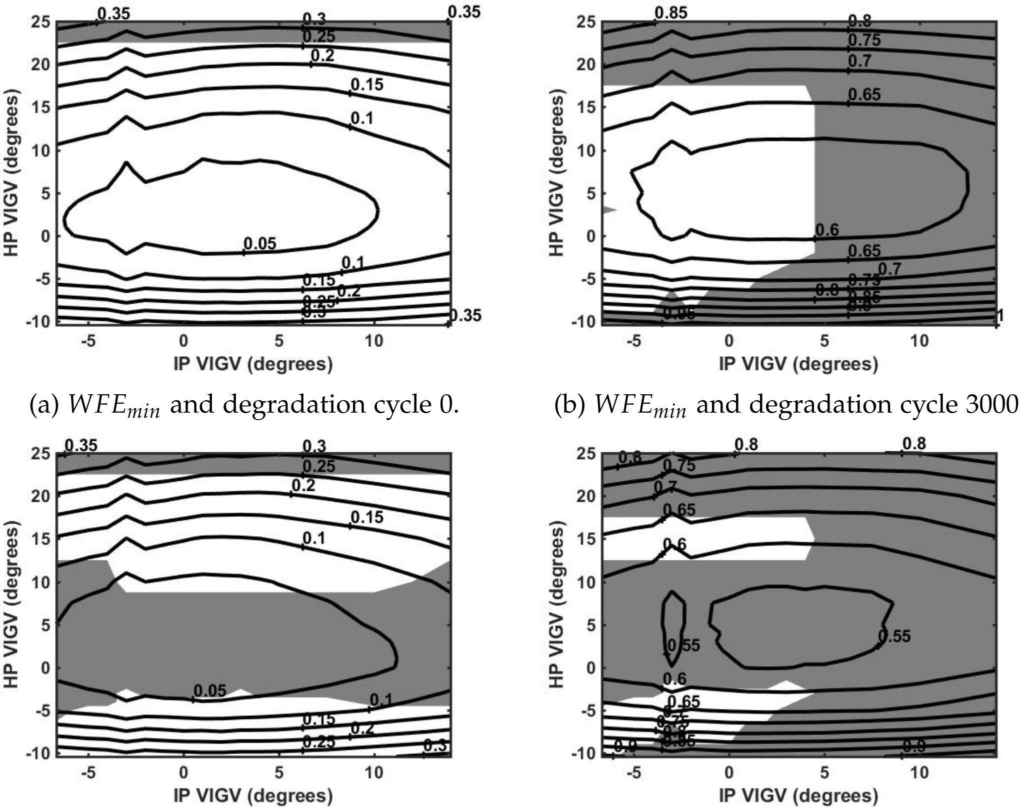
3.3.2.1 Simulation setup

The proposed QFA based RL ADP framework for the CBC problem is demonstrated on representative GTE data sets in MATLAB/SIMULINK environment. The data sets are cruise data from a Roll-Royce engine simulation model for different synthesised degradation conditions between cycle 0 as nominal and cycle 3000 as fully degraded. Inputs to the system are given as the fuel flow, represented by water fuel emulsion (WFE) as the main control variable $\boldsymbol{u}^{main}$ and two sets of VIGV as the auxiliary control parameters $\boldsymbol{u}^{aux}$: the high pressure (HP VIGV) and intermediate pressure (IP VIGV). The fuel flow (WFE) is allowed to vary between ±2.5% of its nominal value at cruise in steps of 0.5% increments/decrements while the IP and HP VIGV vary in steps between high speed stop of -6.67 and -7.5 to low speed stop of 14 to 25 respectively.

System performance measurements $\boldsymbol{y}^{p}$ that reflect changes in the system health due to degradation are given as the thrust specific fuel consumption (TSFC) measurements. Finally, engine limitations $\boldsymbol{g}^{p}$ at cruise for safety and component life are also provided and include surge margin (SM) and various air pressure ratio (APR) limit functions.

Based on Assumption 1, the main control loop computes the required WFE settings $\boldsymbol{u}^{main} = h(\boldsymbol{y})$ and guarantees the thrust response control (i.e. pre-stabilised with min-max limit logic). Similarly, fixed gain schedules for the VIGVs are designed for the worst-case degradation condition. Figure 3.7 shows the offline static variations of the system performance measurements ($\boldsymbol{y}^{p}$ and the limits $\boldsymbol{g}^{p}$) with the control inputs (WFE, IP and HP VIGV) for the different degradation cycles. The worst-case condition from the static variations is at cycle 3000, and the fixed VIGV set-points are scheduled against the steady-state WFE settings ($WFE_{min} \leq WFE \leq WFE_{max}$) at this condition, representative of the conventional design approach. These are designed to satisfy the system constraints with the scheduled gains as shown in Figure 3.8.

Clearly, fixing the VIGV angles for the worst degradation condition will lead to increased fuel consumption at the other conditions. The formulated QFA based RL-ADP scheme is then applied to the system as the engine degrades, using the system performance measurements as reward signals to continually adapt and optimise the VIGV gains at the steady-state WFE settings.



(a) WFE_{min} and degradation cycle 0.

(b) WFE_{min} and degradation cycle 3000.

(c) WFE_{max} and degradation cycle 0.

(d) WFE_{max} and degradation cycle 3000.

Figure 3.7: Contour plots showing the variation of thrust specific fuel consumption (TSFC) with intermediate pressure (IP) and high pressure (HP) variable inlet guide vanes (VIGVs) at two sample steady-state water fuel emulsion (WFE) settings (WFE_{min} and WFE_{max}) at degradation cycles 0 and 3000. The shaded regions indicate infeasible regions of operation due to the safety/reliability limitations.

3.3.2.2 QFA based RL-ADP algorithm implementation

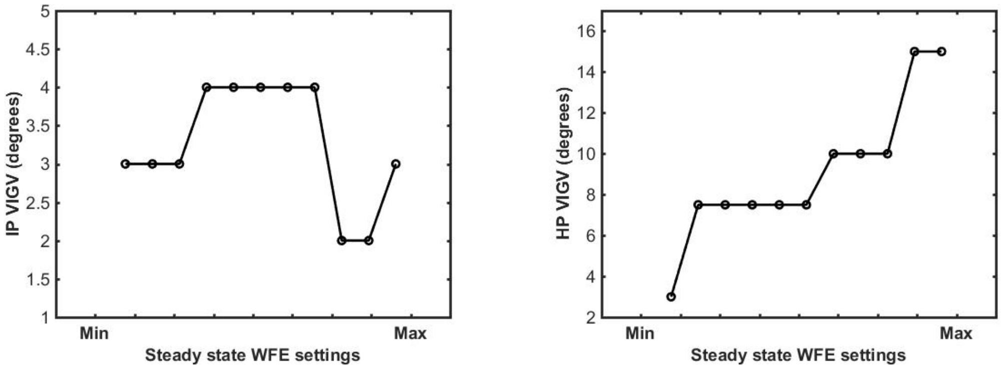
In order to initialise the Q-function model parameters for the system performance measurements, second-order quadratic polynomials were fitted to the offline test data as follows:

$$Q(\boldsymbol{x}, \boldsymbol{u}^{aux}(\boldsymbol{x})) \approx \beta^{\top}\Psi(\boldsymbol{z}) \tag{3.29}$$

where $\beta \in \mathbb{R}^7$ and with

$$\Psi(\boldsymbol{z}) = [WFE^2 \quad IP^2 \quad HP^2 \quad WFE \quad HP \quad IP \quad 1]$$

The polynomial fit was found to give a cross-validated R^2 test statistic of 0.94, negating the need to investigate more complex basis functions. The least squares



(a) IP VIGV schedules.

(b) HP VIGV schedules.

Figure 3.8: Fixed schedules for the intermediate pressure (IP) and high pressure (HP) variable inlet guide vane (VIGV) angles designed for the worst-case system condition that satisfy system constraints and representative of the conventional design approach.

estimation for the Q-function parameter update in Algorithm 3.3 is cast as a KF parameter estimation problem with the parameters modelled as a random-walk signal given as:

$$\begin{aligned} \beta_{k+1} &= \beta_k + \omega_k \\ \omega_k &\sim \mathcal{N}(0, Q_\omega) \end{aligned} \tag{3.30}$$

The TD error from (3.27) therefore becomes:

$$\begin{aligned} \boldsymbol{e}_k &= \boldsymbol{y}_k^p + \beta_{k+1}^\top \big(\gamma \Psi(\boldsymbol{z}_{k+1}) - \Psi(\boldsymbol{z}_k)\big) \\ \boldsymbol{e}_k &\sim \mathcal{N}(0, R_\omega) \end{aligned} \tag{3.31}$$

Q_ω and R_ω are respectively the process and the measurement noise co-variance matrices. Estimation of the parameters using the KF framework operates in a predict-correct cycle as follows:
Predict:

$$\begin{aligned} \beta_{k+1}^- &= \beta_{k+1} \\ \mathcal{P}_{k+1}^- &= \mathcal{P}_k + Q_\omega \end{aligned} \tag{3.32}$$

Correct:

$$
\begin{aligned}
\mathcal{K}_{gain} &= \mathcal{P}_{k+1}^{-}\Psi(z_k)^\top\left(\Psi(z_k)\mathcal{P}_{k+1}^{-}\Psi(z_k)^\top + R_\omega\right)^{-1} \\
\beta_{k+1} &= \beta_{k+1}^{-} + \mathcal{K}_{gain}\boldsymbol{e}_k \\
\mathcal{P}_{k+1} &= \left(I - \mathcal{K}_{gain}\Psi(z_k)\right)\mathcal{P}_{k+1}^{-}
\end{aligned} \tag{3.33}
$$

where β_{k+1}^{-} and $\mathcal{P}_{k+1}^{-}$ are respectively the predicted parameter and error co-variance estimates, $\mathcal{K}_{gain}$ is the Kalman Filter gain, while β_{k+1} and $\mathcal{P}_{k+1}$ are respectively the parameter and error co-variance updates. The matrix Q_ω was selected as $8e^{-8}$ in the simulation for the slowly varying efficiency measurements due to degradation while R_ω was selected as $4e^5$ for the noisy measurements. The KF parameter estimation is run till convergence of the Q-function parameters and constitutes the Q-function update step in Algorithm 3.3. Figure 3.9 shows the Q-function adaptation for both the cost (TSFC) and constraint functions using the Kalman filter framework.

A nonlinear constrained optimisation sub-problem is then solved for the VGC set-points update step described in the algorithm. Due to the computational complexity of gradient based optimisation methods, an adapted direct search method from Venkataraman [117] called 'constrained scan and zoom' was used. This is a derivative free method which executes disciplined search for optimal points around the current iterate using the adapted Q-functions, and systematically proceeds to solutions where the objective function value is reduced and satisfies constraints. The set-points for the VGCs are then updated to the identified optimal points and the process is continued till the end of flight.

Snapshots of the adapted online Q-functions and the identified optimal set-points during engine operation are shown in Figure 3.10, representative of the actual (but assumed unknown) TSFC and constraint variations at steady-state conditions. Figure 3.11a and Figure 3.11b show the adapted VIGV angles using the proposed algorithm as the engine undergoes step changes in degradation from cycle 0 to cycle 3000. Figure 3.12 shows the achieved fuel consumption using the adapted gains as compared with their conventional fixed gains from Figure 3.8. This resulted in fuel savings of about 0.6% at the early degradation stages.

As 1% of cruise TSFC can be worth about $150,000 per year on a four-engined civil aircraft [11], the proposed RL-ADP framework therefore leads to a simple,

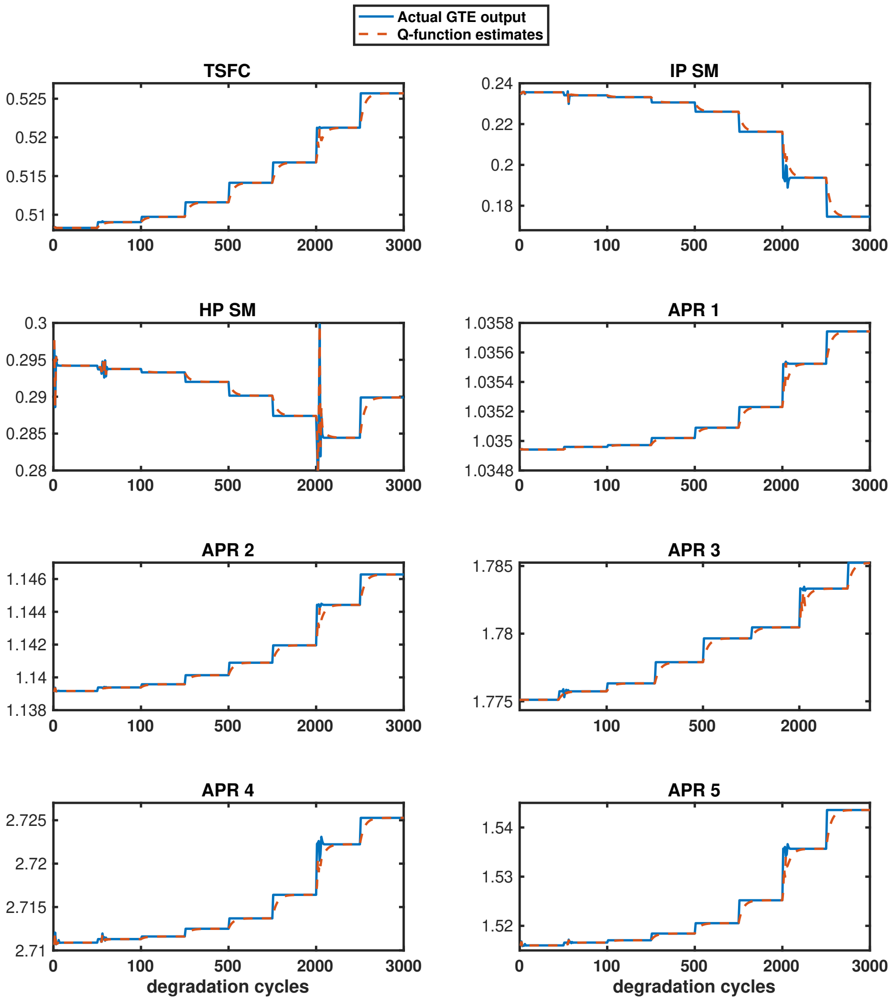
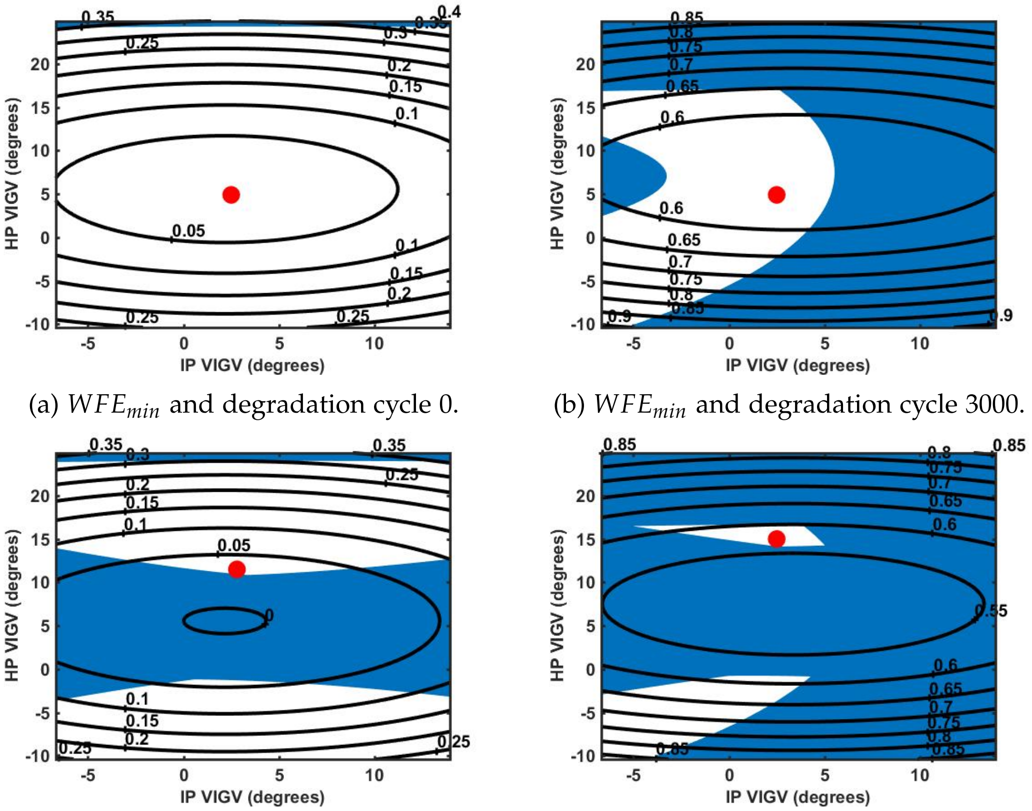


Figure 3.9: Q-function adaptation for both the cost (thrust specific fuel consumption (TSFC)) and constraints (IP and HP surge margin (SM), and air pressure ratio (APR) using the Kalman filter framework.

yet effective and practical means of improving the performance of GTEs across fleets subject to unknown degradation patterns and using only measurements of the desired reward signals.

Summary

Conventional control approaches within the GTE are unable to fully compensate for the gradual engine degradation affecting the system performance. Consequently, the proposed approach as demonstrated in this chapter has shown the



(a) WFE_{min} and degradation cycle 0.

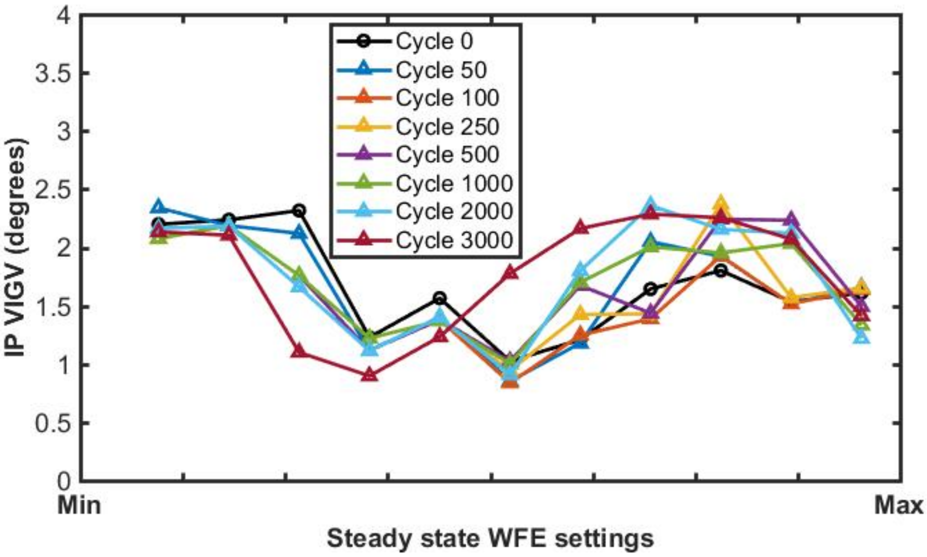
(b) WFE_{min} and degradation cycle 3000.

(c) WFE_{max} and degradation cycle 0.

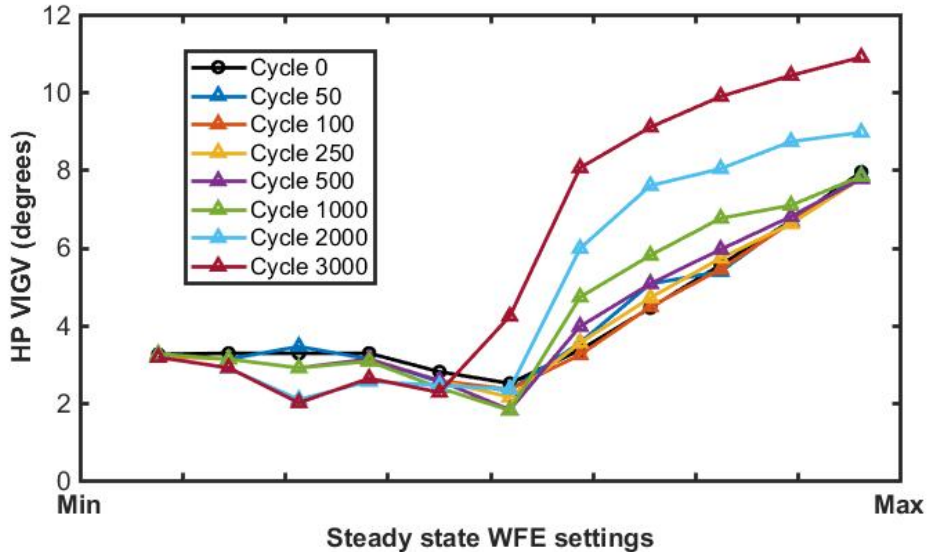
(d) WFE_{max} and degradation cycle 3000.

Figure 3.10: Adapted online Q-function of the system performance measurements showing the variations with intermediate pressure (IP) and high pressure (HP) variable inlet guide vanes (VIGVs) at two sample steady-state water fuel emulsion (WFE) settings (WFE_{min} and WFE_{max}) at degradation cycles 0 and 3000. The shaded regions indicate infeasible regions of operation due to the safety/reliability limitations, while the red dots represent sample identified optimal points using the proposed Q-function approximation based condition-based control framework.

suitability of a RL framework for the condition-based control problem of GTEs in extracting improved performance as the engines degrade over time. A proposed dual-loop control architecture which is essential to providing a potential route to certification for the overall framework integrates the RL adaptations into the existing controller structure. Simulation results on representative engine data sets delivered improved fuel consumption to the GTE as compared to the conventional fixed gain scheduling by adapting the controller set-points to through-life degradation and variations.



(a) IP VIGV schedules.



(b) HP VIGV schedules.

Figure 3.11: Adapted schedules for intermediate pressure (IP) and high pressure (HP) variable inlet guide vane (VIGV) angles through degradation cycles 0 to 3000 using the proposed Q-function approximation based condition-based control framework.

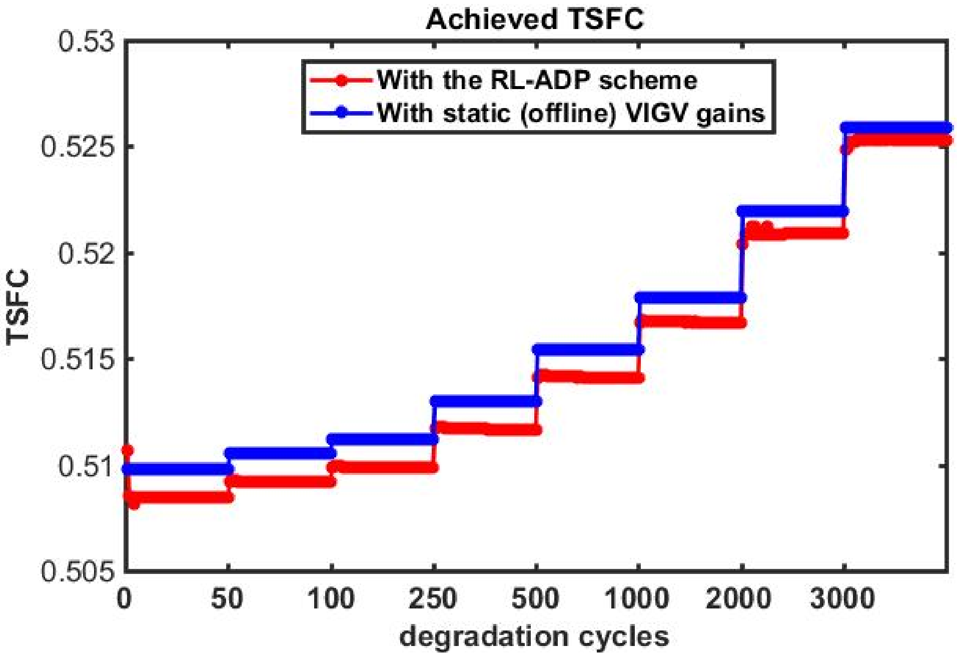


Figure 3.12: Achieved thrust specific fuel consumption (TSFC) using the adapted variable inlet guide vanes′ angles from the proposed Q-function approximation based condition-based control framework as compared with their conventional fixed gain scheduling.

Chapter 4

Reinforcement learning for optimal tracking control - novel condition-based approach

The conventional closed-form solution to the optimal control problem using optimal control theory is only available under the assumption that there are known system dynamics/models described as differential equations. Without such models, RL as a candidate technique has been successfully applied to iteratively solve the optimal control problem for unknown or time-varying systems. The previous chapter has considered the development of RL control frameworks for varying dynamical systems and presented novel approaches for the condition-based control (CBC) of gas turbine engines (GTEs). The developed frameworks provide techniques by which to continually adapt the open-loop part of the GTE control architecture to optimal values subject to gradual system degradation. This chapter extends the RL control frameworks to the closed-loop control part of the GTE control architecture which is responsible for providing desired reference tracking, and hence to the general class of tracking controller applications.

For the optimal tracking control problem, existing RL techniques in the literature either assume the use of a predetermined feedforward input for the tracking control, or use restrictive assumptions on the reference model dynamics and discounted tracking costs. Furthermore, by using a discounted tracking cost, zero steady-state error can no longer be guaranteed by the existing RL methods. This chapter therefore presents an online optimal RL tracking control framework for discrete-time (DT) systems that does not impose any restrictive assumptions on the existing methods and equally guarantees zero steady-state tracking error. This is achieved by forming an augmented system consisting of the original system dynamics and the integral of the error between the reference inputs and the tracked outputs for use in the online RL framework. It is further shown that the resulting value function for the DT linear quadratic tracker (LQT) using the augmented formulation with integral control is also quadratic. This enables the development of Bellman equations which use only the system measurements to solve the corresponding DT algebraic Riccati equation (ARE) and obtain the optimal tracking control inputs online. The strategies and results discussed in this chapter are based on the author′s work in Sanusi et al. [118]. A summary of the main contributions presented in this chapter are as follows:

- An online optimal RL tracking control framework that uses an augmented formulation with integral control is proposed. The proposal does not have the limitations of existing tracking RL methods in the literature that assume either the use of a predetermined feedforward control input or impose restrictions on the reference dynamics and the use of discounted cost. Furthermore, by using a discounted cost, zero steady-state error can no longer be guaranteed by the existing RL methods, but this is overcome by the proposed method.

- Two condition-based RL frameworks that use the augmented formulation with integral control are developed for the tracking control of time-varying dynamical systems along with bounds on excitation needed for the convergence of the RL parameter estimates. These approaches integrate the RL adaptations into the existing controller structure and provide a through-life adaptation strategy.

In the following, Section 4.1 introduces the general online optimal tracking control problem for DT systems and provides the conventional model-based and model-free solution approaches. An augmented formulation with integral control for the online optimal tracking problem is introduced in Section 4.2 along with the model-based and model-free RL solution approaches. Lastly, Section 4.3 provides the simulation of the proposed techniques on two representative case studies.

4.1 Optimal tracking control for discrete-time systems

The tracking design problem aims to steer the system output to follow a desired reference trajectory. This has many practical applications such as in aircraft control systems that are designed to follow desired command inputs from the pilot (e.g. throttle position demand, speed reference etc.). For the development of the tracking control problem, consider the control affine-in-input DT system with the following dynamics:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= f(\boldsymbol{x}_k) + g(\boldsymbol{x}_k)\boldsymbol{u}_k \\ \boldsymbol{y}_k &= h(\boldsymbol{x}_k) \end{aligned} \tag{4.1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} \in \mathbb{R}^m$ and $\boldsymbol{y} \in \mathbb{R}^p$ are respectively the system states, inputs and outputs. Similar to the regulation problem, it is assumed that the system is both controllable and observable [4]. An associated finite-horizon performance cost for the tracking control over the time interval $[0, N]$ is given as:

$$J(\boldsymbol{x}_k) = \phi_N + \sum_{n=k}^{N-1} \gamma^{n-k} \mathcal{L}(\boldsymbol{x}_n, \boldsymbol{u}_n) \tag{4.2}$$

where the reward signal is given as the quadratic energy function $\mathcal{L}(\boldsymbol{x}_k, \boldsymbol{u}_k) = (\boldsymbol{y}_k - \boldsymbol{r}_k)^\top Q_T(\boldsymbol{y}_k - \boldsymbol{r}_k) + \boldsymbol{u}_k^\top R\boldsymbol{u}_k$ with $\phi_N = (\boldsymbol{y}_N - \boldsymbol{r}_N)^\top Q_N(\boldsymbol{y}_N - \boldsymbol{r}_N)$, $\boldsymbol{r}$ is the desired reference trajectory and $0 < \gamma \leq 1$ is the discount factor. For the infinite-horizon case, $\phi_N \to 0$ as $N \to \infty$. The aim of the optimal tracking control problem is to determine the control policy $\boldsymbol{u} = \pi(\boldsymbol{x})$ that minimises the tracking cost (4.2) and guarantees the system stability such that the system output tracks the desired reference i.e. $\boldsymbol{y} \to \boldsymbol{r}$. Hence, the control policy for the LQR problem discussed in Section 2.2.1.2 is no longer valid due to the dependence on the external reference trajectory and warrants new solution strategies.

4.1.1 Model-based linear quadratic tracker

In the linear case, consider the system of (4.1) modelled by the LTI system given as:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A\boldsymbol{x}_k + B\boldsymbol{u}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k \end{aligned} \tag{4.3}$$

A conventional solution to the LQT problem using the calculus of variations is first considered. For this, a corresponding Hamiltonian is defined as:

$$H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1}) = \mathcal{L}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{\lambda}_{k+1}^\top \boldsymbol{x}_{k+1} \tag{4.4}$$

where $\boldsymbol{\lambda}$ is the Lagrange multiplier which is used to adjoin the state equation to the performance cost. The first order necessary conditions of optimality (NCO) to compute a minimum for the Hamiltonian for the un-discounted case i.e. with $\gamma = 1$ are:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \frac{\partial H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1})}{\partial \boldsymbol{\lambda}_{k+1}} = A\boldsymbol{x}_k + B\boldsymbol{u}_k && \text{(4.5a)} \\ \boldsymbol{\lambda}_k &= \frac{\partial H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1})}{\partial \boldsymbol{x}_k} = A^\top \boldsymbol{\lambda}_{k+1} + C^\top Q_T C \boldsymbol{x}_k - C^\top Q_T \boldsymbol{r}_k && \text{(4.5b)} \\ 0 &= \frac{\partial H(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\lambda}_{k+1})}{\partial \boldsymbol{u}_k} = B^\top \boldsymbol{\lambda}_{k+1} + R\boldsymbol{u}_k && \text{(4.5c)} \end{aligned}$$

with boundary conditions:

$$\begin{aligned} &\boldsymbol{x}_0 && \text{(4.5d)} \\ \boldsymbol{\lambda}_N &= C^\top Q_N C \boldsymbol{x}_N - C^\top Q_N \boldsymbol{r}_N && \text{(4.5e)} \end{aligned}$$

From (4.5c) the optimal tracking control input is derived as:

$$\boldsymbol{u}_k^* = -R^{-1}B^\top \boldsymbol{\lambda}_{k+1} \tag{4.6}$$

Substituting (4.6) in (4.5a) yields the non-homogeneous Hamiltonian system driven by a forcing external input $-C^\top Q \boldsymbol{r}_k$ as:

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{\lambda}_k \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B^\top \\ C^\top Q_T C & A^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{\lambda}_{k+1} \end{bmatrix} + \begin{bmatrix} 0 \\ -C^\top Q_T \end{bmatrix} \boldsymbol{r}_k \tag{4.7}$$

The top and bottom parts of (4.7) give the state and co-state equations respectively. From the boundary conditions, the following linear relationship is assumed:

$$\boldsymbol{\lambda}_k = S_k \boldsymbol{x}_k - v_k \quad \forall k \leq N \tag{4.8}$$

where $S_k \in \mathbb{R}^{n \times n}$ and $v_k \in \mathbb{R}^n$ are respectively some yet to be determined intermediate matrix and vector sequences.

Substituting for (4.8) in the state equation of the non-homogeneous Hamiltonian system gives:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A\boldsymbol{x}_k - BR^{-1}B^\top S_{k+1}\boldsymbol{x}_{k+1} + BR^{-1}B^\top v_{k+1} \\ &= (I + BR^{-1}B^\top S_{k+1})^{-1}(A\boldsymbol{x}_k + BR^{-1}B^\top v_{k+1}) && \text{(4.9)} \end{aligned}$$

To compute the intermediate matrix and vector sequences, substitute for both (4.8) and (4.9) in the co-state equation of the non-homogeneous Hamiltonian system to give:

$$S_k \boldsymbol{x}_k - v_k = C^\top Q_T C \boldsymbol{x}_k + A^\top S_{k+1}(I + BR^{-1}B^\top S_{k+1})^{-1}(A\boldsymbol{x}_k + BR^{-1}B^\top v_{k+1}) - \cdots$$
$$\cdots A^\top v_{k+1} - C^\top Q_T \boldsymbol{r}_k \tag{4.10}$$

Simplifying (4.10) gives:

$$[S_k + A^\top S_{k+1}(I + BR^{-1}B^\top S_{k+1})^{-1}A + C^\top Q_T C]\boldsymbol{x}_k + \cdots$$
$$\cdots [v_k + A^\top S_{k+1}(I + BR^{-1}B^\top S_{k+1})^{-1}BR^{-1}B^\top v_{k+1} - A^\top v_{k+1} - C^\top Q_T \boldsymbol{r}_k] = 0 \tag{4.11}$$

Equating the bracketed terms in (4.11) to zero and using the matrix inversion lemma given in (2.37) with $A_m = I$, $B_m = B$, $C_m = B^\top S_{k+1}$ and $D_m = R^{-1}$ gives the Riccati recursion for S_k and v_k as:

$$S_k = A^\top [S_{k+1} - S_{k+1}B(B^\top S_{k+1}B + R)^{-1}B^\top S_{k+1}]A + C^\top Q_T C \tag{4.12}$$
$$v_k = [A^\top - A^\top S_{k+1}B(B^\top S_{k+1}B + R)^{-1}B^\top]v_{k+1} + C^\top Q_T \boldsymbol{r}_k \tag{4.13}$$

with boundary conditions $S_N = C^\top Q_N C$ and $v_N = C^\top Q_N \boldsymbol{r}_N$.

Therefore, the optimal tracking control input (4.6) becomes:

$$\begin{aligned} \boldsymbol{u}_k^* &= -R^{-1}B^\top \boldsymbol{\lambda}_{k+1} \\ &= -R^{-1}B^\top S_{k+1}(A\boldsymbol{x}_k + B\boldsymbol{u}_k^*) + R^{-1}B^\top v_{k+1} \end{aligned} \tag{4.14}$$

Pre-multiplying both sides by R and solving for $\boldsymbol{u}_k^*$ gives:

$$\begin{aligned} \boldsymbol{u}_k^* &= (B^\top S_{k+1}B + R)^{-1}B^\top(-S_{k+1}A\boldsymbol{x}_k + v_{k+1}) \\ &= -K_k^x \boldsymbol{x}_k + K_k^v v_{k+1} \end{aligned} \tag{4.15}$$

where $K_k^x = (B^\top S_{k+1}B + R)^{-1}B^\top S_{k+1}A$ and $K_k^v = (B^\top S_{k+1}B + R)^{-1}B^\top$. For the infinite horizon case i.e. as $N \to \infty$, the Riccati recursion (4.12) becomes the DT ARE given as:

$$\begin{aligned} S &= A^\top SA - A^\top SB(B^\top SB + R)^{-1}B^\top SA + C^\top Q_T C \\ &= A^\top S(A - BK^x) + C^\top Q_T C \end{aligned} \tag{4.16}$$

where $K^x = (B^\top SB + R)^{-1}B^\top SA$ and with $S = S^\top > 0$. The optimal control input

(4.15) becomes:

$$\boldsymbol{u}_k^* = -K^x \boldsymbol{x}_k + K^v v_{k+1} \tag{4.17}$$

where $K^v = (B^\top SB + R)^{-1} B^\top$ and with $v_k = (A - BK_x)^\top v_{k+1} + C^\top Q_T \boldsymbol{r}_k$. Sufficient conditions for a solution are that the pair (A, B) and $(A, \sqrt{Q_T} C)$ are respectively stabilisable and observable. It is noted that the given conventional model-based solution to the LQT optimal control problem consists of both a feedback term K^x that stabilises the system and a feedforward term K^v for the reference tracking. Moreover, the given solution is non-causal [75] as it is dependent on a *backwards-in-time* recursion of the vector sequence v_k. A direct implication of this is that the conventional model-based solution to the tracking problem can only be obtained offline and with full knowledge of the system dynamics. A simulation example is presented to illustrate the conventional tracking solution.

Optimal tracking control example using the conventional model-based approach

To demonstrate the conventional model-based approach, consider the tracking control problem for the 2-state linear system of Equation (3.16) with sampling time $t_s = 0.03s$ and output dynamics:

$$\boldsymbol{y}_k = \underbrace{\begin{bmatrix} 1 & 1 \end{bmatrix}}_{C} \boldsymbol{x}_k \tag{4.18}$$

The tracking control problem is to track a step reference signal from any finite initial condition $\boldsymbol{x}_0$. The tracking cost parameters are given as $\gamma = 1$, $Q_T = Q_N = 2$ and $R = 1$. Figure 4.1 shows the DT evolution and convergence of the Riccati matrix $S^* = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} = \begin{bmatrix} 17.4793 & 23.3872 \\ 23.3872 & 34.5027 \end{bmatrix}$ using the system dynamics, and the corresponding convergence of both the feedforward gain $K^v = \begin{bmatrix} 0.0473 & 0.0376 \end{bmatrix}$ and feedback gain $K^x = \begin{bmatrix} 1.8195 & 2.6378 \end{bmatrix}$ of the optimal tracking input (4.17). Figure 4.2 shows the performance of the tracking controller after convergence of the controller parameters to the optimal values using the offline backwards-in-time recursion.

Extension of the conventional model-based solution to a model-free approach to enable online adaptations is impossible as a result of its non-causal strategy. Furthermore, the approach does not guarantee zero steady-state tracking error. Causal solution strategies that enable model-free approaches have however been proposed in the literature and will now be discussed.

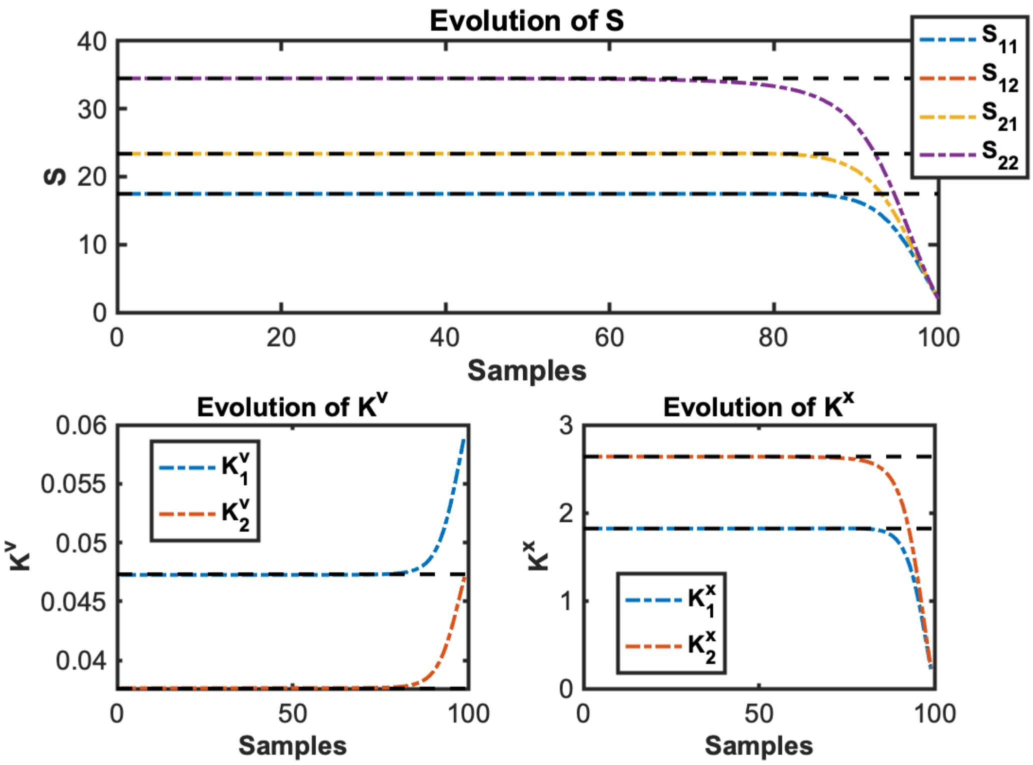


Figure 4.1: Backwards-in-time evolution and convergence of the Riccati matrix S and the corresponding feedforward gain K^v and feedback gain K^x of the tracking control input to the optimal values (in black dashed lines) using the baseline model-based approach.

4.1.2 Model-free linear quadratic tracker

Existing causal solution strategies to the online tracking control problem that enable model-free approaches can be categorised into two as follows:

4.1.2.1 Strategies using dynamics inversion

These methods [77], [78], [79], enable the simultaneous online computation of both the feedforward and feedback terms of the tracking control input. This approach assumes that the desired reference dynamics is governed by:

$$\boldsymbol{r}_{k+1} = f(\boldsymbol{r}_k) + g(\boldsymbol{r}_k)\boldsymbol{u}_{d,k} \tag{4.19}$$

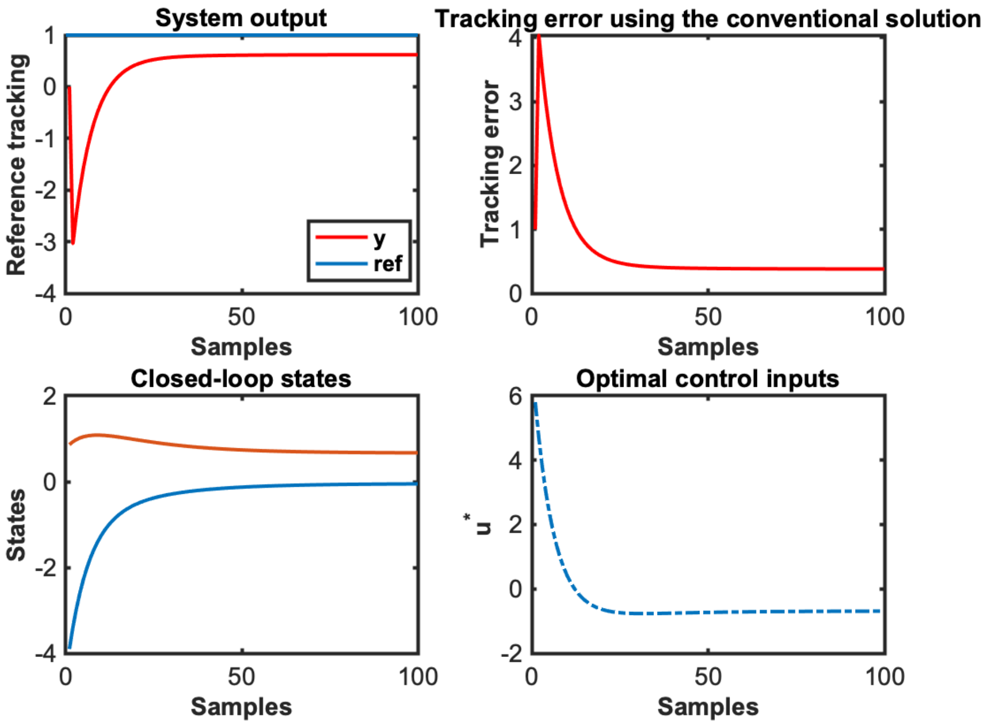


Figure 4.2: Performance of the tracking control input to a step input using the offline computed gains from the baseline model-based approach.

where $\boldsymbol{u}_{d,k} = g(\boldsymbol{r}_k)^{-1}\big(\boldsymbol{r}_{k+1} - f(\boldsymbol{r}_k)\big)$ is the feedforward tracking control input. The dynamics of the tracking error $\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{r}_k$ is given as:

$$\begin{aligned} \boldsymbol{e}_{k+1} &= \boldsymbol{x}_{k+1} - \boldsymbol{r}_{k+1} \\ &= f(\boldsymbol{e}_k + \boldsymbol{r}_k) + g(\boldsymbol{e}_k + \boldsymbol{r}_k)\boldsymbol{u}_k - \boldsymbol{r}_{k+1} \end{aligned} \tag{4.20}$$

where $\boldsymbol{u}_k = \boldsymbol{u}_{e,k} + \boldsymbol{u}_{d,k}$. Substituting for $\boldsymbol{u}_{d,k}$ and simplifying yields:

$$\begin{aligned} \boldsymbol{e}_{k+1} &= f(\boldsymbol{e}_k + \boldsymbol{r}_k) + g(\boldsymbol{e}_k + \boldsymbol{r}_k)g(\boldsymbol{r}_k)^{-1}\big(\boldsymbol{r}_{k+1} - f(\boldsymbol{r}_k)\big) - \boldsymbol{r}_{k+1} + g(\boldsymbol{e}_k + \boldsymbol{r}_k)\boldsymbol{u}_{e,k} \\ &= f_{e,k} + g_{e,k}\boldsymbol{u}_{e,k} \end{aligned} \tag{4.21}$$

where $f_{e,k} = f(\boldsymbol{e}_k + \boldsymbol{r}_k) + g(\boldsymbol{e}_k + \boldsymbol{r}_k)g(\boldsymbol{r}_k)^{-1}\big(\boldsymbol{r}_{k+1} - f(\boldsymbol{r}_k)\big) - \boldsymbol{r}_{k+1}$ and $g_{e,k} = g(\boldsymbol{e}_k + \boldsymbol{r}_k)$. A tracking cost function can then be defined using the quadratic energy function:

$$J(\boldsymbol{e}_k) = \sum_{n=k}^{\infty} \gamma^{n-k}\big(\boldsymbol{e}_n^\top Q_e \boldsymbol{e}_n + \boldsymbol{u}_{e,n}^\top R_e \boldsymbol{u}_{e,n}\big) \tag{4.22}$$

with $Q_e \geq 0$ and $R_e > 0$. Equating the derivative of the cost function with respect to the control input to zero and using the Bellman optimality principle yields the optimal tracking control input as:

$$
\begin{aligned}
\frac{\partial J^*(\boldsymbol{e}_k)}{\partial \boldsymbol{u}_{e,k}} &= \frac{\partial\left(\boldsymbol{e}_k^\top Q_e \boldsymbol{e}_k + \boldsymbol{u}_{e,k}^\top R_e \boldsymbol{u}_{e,k}\right)}{\partial \boldsymbol{u}_{e,k}} + \gamma \frac{\partial J^*(\boldsymbol{e}_{k+1})}{\partial \boldsymbol{u}_{e,k}} = 0 \\
&= 2R_e \boldsymbol{u}_{e,k} + \gamma \frac{\partial J^*(\boldsymbol{e}_{k+1})}{\partial \boldsymbol{e}_{k+1}} \cdot \frac{\partial \boldsymbol{e}_{k+1}}{\partial \boldsymbol{u}_{e,k}} = 0 \\
&= 2R_e \boldsymbol{u}_{e,k} + \gamma g_{e,k}^\top \frac{\partial J^*(\boldsymbol{e}_{k+1})}{\partial \boldsymbol{e}_{k+1}} = 0
\end{aligned}
$$

$$
\therefore \boldsymbol{u}_{e,k}^* = -\frac{\gamma}{2} R_e^{-1} g_{e,k}^\top \frac{\partial J^*(\boldsymbol{e}_{k+1})}{\partial \boldsymbol{e}_{k+1}} \tag{4.23}
$$

The overall control input thus consists of both the feedforward and feedback terms given as:

$$
\boldsymbol{u}_k^* = \boldsymbol{u}_{d,k} + \boldsymbol{u}_{e,k}^* \tag{4.24}
$$

Being causal, the strategy can be implemented online to compute $\boldsymbol{u}_{e,k}$ as it eliminates the need to use backwards-in-time recursion associated with the conventional model-based approach. Standard RL approximation methods introduced in Sections 3.1.1 and 3.1.2 can then be used to develop model-free online strategies to cope with varying or unknown system dynamics.

Remarks

- Complete knowledge of the system dynamics is needed to compute the feedforward term of the tracking control input $\boldsymbol{u}_{d,k}$, with a further assumption that the input function $g(\boldsymbol{r})$ is invertible.

- Online implementation of this approach therefore assumes $\boldsymbol{u}_d$ is known a priori, and only the feedback term $\boldsymbol{u}_e$ is computed online. As a result, practical online adaptation strategies to cope with varying systems are limited using this strategy.

- Similar to the conventional model-based approach, this strategy does not guarantee zero steady-state tracking error.

4.1.2.2 Strategies using augmented formulation

In contrast to the dynamics inversion methods, these methods [38], [75], [81], [87], [89], enable the simultaneous online computation of both the feedforward and feedback terms of the tracking control input. For this, the reference dynamics is assumed to be governed by:

$$\boldsymbol{r}_{k+1} = \psi(\boldsymbol{r}_k) \tag{4.25}$$

where $\psi(\boldsymbol{r}_k)$ is some reference generator model with $\psi(0) = 0$. Similar to (4.20), the error dynamics is defined as:

$$\begin{aligned} \boldsymbol{e}_{k+1} &= \boldsymbol{x}_{k+1} - \boldsymbol{r}_{k+1} \\ &= f(\boldsymbol{e}_k + \boldsymbol{r}_k) + g(\boldsymbol{e}_k + \boldsymbol{r}_k)\boldsymbol{u}_k - \psi(\boldsymbol{r}_k) \end{aligned} \tag{4.26}$$

An augmented system is then formulated using both the error and the reference dynamics as follows:

$$\begin{aligned} \boldsymbol{X}^r_{k+1} &= \begin{bmatrix} f(\boldsymbol{e}_k + \boldsymbol{r}_k) - \psi(\boldsymbol{r}_k) \\ \psi(\boldsymbol{r}_k) \end{bmatrix} + \begin{bmatrix} g(\boldsymbol{e}_k + \boldsymbol{r}_k) \\ 0 \end{bmatrix} \boldsymbol{u}_k \\ &= F^r(\boldsymbol{X}^r_k) + G^r(\boldsymbol{X}^r_k)\boldsymbol{u}_k \end{aligned} \tag{4.27}$$

where $\boldsymbol{X}^r_k = \begin{bmatrix} \boldsymbol{e}_k \\ \boldsymbol{r}_k \end{bmatrix}$. Following this, a tracking cost is defined as:

$$J(\boldsymbol{X}^r_k) = \sum_{n=k}^{\infty} \gamma^{n-k} \left(\boldsymbol{X}^{r\top}_n Q_r \boldsymbol{X}^r_n + \boldsymbol{u}^\top_n R \boldsymbol{u}_n \right) \tag{4.28}$$

where $Q_r = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix}$. Similar to (4.23), the optimal tracking control input is obtained by equating the derivative of the cost function with respect to the control input to zero, which gives:

$$\boldsymbol{u}^*_k = -\frac{\gamma}{2} R^{-1} G^{r\top}(\boldsymbol{X}^r_k) \frac{\partial J^*(\boldsymbol{X}^r_{k+1})}{\partial \boldsymbol{X}^r_{k+1}} \tag{4.29}$$

This way, the tracking problem is recast as a regulation problem, the solution of which gives both the feedforward and feedback terms of the control input. Ditto, standard RL approximation methods introduced in Sections 3.1.1 and 3.1.2 can then be used to develop model-free online strategies to cope with varying or unknown system dynamics.

Remarks

- It is assumed that $\psi(\boldsymbol{r}_k) \to 0$ as $k \to \infty$; where this is not the case, a discounted cost function with $0 < \gamma < 1$ must be used to ensure the value of the cost function remains finite [75]. This assumption poses a restriction on the class of reference generator that can be used with the approach.

- By using a discount factor in the cost function, this approach cannot guarantee zero steady-state tracking error as discussed in [75]. This restrictive assumption on the reference dynamics and discounted cost makes the approach less desirable for use in practical tracking applications.

Consequently, existing RL techniques for the online optimal tracking control problem either assume the use of a predetermined feedforward input for the tracking control, or use restrictive assumptions on the reference model dynamics and discounted tracking costs. In the next section, a new augmented formulation for the online optimal tracking control problem that guarantees zero steady-state tracking error without imposing any restrictive assumptions on the reference dynamics or discounted cost function is proposed to overcome the limitations of the existing strategies.

4.2 Augmented formulation for the optimal tracking problem with integral control

In the following, a new augmented formulation for the online optimal tracking problem with integral control is developed. Consider a new state $\dot{\boldsymbol{z}}$ for the system described in (4.1), defined as the integral of the difference between the desired reference and the system output as:

$$\dot{\boldsymbol{z}}(t) = \int \big(\boldsymbol{r}(t) - \boldsymbol{y}(t)\big)dt \tag{4.30}$$

where $\boldsymbol{z} \in \mathbb{R}^p$. Using Euler's approximation, an equivalent discrete-time state with sampling time t_s is given as:

$$\boldsymbol{z}_{k+1} = \boldsymbol{z}_k + t_s\big(\boldsymbol{r}_k - h(\boldsymbol{x}_k)\big) \tag{4.31}$$

An augmented system can then be formulated using the new integral state as follows:

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} f(\boldsymbol{x}_k) \\ z_k - t_s h(\boldsymbol{x}_k) \end{bmatrix} + \begin{bmatrix} g(\boldsymbol{x}_k) \\ 0 \end{bmatrix} \boldsymbol{u}_k + \begin{bmatrix} 0 \\ t_s I \end{bmatrix} \boldsymbol{r}_k \tag{4.32}$$

At steady-state, the augmented system (4.32) becomes:

$$\begin{bmatrix} \boldsymbol{x}_{\infty} \\ z_{\infty} \end{bmatrix} = \begin{bmatrix} f(\boldsymbol{x}_{\infty}) \\ z_{\infty} - t_s h(\boldsymbol{x}_{\infty}) \end{bmatrix} + \begin{bmatrix} g(\boldsymbol{x}_{\infty}) \\ 0 \end{bmatrix} \boldsymbol{u}_{\infty} + \begin{bmatrix} 0 \\ t_s I \end{bmatrix} \boldsymbol{r}_{\infty} \tag{4.33}$$

For a constant reference signal i.e. $\boldsymbol{r}_{\infty} = \boldsymbol{r}_k$, subtracting (4.33) from (4.32) gives:

$$\begin{bmatrix} \boldsymbol{x}_{k+1} - \boldsymbol{x}_{\infty} \\ z_{k+1} - z_{\infty} \end{bmatrix} = \begin{bmatrix} f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{\infty}) \\ z_k - z_{\infty} - t_s\big(h(\boldsymbol{x}_k) - h(\boldsymbol{x}_{\infty})\big) \end{bmatrix} + \begin{bmatrix} g(\boldsymbol{x}_k)\boldsymbol{u}_k - g(\boldsymbol{x}_{\infty})\boldsymbol{u}_{\infty} \\ 0 \end{bmatrix} \tag{4.34}$$

Further simplification of (4.34) becomes:

$$\boldsymbol{X}_{k+1} = F(\boldsymbol{X}_k) + G(\boldsymbol{X}_k)\tilde{\boldsymbol{u}}_k \tag{4.35}$$

with $\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_k - \boldsymbol{x}_{\infty} \\ z_k - z_{\infty} \end{bmatrix} \in \mathbb{R}^{n+p}$, $\tilde{\boldsymbol{u}}_k = (\boldsymbol{u}_k - \boldsymbol{u}_{\infty}) \in \mathbb{R}^m$ and where $F(\boldsymbol{X}_k) = \begin{bmatrix} f(\boldsymbol{x}_k) - f(\boldsymbol{x}_{\infty}) + g(\boldsymbol{x}_k)\boldsymbol{u}_{\infty} - g(\boldsymbol{x}_{\infty})\boldsymbol{u}_{\infty} \\ z_k - z_{\infty} - t_s\big(h(\boldsymbol{x}_x) - h(\boldsymbol{x}_{\infty})\big) \end{bmatrix}$ and $G(\boldsymbol{X}_k) = \begin{bmatrix} g(\boldsymbol{x}_k) \\ 0 \end{bmatrix}$.

The tracking cost is then redefined as:

$$J(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) = \sum_{n=k}^{\infty} \gamma^{n-k}\big(\boldsymbol{X}_n^{\top} Q_1 \boldsymbol{X}_n + \tilde{\boldsymbol{u}}_n^{\top} R \tilde{\boldsymbol{u}}_n\big) \tag{4.36}$$

where $Q_1 \in \mathbb{R}^{(n+p)\times(n+p)}$. This way, the tracking problem is converted to that of regulation such that the control input for a minimum of (4.36) eliminates the steady-steady error by ensuring that $\boldsymbol{x}_k \to \boldsymbol{x}_{\infty}$ and $z_k \to z_{\infty}$ as $\boldsymbol{X}_k \to 0$. Furthermore, as the new augmented system states are not dependent on the reference dynamics, this approach removes the restrictive assumptions of the existing methods.

An equivalent difference equation to (4.36) for a given fixed policy is given by the value function defined as:

$$\begin{aligned} V(\boldsymbol{X}_k) &= \sum_{n=k}^{\infty} \gamma^{n-k} \big(\boldsymbol{X}_n^\top Q_1 \boldsymbol{X}_n + \tilde{\boldsymbol{u}}_n^\top R \tilde{\boldsymbol{u}}_n\big) \\ &= \mathcal{R}_1(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) + \gamma \sum_{n=k+1}^{\infty} \gamma^{n-(k+1)} \mathcal{R}_1(\boldsymbol{X}_n, \tilde{\boldsymbol{u}}_n) \\ \therefore V(\boldsymbol{X}_k) &= \mathcal{R}_1(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) + \gamma V(\boldsymbol{X}_{k+1}) \end{aligned} \tag{4.37}$$

where $\mathcal{R}_1(\boldsymbol{X}, \tilde{\boldsymbol{u}}) = \boldsymbol{X}^\top Q_1 \boldsymbol{X} + \tilde{\boldsymbol{u}}^\top R \tilde{\boldsymbol{u}}$ and $V(0) = 0$. Using the Bellman principle of optimality, the optimum value becomes:

$$V^*(\boldsymbol{X}_k) = \min_{\tilde{\boldsymbol{u}}} \big(\mathcal{R}_1(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) + \gamma V^*(\boldsymbol{X}_{k+1})\big) \tag{4.38}$$

Equation (4.38) gives the DT HJB equation for the augmented tracking formulation with integral control from which the optimal tracking control input is obtained as:

$$\tilde{\boldsymbol{u}}_k^* = \arg\min_{\tilde{\boldsymbol{u}}} \big(\mathcal{R}_1(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) + \gamma V^*(\boldsymbol{X}_{k+1})\big) \tag{4.39}$$

4.2.1 Model-based solution to the augmented tracking formulation with integral control

A model-based control solution to the optimal tracking problem using the augmented formulation with integral control for DT linear systems is first presented to be used in comparison with the model-free RL approaches introduced in later sections. Using the linear DT system dynamics in (4.3), the augmented system of (4.35) becomes:

$$\begin{aligned} \boldsymbol{X}_{k+1} = \begin{bmatrix} \boldsymbol{x}_{k+1} - \boldsymbol{x}_\infty \\ \boldsymbol{z}_{k+1} - \boldsymbol{z}_\infty \end{bmatrix} &= \begin{bmatrix} A & 0 \\ -t_s C & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k - \boldsymbol{x}_\infty \\ \boldsymbol{z}_k - \boldsymbol{z}_\infty \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} (\boldsymbol{u}_k - \boldsymbol{u}_\infty) \\ &= A_1 \boldsymbol{X}_k + B_1 \tilde{\boldsymbol{u}}_k \end{aligned} \tag{4.40}$$

Lemma 1. *(Quadratic Value Function for LQT using augmented formulation with integral control). Given the LQT cost function (4.36) and dynamics (4.40), for any stabilising control law:*

$$\tilde{\boldsymbol{u}}_k = -\begin{bmatrix} K_x & -K_I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k - \boldsymbol{x}_\infty \\ \boldsymbol{z}_k - \boldsymbol{z}_\infty \end{bmatrix} = -K_1 \boldsymbol{X}_k \tag{4.41}$$

where $K_1 \in \mathbb{R}^{m \times (n+p)}$, $K_x \in \mathbb{R}^{m \times n}$ and $K_I \in \mathbb{R}^{m \times p}$; the value function for the augmented formulation with integral control is quadratic for some matrix $P_1 = P_1^\top > 0 \in$

$\mathbb{R}^{(n+p)\times(n+p)}$ *and given as:*

$$V(\boldsymbol{X}_k) = \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_k \tag{4.42}$$

For simplicity of notation in subsequent analysis, $\boldsymbol{x}_\infty$ and $\boldsymbol{z}_\infty$ are dropped in the augmented states.

Proof. Change the lower limit for the summation in (4.37) and substituting for $\tilde{\boldsymbol{u}}_k$ gives:

$$V(\boldsymbol{X}_k) = \sum_{n=0}^{\infty} \gamma^n \left[\boldsymbol{X}_{n+k}^\top Q_1 \boldsymbol{X}_{n+k} + \boldsymbol{X}_{n+k}^\top K_1^\top R K_1 \boldsymbol{X}_{n+k}\right] \tag{4.43}$$

Noting that:

$$\begin{aligned} \boldsymbol{X}_{n+k} &= (A_1 - B_1 K_1)^n \boldsymbol{X}_k \\ &= \left(\begin{bmatrix} A & 0 \\ -t_s C & I \end{bmatrix} - \begin{bmatrix} BK_x & -BK_I \\ 0 & 0 \end{bmatrix} \right)^n \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{z}_k \end{bmatrix} \\ &= M \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{z}_k \end{bmatrix} \end{aligned} \tag{4.44}$$

where $M = \begin{bmatrix} A - BK_x & BK_I \\ -t_s C & I \end{bmatrix}^n = \begin{bmatrix} M_{11} \in \mathbb{R}^{n\times n} & M_{12} \in \mathbb{R}^{n\times p} \\ M_{21} \in \mathbb{R}^{p\times n} & M_{22} \in \mathbb{R}^{p\times p} \end{bmatrix}$ and $Q_1 = \begin{bmatrix} Q_{11} \in \mathbb{R}^{n\times n} & Q_{12} \in \mathbb{R}^{n\times p} \\ Q_{21} \in \mathbb{R}^{p\times n} & Q_{22} \in \mathbb{R}^{p\times p} \end{bmatrix}$

Equation (4.43) becomes:

$$\begin{aligned} V(\boldsymbol{X}_k) = \sum_{n=0}^{\infty} \gamma^n \Bigg[& \begin{bmatrix} M_{11}\boldsymbol{x}_k + M_{12}\boldsymbol{z}_k \\ M_{21}\boldsymbol{x}_k + M_{22}\boldsymbol{z}_k \end{bmatrix}^\top \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \begin{bmatrix} M_{11}\boldsymbol{x}_k + M_{12}\boldsymbol{z}_k \\ M_{21}\boldsymbol{x}_k + M_{22}\boldsymbol{z}_k \end{bmatrix} \\ + & \begin{bmatrix} M_{11}\boldsymbol{x}_k + M_{12}\boldsymbol{z}_k \\ M_{21}\boldsymbol{x}_k + M_{22}\boldsymbol{z}_k \end{bmatrix}^\top \begin{bmatrix} K_x^\top R K_x & -K_x^\top R K_I \\ -K_I^\top R K_x & K_I^\top R K_I \end{bmatrix} \begin{bmatrix} M_{11}\boldsymbol{x}_k + M_{12}\boldsymbol{z}_k \\ M_{21}\boldsymbol{x}_k + M_{22}\boldsymbol{z}_k \end{bmatrix} \Bigg] \end{aligned} \tag{4.45}$$

Therefore,

$$\begin{aligned} V(\boldsymbol{X}_k) &= \boldsymbol{x}_k^\top P_1^{(11)} \boldsymbol{x}_k + \boldsymbol{x}_k^\top P_1^{(12)} \boldsymbol{z}_k + \boldsymbol{z}_k^\top P_1^{(21)} \boldsymbol{x}_k + \boldsymbol{z}_k^\top P_1^{(22)} \boldsymbol{z}_k \\ &= \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_k \end{aligned} \tag{4.46}$$

where $P_1 = \begin{bmatrix} P_1^{(11)} & P_1^{(12)} \\ P_1^{(21)} & P_1^{(22)} \end{bmatrix}$ and

$P_1^{(11)} = \sum_{n=0}^{\infty} \gamma^n [M_{11}^\top Q_{11} M_{11} + M_{12}^\top Q_{12} M_{11} + M_{11}^\top Q_{12} M_{21} + M_{12}^\top Q_{22} M_{21} + M_{11}^\top K_x^\top R K_x M_{11} - M_{21}^\top K_I^\top R K_x M_{11} - M_{11}^\top K_x^\top R K_I M_{12} + M_{21}^\top K_I^\top R K_I M_{12}]$

$P_1^{(12)} = \sum_{n=0}^{\infty} \gamma^n [M_{11}^\top Q_{11} M_{12} + M_{12}^\top Q_{21} M_{12} + M_{11}^\top Q_{12} M_{22} + M_{12}^\top Q_{22} M_{22} + M_{11}^\top K_x^\top R K_x M_{12} - M_{21}^\top K_I^\top R K_x M_{12} - M_{11}^\top K_x^\top R K_I M_{22} + M_{21}^\top K_I^\top R K_I M_{22}]$

$P_1^{(21)} = \sum_{n=0}^{\infty} \gamma^n [M_{12}^\top Q_{11} M_{11} + M_{22}^\top Q_{21} M_{11} + M_{12}^\top Q_{12} M_{21} + M_{22}^\top Q_{22} M_{21} + M_{12}^\top K_x^\top R K_x M_{11} - M_{22}^\top K_I^\top R K_x M_{11} - M_{12}^\top K_x^\top R K_I M_{12} + M_{22}^\top K_I^\top R K_I M_{12}]$

$P_1^{(22)} = \sum_{n=0}^{\infty} \gamma^n [M_{12}^\top Q_{11} M_{12} + M_{22}^\top Q_{21} M_{12} + M_{12}^\top Q_{12} M_{22} + M_{22}^\top Q_{22} M_{22} + M_{12}^\top K_x^\top R K_x M_{12} - M_{22}^\top K_I^\top R K_x M_{12} - M_{12}^\top K_x^\top R K_I M_{22} + M_{22}^\top K_I^\top R K_I M_{22}]$

□

From (4.38), the Bellman equation for the optimal value function is thus given as:

$$V^*(\boldsymbol{X}_k) = \boldsymbol{X}_k^\top P_1^* \boldsymbol{X}_k = \boldsymbol{X}_k^\top Q_1 \boldsymbol{X}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + \gamma \boldsymbol{X}_{k+1}^\top P_1^* \boldsymbol{X}_{k+1} \tag{4.47}$$

and the optimal control input of (4.39) with $\gamma = 1$ becomes:

$$\begin{aligned}\tilde{\boldsymbol{u}}_k &= \arg\min_{\tilde{u}} \left(\boldsymbol{X}_k^\top Q_1 \boldsymbol{X}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + \boldsymbol{X}_{k+1}^\top P_1 \boldsymbol{X}_{k+1} \right) \\ &= -(R + B_1^\top P_1 B_1)^{-1} B_1^\top P_1 A_1 \boldsymbol{X}_k \\ &= -K_1 \boldsymbol{X}_k \end{aligned} \tag{4.48}$$

where $K_1 = \left((R + B_1^\top P_1 B_1)^{-1} B_1^\top P_1 A_1 \right) = \begin{bmatrix} K_x & -K_I \end{bmatrix}$

Equation (4.48) gives the optimal model-based control solution to the augmented formulation for the DT LQT problem consisting of both the integral feedforward K_I and feedback K_x gains. Substituting for $\tilde{\boldsymbol{u}}_k$ in (4.47) and simplifying gives the corresponding ARE for the system as:

$$P_1 = Q_1 + A_1^\top P_1 A_1 - A_1^\top P_1 B_1 (R + B_1^\top P_1 B_1)^{-1} B_1^\top P_1 A_1 \tag{4.49}$$

Lyapunov stability can be shown for the LQT system by using the Lyapunov function:

$$\begin{aligned}\Delta V(\boldsymbol{X}_k) &= V(\boldsymbol{X}_{k+1}) - V(\boldsymbol{X}_k) = \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_{k+1} - \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_k < 0 \\ &= (A_1\boldsymbol{X}_k + B_1\tilde{\boldsymbol{u}}_k)^\top P_1 (A_1\boldsymbol{X}_k + B_1\tilde{\boldsymbol{u}}_k) - \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_k < 0 \\ &= \boldsymbol{X}_k^\top A_1^\top P_1 A_1 \boldsymbol{X}_k + \boldsymbol{X}_k^\top A_1^\top P_1 B_1 \tilde{\boldsymbol{u}}_k + \tilde{\boldsymbol{u}}_k^\top B_1^\top P_1 A_1 \boldsymbol{X}_k + \tilde{\boldsymbol{u}}_k^\top B_1^\top P_1 B_1 \tilde{\boldsymbol{u}}_k - \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_k < 0\end{aligned} \tag{4.50}$$

Substitute for control input (4.48) as:

$$\begin{aligned}\Delta V(\boldsymbol{X}_k) = \boldsymbol{X}_k^\top A_1^\top P_1 A_1 \boldsymbol{X}_k - \boldsymbol{X}_k^\top A_1^\top P_1 B_1 K_1 \boldsymbol{X}_k - \boldsymbol{X}_k^\top K_1^\top B_1^\top P_1 A_1 \boldsymbol{X}_k \\ + \boldsymbol{X}_k^\top K_1^\top B_1^\top P_1 B_1 K_1 \boldsymbol{X}_k - \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_k < 0 \\ = \boldsymbol{X}_k^\top \left[A_1^\top P_1 A_1 - A_1^\top P_1 B_1 K_1 - K_1^\top B_1^\top P_1 A_1 + K_1^\top B_1^\top P_1 B_1 K_1 - P_1\right] \boldsymbol{X}_k < 0\end{aligned} \tag{4.51}$$

Add and subtract $K_1^\top R K_1$, then simplify further to give:

$$\begin{aligned}\Delta V(\boldsymbol{X}_k) = \boldsymbol{X}_k^\top \big[A_1^\top P_1 A_1 - A_1^\top P_1 B_1 K_1 - K_1^\top B_1^\top P_1 A_1 + K_1^\top B_1^\top P_1 B_1 K_1 \\ - P_1 + K_1^\top R K_1 - K_1^\top R K_1\big] \boldsymbol{X}_k < 0 \\ = \boldsymbol{X}_k^\top \big[A_1^\top P_1 A_1 - A_1^\top P_1 B_1 K_1 - K_1^\top B_1^\top P_1 A_1 + K_1^\top (R + B_1^\top P_1 B_1) K_1 \\ - K_1^\top R K_1 - P_1\big] \boldsymbol{X}_k < 0\end{aligned} \tag{4.52}$$

Finally, substitute for $K_1 = (R + B_1^\top P_1 B_1)^{-1} B_1^\top P_1 A_1$ in (4.52):

$$\begin{aligned}\Delta V(\boldsymbol{X}_k) = \boldsymbol{X}_k^\top \big[A_1^\top P_1 A_1 - A_1^\top P_1 B_1 (R + B_1^\top P_1 B_1)^{-1} B_1^\top P_1 A_1 \\ - K_1^\top R K_1 - P_1\big] \boldsymbol{X}_k < 0\end{aligned} \tag{4.53}$$

But the ARE for the LQT system is given in terms of P_1 in (4.49), therefore, Lyapunov stability is guaranteed for the following condition:

$$\Delta V(\boldsymbol{X}_k) = \boldsymbol{X}_k \big[- Q_1 - K_1^\top R K_1\big] \boldsymbol{X}_k < 0 \tag{4.54}$$

if and only if Q_1 and R are positive semi-definite.

Figure 4.3 shows the block diagram of the augmented tracking control framework with integral control consisting of both a feedforward integral gain K_I and feedback gain K_x. The given baseline integral-proportional (I-P) control structure is widely used in practice where the tracking error is fed into the feedforward integral term while the proportional term is implemented in feedback [119], [120].

Therefore, using knowledge of the system dynamics, the above tracking frame-

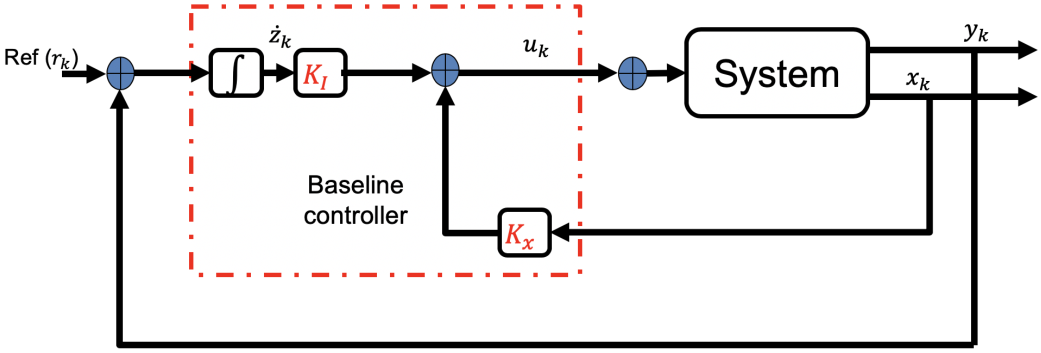


Figure 4.3: Block diagram of an augmented tracking control framework with integral control consisting of both a feedforward integral gain K_I and feedback gain K_x.

work with integral control can be used to achieve optimal tracking control online and does not impose restrictions on the reference model dynamics or use of discounted tracking costs. For systems with unknown or varying dynamics, an approximate online solution to the optimal tracking control framework with integral control is developed in the next section using reinforcement learning. This offers the advantage of not requiring the full knowledge of the system dynamics whilst converging to the optimum values.

4.2.2 Reinforcement learning framework for the optimal tracking control using the augmented formulation with integral control

As discussed in Section 4.1.2, existing approaches for the optimal tracking control problem using RL either assume that the feedforward part of the control is known a priori or make restrictive assumptions on the reference model dynamics and use of discounted tracking costs. These restrictive assumptions are eliminated by using the augmented formulation with integral control as proposed in Section 4.2. Consequently, a novel optimal RL framework is proposed for the LQT problem that converges to the optimum solution for systems with varying or unknown system dynamics using the augmented formulation with integral control. Furthermore, unlike the previously proposed RL tracking approaches [38], [75], [77], [78], [79], [81], [87], [89], the proposed formulation is able to guarantee zero steady-state tracking error and provides adaptation for both the feedforward and feedback controller gains. The framework continually adapts the controller gains to optimum values and provides a through-life adaptation strategy.

Consistent with the techniques discussed in Section 3.2, these approaches are termed condition-based by using the system measurements to optimise to slow and varying changes in the system performance. Using the RL approximation techniques developed in Sections 3.1.1 and 3.1.2, the condition-based tracking control frameworks are enabled by approximating the associated value functions as follows. For the VFA method, the state value function is approximated as:

$$V^{\pi}(\boldsymbol{X}_k) \approx \theta_c^{\top}\Phi_c(\boldsymbol{X}_k) = \sum_{n=k}^{\infty} \gamma^{n-k}\mathcal{R}_1(\boldsymbol{X}_n, \tilde{\boldsymbol{u}}_n) \tag{4.55}$$

where $\Phi_c(\boldsymbol{X})$ is a set of basis function with weights θ_c. Equation (4.55) gives the approximated sum of the discounted reward signal $\mathcal{R}_1(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k)$ starting from $\boldsymbol{X}_k$ under some policy $\pi(\boldsymbol{X})$. Similarly, for the QFA method, the state-action value function is approximated as:

$$Q^{\pi}(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) \approx \beta^{\top}\Psi(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) = \sum_{n=k}^{\infty} \gamma^{n-k}\mathcal{R}_1(\boldsymbol{X}_n, \tilde{\boldsymbol{u}}_n) \tag{4.56}$$

where $\Psi(\boldsymbol{X}, \tilde{\boldsymbol{u}})$ is a set of basis function with weights β. Equation (4.56) gives the approximated sum of discounted reward signal $\mathcal{R}_1(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k)$ starting from state $\boldsymbol{X}_k$ and taking action $\tilde{\boldsymbol{u}}_k$, then following policy $\pi(\boldsymbol{X})$ thereon. Depending on the function that is being approximated, two RL strategies are therefore proposed for the condition-based optimal tracking control.

4.2.2.1 VFA based optimal tracking control

In the VFA RL approximation method, the Bellman equation for the state value function (4.55) becomes:

$$\theta_c^{\top}\Phi_c(\boldsymbol{X}_k) = \boldsymbol{X}_k^{\top} Q_1 \boldsymbol{X}_k + \tilde{\boldsymbol{u}}_k^{\top} R \tilde{\boldsymbol{u}}_k + \gamma\theta_c^{\top}\Phi_c(\boldsymbol{X}_{k+1}) \tag{4.57}$$

This represents the critic network, and the parameters are updated using the TD error from either a VI or PI recursion as given in Section 3.1.1. A second function approximation which serves as the actor network is used to adapt the controller gains and is given as:

$$\tilde{\boldsymbol{u}}_k = \pi(\boldsymbol{X}_k) = \theta_a^{\top}\boldsymbol{X}_k = -\hat{K}_1\boldsymbol{X}_k \tag{4.58}$$

The RL adaptations consist of both a value and policy update steps. For the value update step, the policy is kept fixed while the value function parameters

Algorithm 4.1 VFA based RL tracking algorithm using PI

```
Initialise $V(\boldsymbol{X}) \approx \theta_{c,k}^{\top}\Phi_c(\boldsymbol{X})$ at $k = 0$ for some stabilising policy $\pi(\boldsymbol{X}) = \theta_{a,k}^{\top}\boldsymbol{X}$, and do till convergence:
        **Value function update step**
**for** $j = 0 : N$ **do**
    At $\boldsymbol{X}_j$, compute the control input $\tilde{\boldsymbol{u}}_j$ with exploration signal $\epsilon$ as $\tilde{\boldsymbol{u}}_j = \pi(\boldsymbol{X}_j) + \epsilon$.
    Compute the least squares solution for $\theta_{c,j+1}$ using measurements $\mathcal{R}_1(\boldsymbol{X}_j, \tilde{\boldsymbol{u}}_j)$, $\boldsymbol{X}_j$ and $\boldsymbol{X}_{j+1}$ as:
        $$\theta_{c,j+1}^{\top}\big(\Phi_c(\boldsymbol{X}_j) - \gamma\Phi_c(\boldsymbol{X}_{j+1})\big) = \boldsymbol{X}_j^{\top} Q_1 \boldsymbol{X}_j + \tilde{\boldsymbol{u}}_j^{\top} R \tilde{\boldsymbol{u}}_j$$
    $j = j + 1$.
**end for**
        **Policy update step**
**Require:** Set $\theta_{c,k+1} = \theta_{c,j+1} \,|_{j=N}$
Update the policy parameters using the gradient descent tuning as:
    $$\theta_{a,k}^{i+1} = \theta_{a,k}^{i} - l_a \boldsymbol{X}_i \left( 2R\theta_{a,k}^{i\top}\boldsymbol{X}_i + \gamma B_1^{\top} \frac{\partial \theta_{c,k+1}^{\top}\Phi_c(\boldsymbol{X}_{i+1})}{\partial \boldsymbol{X}_{i+1}} \right)$$
At the end of the gradient tuning, set $\theta_{a,k+1} = \theta_{a,k}^{i+1}$ and update the policy as:
    $$\pi(\boldsymbol{X}) = \theta_{a,k+1}^{\top}\boldsymbol{X} = -K_1\boldsymbol{X}$$
Increment time step $k = k + 1$.
```

are updated using the system measurements at N episodic intervals (i.e. from some initial state $\boldsymbol{X}_0$ to a terminal state $\boldsymbol{X}_N$). After each episode, the controller parameters are adapted from (4.39) using a gradient tuning update as described in Section 3.1.1. This is repeated till convergence of both the critic and actor network parameters. This way, the VFA based RL method solves the online LQT problem of Section 4.1 using the proposed augmented formulation with integral control and without requiring knowledge of the system dynamics. Algorithm 4.1 describes the VFA based RL adaptations for the controller gains using a PI recursion.

The VFA based RL algorithm is not completely model-free as knowledge of the input matrix B_1 is needed in computing the actor network parameter update. Consequently, the approach is limited to systems with variations occurring only in the drift or dynamics matrix A_1.

4.2.2.2 QFA based optimal tracking control

In the QFA RL approximation method, the Bellman equation for the state-action value function (4.56) becomes:

$$\beta^\top \Psi(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) = \boldsymbol{X}_k^\top Q_1 \boldsymbol{X}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + \gamma \beta^\top \Psi(\boldsymbol{X}_{k+1}, \tilde{\boldsymbol{u}}_{k+1}) \tag{4.59}$$

The RL adaptations equally consist of both a Q-function and policy update steps. In contrast to the VFA RL method, the Q-function explicitly approximates the control inputs for each state from which the optimal control input can be obtained via a greedy optimisation. This makes the QFA RL method completely model-free by using only the measurements observed along the system trajectories for the controller updates and is further described in Algorithm 4.2.

The Q-function parameters are updated in each episode whilst keeping the policy fixed and constitutes the Q-function update step. For the policy update, a greedy optimisation is performed after each episode using the adapted Q-function parameters as described in Section 3.1.2 as:

$$\tilde{\boldsymbol{u}}_k = \arg\min_{\tilde{u}} \left(\beta^\top \Psi(\boldsymbol{X}_k, \tilde{\boldsymbol{u}}_k) \right) = \hat{K}_1 \boldsymbol{X}_k \tag{4.60}$$

Remarks for Algorithms 3.1 and 3.2 also apply to Algorithms 4.1 and 4.2 respectively.

The RL control strategies described above solve the online LQT problem without knowledge of the system dynamics or variations. Furthermore, by using the proposed augmented formulation with integral control, the RL frameworks do not require any predetermined feedforward tracking control input or restrictive assumptions on the reference generator dynamics and use of discounted tracking costs. In the development of the condition-based RL tracking control framework, the following considerations as highlighted in Section 2.4 are noted:

- The reward signal for the tracking problem is selected as the quadratic energy function $\mathcal{R}_1(\boldsymbol{X}, \tilde{\boldsymbol{u}}) = \boldsymbol{X}^\top Q_1 \boldsymbol{X} + \tilde{\boldsymbol{u}}^\top R \tilde{\boldsymbol{u}}$. However, to be condition-based, variations or decline in system performance can be detected by measuring standard step response parameters like the percentage overshoot (P.O.), rise time, etc. This can then be used as an enable signal to initiate the RL framework to learn new optimal tracking parameters.
- The condition-based RL tracking control assumes an unconstrained formulation for adaptations of its controller gains. For the GTE control architecture,

Algorithm 4.2 QFA based RL tracking algorithm using PI

```
Initialise $Q(\boldsymbol{X}, \tilde{\boldsymbol{u}}) \approx \beta_k^\top \Psi(\boldsymbol{X}, \tilde{\boldsymbol{u}})$ at $k = 0$ for some stabilising policy $\pi(\boldsymbol{X}) = \arg\min_{\tilde{u}} \left(\beta_k^\top \Psi(\boldsymbol{X}, \tilde{\boldsymbol{u}})\right)$, and do till convergence:
        Q-function update step
for $j = 0 : N$ do
    At $\boldsymbol{X}_j$, compute the control input $\tilde{\boldsymbol{u}}_j$ with exploration signal $\epsilon$ as $\tilde{\boldsymbol{u}}_j = \pi(\boldsymbol{X}_j) + \epsilon$.
    Compute the least squares solution for $\beta_{j+1}$ using measurements $\mathcal{R}_1(\boldsymbol{X}_j, \tilde{\boldsymbol{u}}_j)$, $\tilde{\boldsymbol{u}}_j$, $\boldsymbol{X}_j$ and $\boldsymbol{X}_{j+1}$ as:
        $$\beta_{j+1}^\top \big(\Psi(\boldsymbol{X}_j, \tilde{\boldsymbol{u}}_j) - \gamma \Psi(\boldsymbol{X}_{j+1}, \tilde{\boldsymbol{u}}_{j+1})\big) = \boldsymbol{X}_j^\top Q_1 \boldsymbol{X}_j + \tilde{\boldsymbol{u}}_j^\top R \tilde{\boldsymbol{u}}_j$$
    where $\tilde{\boldsymbol{u}}_{j+1} = \pi(\boldsymbol{X}_{j+1})$
    $j = j + 1$.
end for
        Policy update step
Require: Set $\beta_{k+1} = \beta_{j+1} \mid_{j=N}$
Update the policy parameters using a greedy optimisation as:
        $$\pi(\boldsymbol{X}) = \arg\min_{\tilde{u}} \left(\beta_{k+1}^\top \Psi(\boldsymbol{X}, \tilde{\boldsymbol{u}})\right) = -K_1 \boldsymbol{X}$$
Increment time step $k = k + 1$.
```

the RL adaptations can occur prior to the limiter logic used to maintain operational safety/reliability and thus consistent with other gain modifier techniques reported in [113].

- A baseline integral-proportional (I-P) control architecture of Figure 4.3 where the tracking error is fed into a feedforward integral term while the proportional term is implemented in feedback is assumed. The condition-based framework therefore aims to provide adaptations for the controller gains using RL to optimum values subject to the gradual system variations.

The control strategy described above is represented schematically in Figure 4.4 where the RL block represents either the VFA or QFA algorithm that continually uses the observed system measurements to adapt the tracking controller gains to optimum values subject to varying or unknown system dynamics.

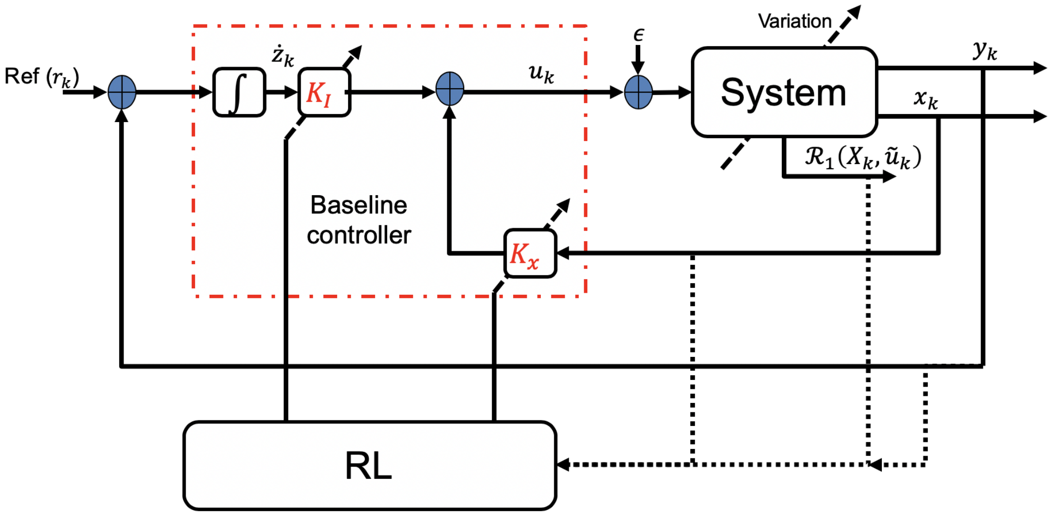


Figure 4.4: Schematic of a condition-based reinforcement learning (RL) framework for the optimal tracking control using the augmented formulation with integral control. The RL block represents either the value function approximation (VFA) or Q-function approximation (QFA) algorithm that continually uses the observed system measurements to adapt the tracking controller gains to optimum values subject to varying or unknown system dynamics.

4.3 Simulation case studies for the condition-based RL tracking control framework

The condition-based RL tracking framework is demonstrated on two representative case studies. The first is a system with an initially unstable and unknown dynamics that shows convergence of the proposed RL tracking methods to the optimal tracking controller gains. The second case study addresses the optimal tracking control problem in a buck power converter system which is subject to uncertain or varying component tolerances under different operating conditions.

4.3.1 Case study 1

The first case study uses the system described in Equation (3.17) with:

$$A_{(1)} = \begin{bmatrix} 0.9724 & 0.0607 \\ 0.0668 & 1.0544 \end{bmatrix}; \quad B = \begin{bmatrix} 0.0605 \\ 0.0482 \end{bmatrix}$$

$$y_k = \underbrace{\begin{bmatrix} 1 & 1 \end{bmatrix}}_{C} x_k \tag{4.61}$$

for which a baseline model-based optimal tracking control solution has been derived in Section 4.1.1. The tracking control problem is to track a time-varying step reference input from any finite initial condition $\boldsymbol{x}_0$ representative of pilot reference commands in a GTE or precision tracking applications. Tracking cost parameters in (4.36) for the augmented formulation are considered as $Q_1 = 2 \times \mathcal{I}(3)$, $R = 0.05$ and $\gamma = 1$.

4.3.1.1 Existing online solution approach with the use of discounted cost

The existing online solution to the optimal tracking control problem as discussed in Section 4.1.2 requires knowledge of the reference dynamics and the use of discounted tracking cost. For the given tracking problem, consider the reference dynamics of (4.25) to be given by the linear difference equation:

$$\boldsymbol{r}_{k+1} = F\boldsymbol{r}_k \tag{4.62}$$

where $F = 1$. An augmented system with the reference dynamics can then be formulated according to (4.27). Furthermore, as a result of using a reference dynamics that does not tend to zero, a discounted cost must be used. Comparison of the performance of this approach using different discount factors to the proposed augmented formulation with integral control is shown in Figure 4.5.

As could be observed in the simulation result, a discount factor of $\gamma = 0.8$ had a slower response but a reduced steady-state error while a discount factor of $\gamma = 0.7$ had a faster response but larger steady-state error. Existing online tracking approaches with the use of a discount factor are therefore not only restrictive to the type of reference dynamics that can be used, but also cannot guarantee zero steady-state tracking error. In the following, the proposed online solution approaches that do not require knowledge of the reference dynamics or the use of discounted cost will now be presented.

4.3.1.2 Model-based solutions using the proposed augmented formulation with integral control

Baseline solution for the augmented formulation with integral control using the system models is first presented. An augmented system with integral control is formed according to (4.40) as:

$$X_{k+1} = \underbrace{\begin{bmatrix} 0.9724 & 0.0607 & 0 \\ 0.0668 & 1.0544 & 0 \\ -0.03 & -0.03 & 1 \end{bmatrix}}_{A_{1(1)}} X_k + \underbrace{\begin{bmatrix} 0.0605 \\ 0.0482 \\ 0 \end{bmatrix}}_{B_1} \tilde{u}_k \tag{4.63}$$

Using the given system models $(A_{1(1)}, B_1)$, the optimal solution to the corresponding ARE (4.49) is given as:

$$P^*_{1(1)} = \begin{bmatrix} 10.1584 & -6.9476 & -8.8170 \\ -6.9476 & 18.5835 & 4.5047 \\ -8.8170 & 4.5047 & 68.4224 \end{bmatrix} \tag{4.64}$$

with the optimal tracking controller gains as:

$$K^*_{1(1)} = \left((R + B_1^\top P_1 B_1)^{-1} B_1^\top P_1 A_{1(1)}\right) = \begin{bmatrix} K_x & -K_I \end{bmatrix} = \begin{bmatrix} 3.6277 & 5.7644 & -4.6873 \end{bmatrix} \tag{4.65}$$

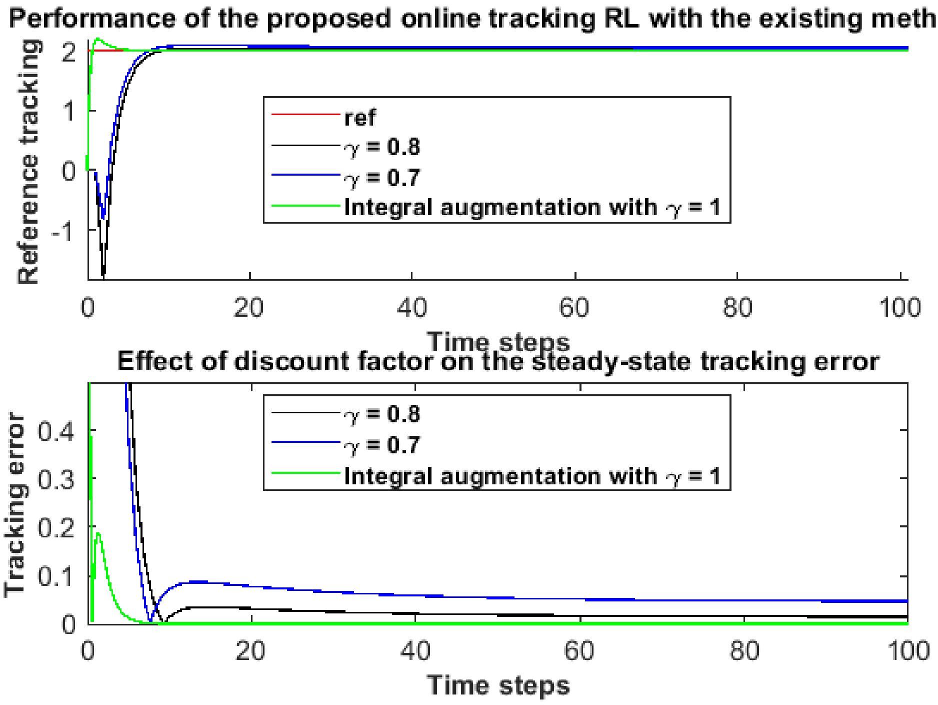


Figure 4.5: Comparison of the existing online tracking methods with the use of discount factors with the proposed integral augmentation approach.

However, in practice the system dynamics may be unknown or time varying therefore motivating the use of the proposed model-free online RL tracking methods.

4.3.1.3 Model-free RL tracking solutions

The proposed model-free RL tracking approaches can be used to obtain the optimal tracking controller gains online subject to the unknown or varying system dynamics.

4.3.1.3.1 VFA based RL adaptation From Lemma 1, the value function for the augmented formulation with integral control is quadratic, thus the critic network in the VFA method for the given 2-state system in Algorithm 4.1 is approximated by the quadratic function:

$$V(\boldsymbol{X}) \approx \theta_c^\top \Phi(\boldsymbol{X}) = \theta_c^\top \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 z \\ x_2^2 \\ x_2 z \\ z^2 \end{bmatrix} \tag{4.66}$$

where $\theta_c \in \mathbb{R}^6$. A linear function approximates the controller gains in the actor network as:

$$\pi(\boldsymbol{X}) \approx \theta_a^\top \boldsymbol{X} = \theta_a^\top \begin{bmatrix} x_1 & x_2 & z \end{bmatrix} \tag{4.67}$$

with $\theta_a \in \mathbb{R}^3$. From Algorithm 4.1, an initially sup-optimal but stabilising policy is arbitrarily selected as:

$$\pi(\boldsymbol{X}) = \underbrace{\begin{bmatrix} 0.4112 & -2.0412 & 2.5011 \end{bmatrix}}_{-\hat{K}_{1(0)}} \tag{4.68}$$

The rest of Algorithm 4.1 is then run online till convergence of the tracking controller parameters using only the observed system measurements. The VFA parameters converged to the following optimal values:

$$\theta^*_{c(1)} = \begin{bmatrix} 9.9155 \\ -14.9830 \\ -16.7696 \\ 18.0048 \\ 10.1510 \\ 68.8826 \end{bmatrix}^\top \tag{4.69}$$

with:

$$\begin{bmatrix} P_1^{11} & P_1^{12} & P_1^{13} \\ P_1^{21} & P_1^{22} & P_1^{23} \\ P_1^{31} & P_1^{32} & P_1^{33} \end{bmatrix} = \begin{bmatrix} \theta_c^{(1)} & 0.5\theta_c^{(2)} & 0.5\theta_c^{(3)} \\ 0.5\theta_c^{(2)} & \theta_c^{(4)} & 0.5\theta_c^{(5)} \\ 0.5\theta_c^{(3)} & 0.5\theta_c^{(5)} & \theta_c^{(6)} \end{bmatrix} \tag{4.70}$$

and $\theta^*_{a(1)} = [-3.4202; -5.5650; 4.6468] = -\hat{K}^*_{1(1)}$.

To demonstrate the continual adaptation of the tracking controller gains to optimal values using the proposed condition-based RL tracking control framework, the system drift matrix A is changed instantaneously during simulation to:

$$A_{(2)} = \begin{bmatrix} 0.8706 & 0.1672 \\ -0.0395 & 1.1654 \end{bmatrix} \tag{4.71}$$

with a new baseline model-based solution from using the system $A_{(2)}$ matrix given as:

$$P^*_{1(2)} = \begin{bmatrix} 23.3462 & -27.4260 & -34.6172 \\ -27.4260 & 49.7383 & 38.9057 \\ -34.6172 & 38.9057 & 127.0261 \end{bmatrix}$$

$$K^*_{1(2)} = \begin{bmatrix} 1.2757 & 8.8839 & -4.6907 \end{bmatrix} \tag{4.72}$$

Following this system variation, the tracking controller gains are no longer optimal resulting in a decline in the system performance. This can be detected in practice by using a threshold on standard step response parameters like percentage overshoot (P.O.), rise time, etc. and used as an enable signal to re-initiate the

RL learning process. The VFA parameters after the system variation converged to:

$$\theta^*_{c(2)} = \begin{bmatrix} 23.3958 \\ -56.2854 \\ -69.6663 \\ 49.9611 \\ 78.9013 \\ 127.1470 \end{bmatrix}^\top \tag{4.73}$$

and $\theta^*_{a(2)} = [-0.9668; -8.7688; -4.6794] = -\hat{K}^*_{1(2)}$. Figure 4.6 shows the parameter convergence using the VFA based RL adaptation to the optimal but assumed unknown values before and after the system variation.

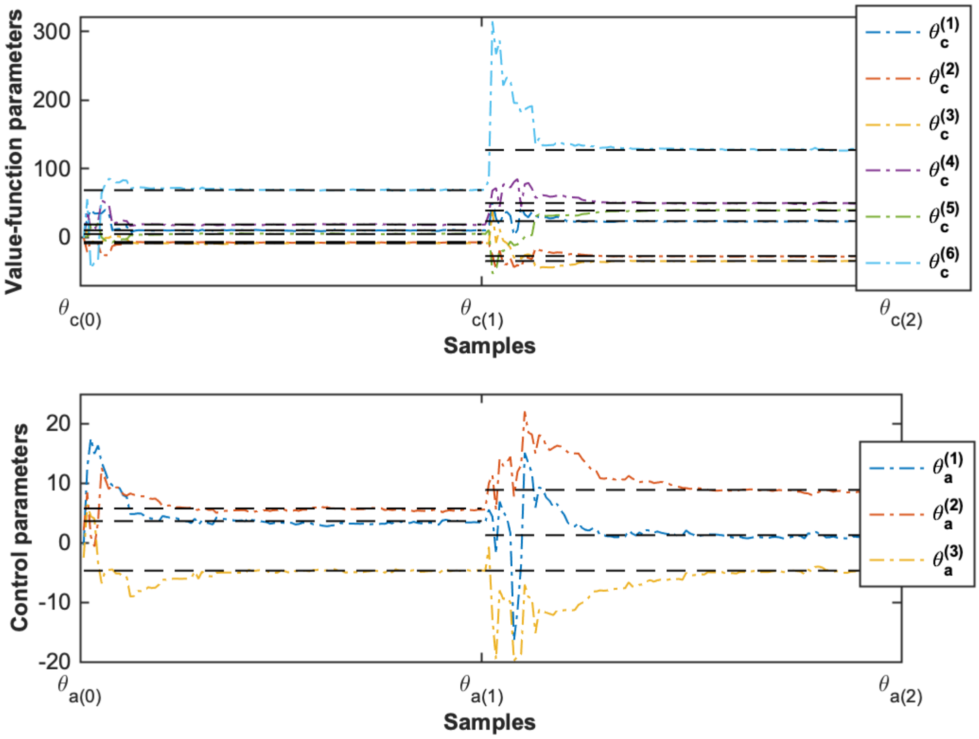


Figure 4.6: Online adaptation and convergence of both the value function and controller parameters to the optimal values (in black dashed lines) using Algorithm 4.1. $\theta_{a,c(0)}$ are the initial sub-optimal controller parameters while $\theta_{a,c(1)}$ and $\theta_{a,c(2)}$ are respectively the identified optimal controller parameters before and after the system variation.

Figure 4.7 shows the overall system response to time-varying step reference inputs at the various stages of the RL adaptation. The region with $\theta_{a,c(0)}$ in the figure corresponds to the system response using the initial sub-optimal controller gains, while the region with $\theta_{a,c(1)}$ shows the system response after convergence to the optimal controller values from the RL adaptation. After the system variation and keeping the controller values fixed, the region with $\theta_{a,c(1)}$ *with variation* shows the decline in system performance following which the RL adaptation is re-enabled. The new system performance after convergence to the new optimal control gains is then shown in the region with $\theta_{a,c(2)}$.

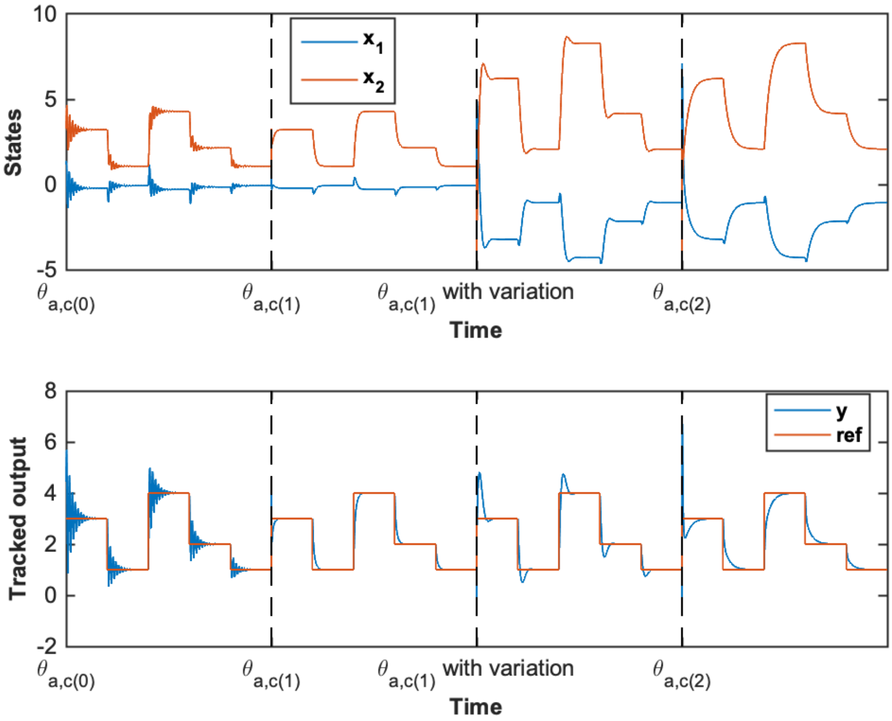


Figure 4.7: System response showing the system states and tracked output at the various stages of the reinforcement learning adaptations. Region with $\theta_{a,c(0)}$ shows the response using the initial sub-optimal controller gains, while region with $\theta_{a,c(1)}$ shows the response from the adapted controller gains to the optimal values using the proposed Algorithms. Region with $\theta_{a,c(1)}$ *with variation* shows the decline in the system performance following variations in the system dynamics whilst keeping the controller values fixed, while region with $\theta_{a,c(2)}$ onwards shows the response after adaptation to the new optimal control gains.

4.3.1.3.2 QFA based RL adaptation The QFA provides a completely model free approach to the LQT problem and similar to the VFA, the Q-functions from Algorithm 4.2 are approximated for the 2-state system using a quadratic basis set as:

$$Q(\boldsymbol{X},\tilde{\boldsymbol{u}}) \approx \beta^\top \Psi(\boldsymbol{X},\tilde{\boldsymbol{u}}) = \beta^\top \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_1z \\ x_1\tilde{u} \\ x_2^2 \\ x_2z \\ x_2\tilde{u} \\ z^2 \\ z\tilde{u} \\ \tilde{u}^2 \end{bmatrix} \tag{4.74}$$

with $\beta \in \mathbb{R}^{10}$. Using Algorithm 4.2, the Q-function parameters converged to:

$$\beta^*_{(1)} = \begin{bmatrix} 11.3564 \\ -10.0880 \\ -20.7299 \\ 0.6605 \\ 21.6084 \\ 4.0903 \\ 1.0495 \\ 70.4224 \\ -0.8534 \\ 0.0910 \end{bmatrix}^\top \tag{4.75}$$

with:

$$Q^* = \begin{bmatrix} Q_1 + \lambda A_1^\top P_1^* A_1 & \lambda A_1^\top P_1^* B_1 \\ \lambda B_1^\top P_1^* A_1 & R + \lambda B_1^\top P_1^* B_1 \end{bmatrix} = \begin{bmatrix} \beta^{(1)} & 0.5\beta^{(2)} & 0.5\beta^{(3)} & 0.5\beta^{(4)} \\ 0.5\beta^{(2)} & \beta^{(5)} & 0.5\beta^{(6)} & 0.5\beta^{(7)} \\ 0.5\beta^{(3)} & 0.5\beta^{(6)} & \beta^{(8)} & 0.5\beta^{(9)} \\ 0.5\beta^{(4)} & 0.5\beta^{(7)} & 0.5\beta^{(9)} & \beta^{(10)} \end{bmatrix} \tag{4.76}$$

Corresponding controller gains are then derived according to (4.60) as:

$$\begin{aligned} \pi(\boldsymbol{X}) &= \arg\min_{\tilde{u}} \left(\beta^\top \Psi(\boldsymbol{X},\tilde{\boldsymbol{u}})\right) \\ &= -0.5 * \beta^{(10)^{-1}} \left(\beta^{(4)} x_1 + \beta^{(7)} x_2 + \beta^{(9)} z\right) = \theta_a^\top \boldsymbol{X} \end{aligned} \tag{4.77}$$

Therefore, the optimal controller gains with β_1^* are computed as:

$$\theta^*_{a(1)} = [-3.6277; -5.7644; 4.6873] = -\hat{K}^*_{1(1)} \tag{4.78}$$

After variation of the system drift matrix to $A_{(2)}$ during simulation, the parameters re-converged to new optimal values as:

$$\beta^*_{(2)} = \begin{bmatrix} 23.4941 \\ -52.7916 \\ -70.3222 \\ 0.2319 \\ 56.9123 \\ 70.2357 \\ 1.6150 \\ 129.0261 \\ -0.8527 \\ 0.0909 \end{bmatrix}^\top \tag{4.79}$$

and

$$\theta^*_{a(2)} = [-1.2757; -8.8839; 4.6907] = -\hat{K}^*_{1(2)} \tag{4.80}$$

Figure 4.8 shows the online adaptation and convergence of the Q-function parameters before and after the system variation respectively. After convergence to the optimal values, the system response using the QFA based RL adaptations are as shown in Figure 4.7. The QFA RL approach therefore provides a completely model-free online tracking control solutions.

4.3.2 Case study 2

This case study addresses the optimal tracking control problem in a buck power converter system which is subject to uncertain or varying component tolerances under different operating conditions. Consider a buck power converter with a switching element and consisting of an inductor L_p with a small series resistance r, a capacitor C_p and a diode. The voltage drop in the diode can be neglected as the value is typically small [121]. For a continuous conduction mode operation (CCM), the control input is defined as the duty-ratio $u \in [0, 1]$ and the buck converter

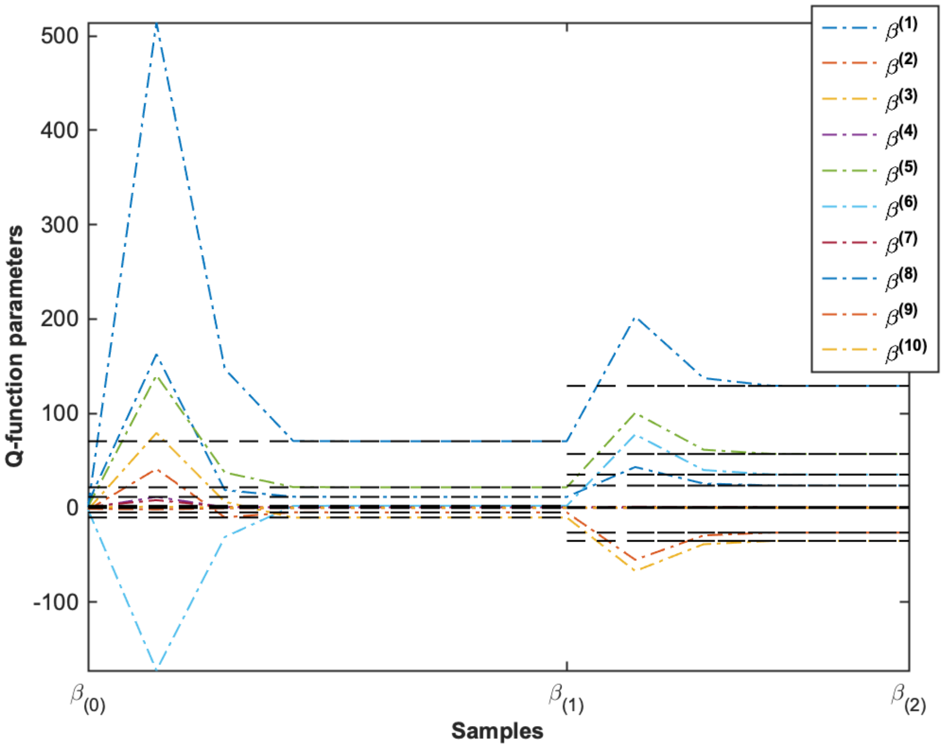


Figure 4.8: Online adaptation and convergence of the Q-function parameters to the optimal values (in black dashed lines) using Algorithm 4.2. $\beta_{(0)}$ are the initial sub-optimal controller parameters while $\beta_{(1)}$ and $\beta_{(2)}$ are respectively the identified optimal controller parameters before and after the system variation.

dynamics are given as [121]:

$$L_p \frac{di(t)}{dt} = -ri(t) - v(t) + Eu(t) \tag{4.81}$$

$$C_p \frac{dv(t)}{dt} = i(t) - i_L \tag{4.82}$$

where E is the dc input voltage, i is the inductor current, v is the output voltage, $i_L = \frac{v}{R_L}$ is the load current and R_L is the load resistor. The aim of the controller is to regulate the output voltage to a given v_{ref}. With the states chosen as the inductor current i and output voltage v, a corresponding state-space dynamics is formulated as:

$$\begin{aligned}\dot{\boldsymbol{x}}(t) = \begin{bmatrix} \dot{i}(t) \\ \dot{v}(t) \end{bmatrix} &= \begin{bmatrix} \frac{-r}{L_p} & \frac{1}{L_p} \\ \frac{1}{C_p} & \frac{-1}{C_pR_L} \end{bmatrix} \begin{bmatrix} i(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} \frac{E}{L_p} \\ 0 \end{bmatrix} u(t) \\ &= A\boldsymbol{x}(t) + Bu(t) \qquad (4.83) \\ y(t) &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} i(t) \\ v(t) \end{bmatrix} \\ y(t) &= C\boldsymbol{x}(t) \qquad (4.84)\end{aligned}$$

The system component parameters are given as $r = 0.5\Omega$, $L_p = 1mH$, $C_p = 50\mu F$ and $E = 48V$. Variations can occur due to modelling uncertainties and component tolerances under different operating conditions. For this example, the load resistor is changed instantaneously from $R_L = 200\Omega$ to 100Ω and is assumed unknown. To demonstrate the proposed online tracking RL framework, the augmented system (4.40) is formed with sampling time $t_s = 100\mu s$ while the tracking cost parameters in (4.36) are given as $Q_1 = \mathcal{I}(3)$, $R = 0.5$ and $\gamma = 1$.

Using Algorithm 4.2, an initially sub-optimal I-P tracking controller is selected as $K_{1(0)} = [0.3086\ 0.1856\ -0.0810]$ while the corresponding Q-functions are approximated as in (4.74). Algorithm 4.2 is thereafter run till convergence as this does not require any knowledge of the system dynamics. With the initially unknown $R_L = 200\Omega$, the Q-function parameters converged to:

$$\beta^*_{(1)} = \begin{bmatrix} 8.9375 \\ 5.3019 \\ -6.7680 \\ 50.0148 \\ 2.1108 \\ -3.5540 \\ 14.5264 \\ 10003.7412 \\ -18.3036 \\ 84.5461 \end{bmatrix}^\top \tag{4.85}$$

while the adapted optimal control gains are computed from (4.77) as:

$$\theta_{a(1)} = [-0.2958; -0.0859; 0.1082] = -\hat{K}^*_{1(1)} \tag{4.86}$$

Figure 4.9 shows the convergence of the online adaptation of the Q-function parameters as compared with the optimal but assumed unknown values.

With a variation in the load resistor to $R_L = 100\Omega$, the Q-function parameters re-converged to:

$$\beta^*_{(2)} = \begin{bmatrix} 8.8961 \\ 5.2366 \\ -6.7850 \\ 49.8444 \\ 2.0918 \\ -3.5425 \\ 14.3679 \\ 10004.1098 \\ -18.3944 \\ 84.3797 \end{bmatrix}^\top \tag{4.87}$$

as shown in Figure 4.10 and to optimal control gains:

$$\theta_{a(2)} = [-0.2954; -0.0851; 0.1090] = -\hat{K}^*_{1(2)} \tag{4.88}$$

Figure 4.11 shows the overall buck power converter system response at the

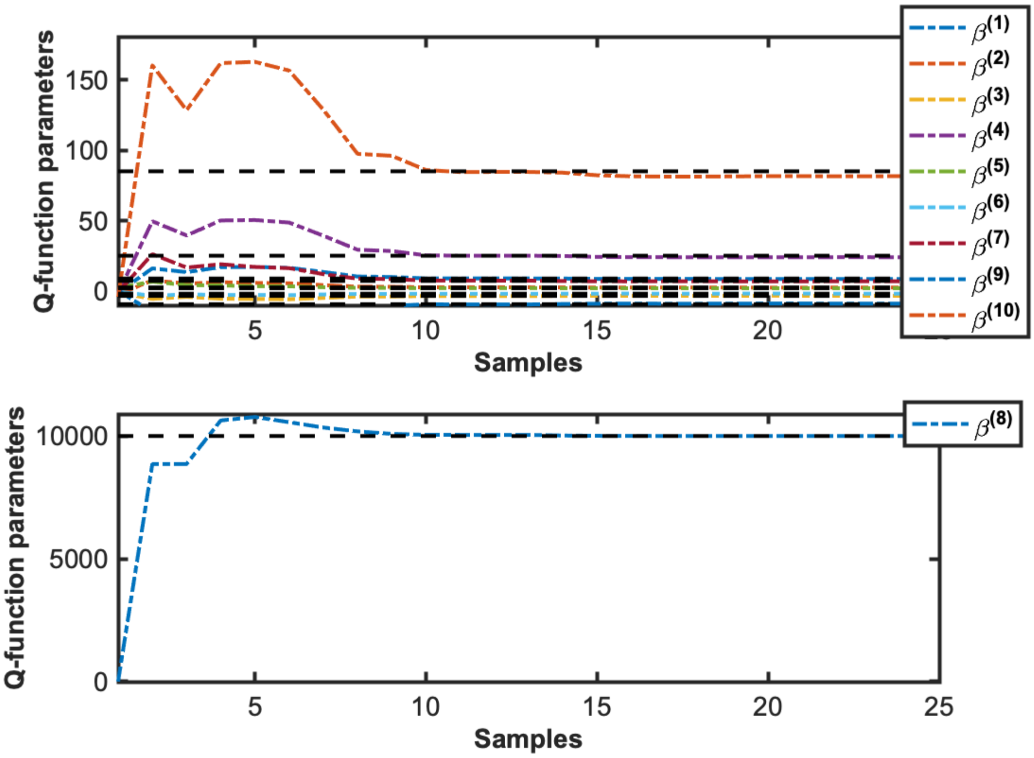


Figure 4.9: Online adaptation and convergence of the Q-function parameters of the buck power converter to the optimal values (in black dashed lines) using Algorithm 4.2.

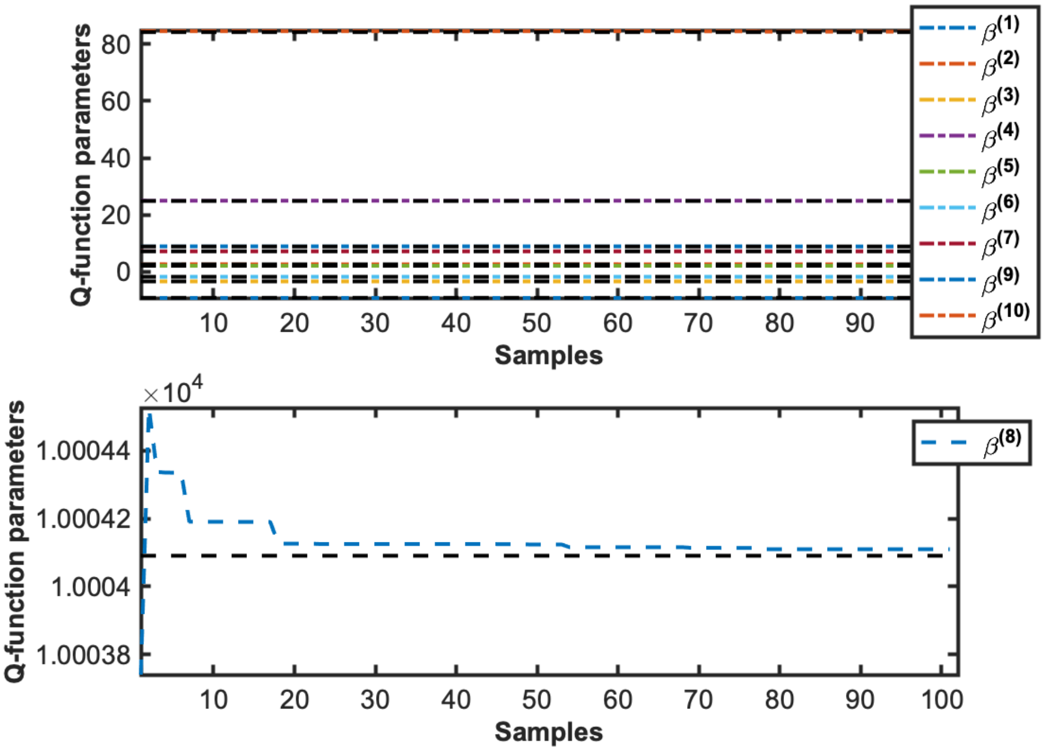


Figure 4.10: Online adaptation and convergence of the Q-function parameters of the buck power converter to the optimal values (in black dashed lines) after variation in the load resistor R_L using Algorithm 4.2.

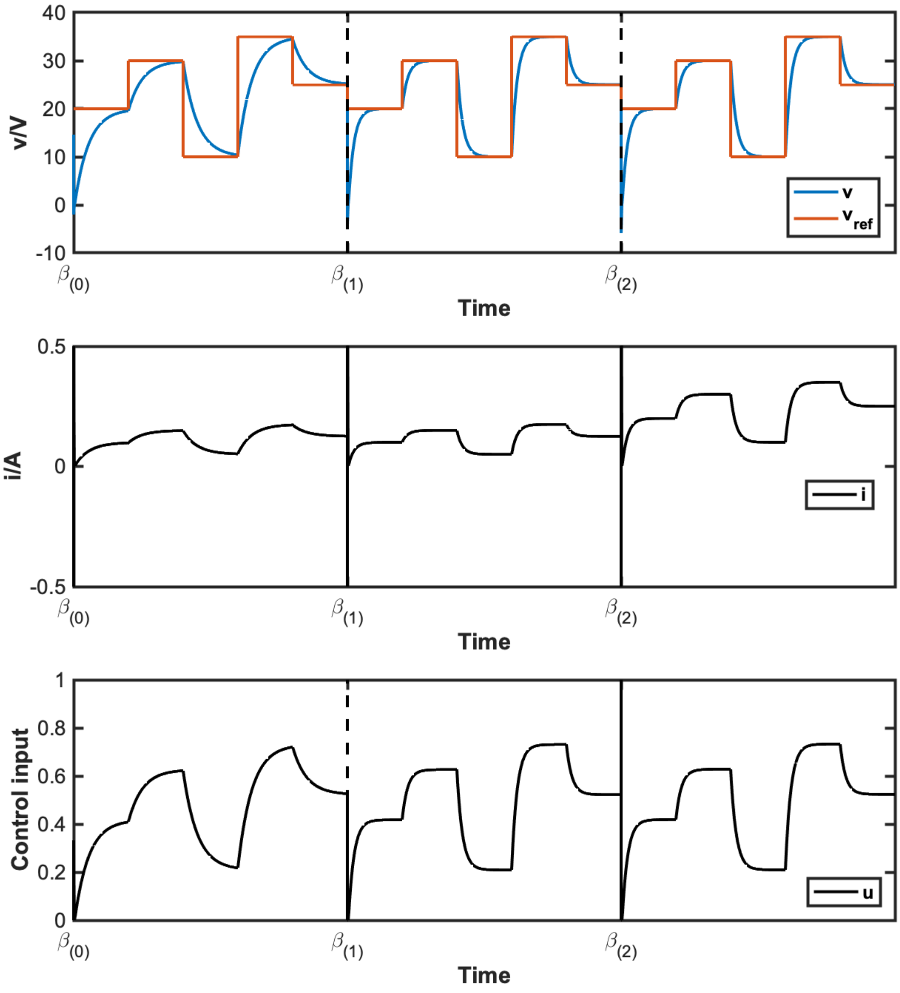


Figure 4.11: Buck power converter response showing the system states and control input at the various stages of the reinforcement learning adaptations. Region with $\beta_{(0)}$ shows the response using the initial sub-optimal controller gains, while region with $\beta_{(1)}$ shows the response from the adapted controller gains to the optimal values using the proposed Algorithms. Following variations in the load resistor R_L, region with $\beta_{(2)}$ onwards shows the response after adaptation to the new optimal control gains.

various stages of the online RL adaptation. The region with $\beta_{(0)}$ in the figure corresponds to the system response using the initially sub-optimal tracking controller gains, while the region with $\beta_{(1)}$ shows the system response after convergence to the optimal controller values from the RL adaptation. Following variation in the load resistor R_L, the system performance after convergence to the new optimal control gains is then shown in the region with $\beta_{(2)}$. This way, the proposed online optimal and adaptive tracking RL framework is able to maintain the desired level of system performance subject to gradual variations in the system parameters.

Summary

This chapter has proposed and demonstrated a condition-based optimal/adaptive online RL tracking controller using an augmented formulation with integral control for varying DT systems. Existing online RL methods either assume a pre-determined feedforward input for the tracking control, or use restrictive assumptions on the reference model dynamics and discounted tracking costs. Moreover, the existing methods are unable to guarantee zero steady-state tracking error. In contrast, the proposed frameworks transform the DT optimal tracking control problem to one of regulation and solves the resulting DT AREs without knowledge of the system dynamics or any restrictive assumptions of the existing online RL methods. Implementation of the framework is shown on representative case studies using the developed VFA and QFA RL approximation techniques to provide a through-life adaptation strategy for the controller gains and guarantee zero steady-state tracking error.

Chapter 5

Output-feedback control for time-varying dynamical systems using reinforcement learning

Previous chapters have developed online RL frameworks for the control of time-varying dynamical systems. Consequently, applications to the class of propulsion and power systems that integrate both the feedforward and feedback RL adaptations into existing controller structures have been shown. However, the developed frameworks have all assumed full measurements of the complete state vectors for use in the control of the dynamical systems. In practice, measurements of the complete state vectors may be unavailable - as a result, existing RL techniques cannot be implemented in their current form. In addition, the usual design of state estimators requires a known model or structure of the system dynamics which is difficult for systems with unknown dynamics and variations.

Information about the unknown system states and variation dynamics can however be obtained in the systems' input/output data. Control techniques that are enabled using the input/output data without any state estimators are called output-feedback (OPFB) methods and belong to the general class of data-based control techniques. This chapter therefore presents the development of output-feedback RL techniques that do not require full measurements of the complete state vector, but make use of only the input/output data for the control of the dynamical systems.

The development of the OPFB RL methods are first shown for the linear quadratic regulation (LQR) problem of discrete-time (DT) systems that produces a polynomial auto-regressive moving-average (ARMA) controller with comparable performance to the state-feedback equivalent. In the absence of state estimators or complete state measurements, extension of the OPFB method is thereafter provided for the condition-based RL framework for the online optimal tracking problem presented in Chapter 4. The strategies and results discussed in this chapter are based on the author's work in [122]. A summary of the main contributions presented in this chapter are as follows:

- An OPFB online RL solution to the discrete-time infinite-horizon linear quadratic tracking (LQT) problem using an augmented formulation with integral control is proposed. The proposed approach does not impose restrictive assumptions on the reference model dynamics and is able to eliminate the steady-state tracking error similar to the state-feedback equivalent, but using only the input/output data.

- A condition-based online OPFB RL framework using the augmented formulation with integral control is developed for systems with unknown or time-varying dynamics. The framework integrates the RL adaptations into an ARMA controller structure and provides a through-life adaptation strategy for use in practical systems.

In the following, Section 5.1 develops the conventional output-feedback formulation for the linear quadratic regulation problem and presents the RL solutions. Section 5.2 provides extension to the linear quadratic tracking problem using the augmented formulation with integral control while Section 5.3 provides simulation examples using the proposed OPFB online RL tracking techniques.

5.1 Optimal and adaptive control using output-feedback reinforcement learning methods

For the development of the OPFB control methods, we consider the optimal regulation problem discussed in Sections 2.2.1.2 and 3.1 for which baseline control solutions using the calculus of variation, dynamic programming and reinforcement learning methods have been provided. The regulation problem is described for the linear quadratic case with dynamics given as:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A\boldsymbol{x}_k + B\boldsymbol{u}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k \end{aligned} \tag{5.1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{u} = \mu(\boldsymbol{x}) \in \mathbb{R}^m$ and $\boldsymbol{y} \in \mathbb{R}^p$ are respectively the system states, control inputs under policy $\mu(\cdot)$ and system outputs. It is assumed that the pairs (A, B) and (A, C) are respectively controllable and observable for any finite initial condition. An associated infinite-horizon performance cost of (3.3) for the regulatory control is expressed as:

$$\begin{aligned} J(\boldsymbol{x}_k) &= \sum_{n=k}^{\infty} \gamma^{n-k} \big(\boldsymbol{x}_n^\top Q \boldsymbol{x}_n + \boldsymbol{u}_n^\top R \boldsymbol{u}_n \big) \\ &= \sum_{n=k}^{\infty} \gamma^{n-k} \big(\boldsymbol{x}_n^\top C^\top Q_y C \boldsymbol{x}_n + \boldsymbol{u}_n^\top R \boldsymbol{u}_n \big) \\ &= \sum_{n=k}^{\infty} \gamma^{n-k} \big(\boldsymbol{y}_n^\top Q_y \boldsymbol{y}_n + \boldsymbol{u}_n^\top R \boldsymbol{u}_n \big) \end{aligned} \tag{5.2}$$

where $Q_y = Q_y^\top \geq 0$ and $Q = C^\top Q_y C$. Equation (5.2) gives the LQR cost in terms of the input/output dynamics. The LQR control therefore aims to regulate the system outputs to zero whilst stabilising the closed-loop system asymptotically on some set $\Omega \in \mathbb{R}^n$. As discussed in Section 2.2.1, the optimal cost or value of the control using the Bellman optimality equation is given as:

$$\begin{aligned} V^*(\boldsymbol{x}_k) &= \min_{\boldsymbol{u}} \left\{ \sum_{n=k}^{\infty} \gamma^{n-k} \big(\boldsymbol{y}_n^\top Q_y \boldsymbol{y}_n + \boldsymbol{u}_n^\top R \boldsymbol{u}_n \big) \right\} \\ &= \min_{\boldsymbol{u}} \big(\boldsymbol{y}_k^\top Q_y \boldsymbol{y}_k + \boldsymbol{u}_k^\top R \boldsymbol{u}_k + \gamma V^*(\boldsymbol{x}_{k+1}) \big) \end{aligned} \tag{5.3}$$

The optimal control input can then be obtained as:

$$\mu^*(\boldsymbol{x}_k) = \arg\min_{\boldsymbol{u}} \big(\boldsymbol{y}_k^\top Q_y \boldsymbol{y}_k + \boldsymbol{u}_k^\top R \boldsymbol{u}_k + \gamma V^*(\boldsymbol{x}_{k+1}) \big) \tag{5.4}$$

For the LQR problem, the value is known to be quadratic in terms of some symmetric positive semi-definite matrix $P = P^\top \in \mathbb{R}^{n \times n}$ given as:

$$V(\boldsymbol{x}_k) = \boldsymbol{x}_k^\top P \boldsymbol{x}_k \tag{5.5}$$

Substituting for (5.5) in (5.4) and setting the derivative with respect to the control input to zero yields the optimal control input in terms of P as:

$$\boldsymbol{u}_k^* = -\Big(\frac{R}{\gamma} + B^\top P B \Big)^{-1} B^\top P A \boldsymbol{x}_k \tag{5.6}$$

with Ricatti equation:

$$P = C^\top Q_y C + \gamma A^\top P A - \gamma A^\top P B(\frac{R}{\gamma} + B^\top P B)^{-1} B^\top P A \tag{5.7}$$

This gives the model-based solution to the LQR problem as discussed in Section 2.2.1.2. To enable OPFB approaches, both the state dynamics and value function are first expressed in terms of the available input/output data.

5.1.1 State dynamics in terms of measured input/output data

Given the current time step k and a time horizon N, the state dynamics for system (5.1) can be written over the horizon $[k-N, k]$ through successive recursions as:

$$\boldsymbol{x}_k = A^N \boldsymbol{x}_{k-N} + U_N \bar{\boldsymbol{u}}_{k-1,k-N} \tag{5.8}$$

where $U_N = \begin{bmatrix} B & AB & A^2B & \cdots & A^{N-1}B \end{bmatrix} \in \mathbb{R}^{n\times mN}$ is the controllability matrix and $\bar{\boldsymbol{u}}_{k-1,k-N} = \begin{bmatrix} \boldsymbol{u}_{k-1} \\ \boldsymbol{u}_{k-2} \\ \vdots \\ \boldsymbol{u}_{k-N} \end{bmatrix} \in \mathbb{R}^{mN}$ is the measured input data over the interval $[k-N, k-1]$. Using the recursive relationship, the output dynamics for system (5.1) becomes:

$$\boldsymbol{y}_k = C\boldsymbol{x}_k = CA^N \boldsymbol{x}_{k-N} + CU_N \bar{\boldsymbol{u}}_{k-1,k-N} \tag{5.9}$$

Similar recursion for the output dynamics over the time interval $[k-N, k-1]$ can then be written as:

$$\bar{\boldsymbol{y}}_{k-1,k-N} = V_N \boldsymbol{x}_{k-N} + T_N \bar{\boldsymbol{u}}_{k-1,k-N} \tag{5.10}$$

where $V_N = \begin{bmatrix} CA^{N-1} \\ \vdots \\ CA \\ C \end{bmatrix} \in \mathbb{R}^{pN\times n}$ is the observability matrix,

$T_N = \begin{bmatrix} 0 & CB & CAB & \cdots & CA^{N-2}B \\ 0 & 0 & CB & \cdots & CA^{N-3}B \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 & CB \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{pN\times mN}$ is the Toeplitz matrix of

Markov parameters and $\bar{\boldsymbol{y}}_{k-1,k-N} = \begin{bmatrix} \boldsymbol{y}_{k-1} \\ \boldsymbol{y}_{k-2} \\ \vdots \\ \boldsymbol{y}_{k-N} \end{bmatrix} \in \mathbb{R}^{pN}$ is the measured output data.

Since the pair (A, C) is observable, it was shown in Lewis and Vamvoudakis [123] that there exists an observability index K such that the controllability matrix V_N has a full column rank $rank(V_N) = n \; |_{N \geq K}$ with left inverse $V_N^+ = (V_N^\top V_N)^{-1} V_N^\top$. The following relationship therefore holds:

$$\begin{aligned} A^N &= \mathcal{M} V_N \\ \therefore \mathcal{M} &= A^N V_N^+ + \mathcal{Z}(I - V_N V_N^+) = \mathcal{M}_0 + \mathcal{M}_1 \end{aligned} \tag{5.11}$$

for matrix $\mathcal{M} \in \mathbb{R}^{n \times pN}$ and any arbitrary vector $\mathcal{Z}$. Let $N \geq K$, then V_N has a full column rank, in which case $(I - V_N V_N^+)$ is a zero matrix, and $\mathcal{M}_1 = 0$. Substituting for (5.11) and for the output dynamics (5.10) in (5.8), the state dynamics becomes:

$$\begin{aligned} \boldsymbol{x}_k &= \mathcal{M}_0 V_N \boldsymbol{x}_{k-N} + U_N \bar{\boldsymbol{u}}_{k-1,k-N} \\ &= \mathcal{M}_0 (\bar{\boldsymbol{y}}_{k-1,k-N} - T_N \bar{\boldsymbol{u}}_{k-1,k-N}) + U_N \bar{\boldsymbol{u}}_{k-1,k-N} \\ &= \mathcal{M}_0 \bar{\boldsymbol{y}}_{k-1,k-N} + (U_N - \mathcal{M}_0 T_N) \bar{\boldsymbol{u}}_{k-1,k-N} \end{aligned} \tag{5.12}$$

$$\therefore \boldsymbol{x}_k = \begin{bmatrix} \mathcal{M}_u & \mathcal{M}_y \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{u}}_{k-1,k-N} \\ \bar{\boldsymbol{y}}_{k-1,k-N} \end{bmatrix} \tag{5.13}$$

where $\mathcal{M}_u = (U_N - \mathcal{M}_0 T_N)$ and $\mathcal{M}_y = \mathcal{M}_0$. Equation (5.13) gives the state dynamics in terms of the measured input/output data. By defining the vector $\bar{\boldsymbol{z}}_{k-1,k-N} = \begin{bmatrix} \bar{\boldsymbol{u}}_{k-1,k-N} \\ \bar{\boldsymbol{y}}_{k-1,k-N} \end{bmatrix}$, the value function (5.5) can then be expressed in terms of the past inputs and outputs as:

$$\begin{aligned} V(\boldsymbol{x}_k) &= \boldsymbol{x}_k^\top P \boldsymbol{x}_k \\ &= \bar{\boldsymbol{z}}_{k-1,k-N}^\top \begin{bmatrix} \mathcal{M}_u^\top P \mathcal{M}_u & \mathcal{M}_u^\top P \mathcal{M}_y \\ \mathcal{M}_y^\top P \mathcal{M}_u & \mathcal{M}_y^\top P \mathcal{M}_y \end{bmatrix} \bar{\boldsymbol{z}}_{k-1,k-N} \\ &= \bar{\boldsymbol{z}}_{k-1,k-N}^\top \bar{P} \bar{\boldsymbol{z}}_{k-1,k-N} \end{aligned} \tag{5.14}$$

where $\bar{P} \in \mathcal{R}^{(m+p)N \times (m+p)N}$. It is noted that the given expressions for the state dynamics and value function in terms of the input/output data still require knowledge of the dynamics i.e. (A, B, C) to compute $\mathcal{M}_u$ and $\mathcal{M}_y$ and are classed as model-based output-feedback solutions. The next section presents the Bellman optimality equations for the output-feedback control that enables the development of model-free RL approaches.

5.1.2 Bellman optimality equations for the output-feedback optimal regulation problem

Using the Bellman principle of optimality, the optimum value for (5.14) is expressed as:

$$\begin{aligned} V^*(\boldsymbol{x}_k) &= \min_{\boldsymbol{u}} \left(\boldsymbol{y}_k^\top Q_y \boldsymbol{y}_k + \boldsymbol{u}_k^\top R \boldsymbol{u}_k + \gamma V^*(\boldsymbol{x}_{k+1}) \right) \\ &= \min_{\boldsymbol{u}} \left(\boldsymbol{y}_k^\top Q_y \boldsymbol{y}_k + \boldsymbol{u}_k^\top R \boldsymbol{u}_k + \gamma \bar{\boldsymbol{z}}_{k,k-N+1}^\top \bar{P}^* \bar{\boldsymbol{z}}_{k,k-N+1} \right) \end{aligned} \tag{5.15}$$

Equation (5.15) gives the discrete-time (DT) Hamilton-Jacobi-Bellman (HJB) equation for the output-feedback (OF) optimal regulation problem using only the system's input/output data and without any state estimations. The optimal regulatory control input can then be obtained as:

$$\mu^*(\boldsymbol{x}_k) = \boldsymbol{u}_k^* = \arg\min_{\boldsymbol{u}} \left(\boldsymbol{y}_k^\top Q_y \boldsymbol{y}_k + \boldsymbol{u}_k^\top R \boldsymbol{u}_k + \gamma \bar{\boldsymbol{z}}_{k,k-N+1}^\top \bar{P}^* \bar{\boldsymbol{z}}_{k,k-N+1} \right) \tag{5.16}$$

Let $\boldsymbol{z}_{k,k-N+1}^\top \bar{P}^* \bar{\boldsymbol{z}}_{k,k-N+1}$ be partitioned as:

$$\begin{bmatrix} \boldsymbol{u}_k \\ \bar{\boldsymbol{u}}_{k-1,k-N+1} \\ \bar{\boldsymbol{y}}_{k,k-N+1} \end{bmatrix}^\top \begin{bmatrix} p_0 & p_u & p_y \\ p_u^\top & p_{22} & p_{23} \\ p_y^\top & p_{32} & p_{33} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_k \\ \bar{\boldsymbol{u}}_{k-1,k-N+1} \\ \bar{\boldsymbol{y}}_{k,k-N+1} \end{bmatrix} \tag{5.17}$$

where $p_0 \in \mathbb{R}^{m \times m}$, $p_u \in \mathbb{R}^{m \times (m(N-1))}$ and $p_y \in \mathbb{R}^{m \times pN}$. Equating the derivative of (5.16) to zero and simplifying gives:

$$\boldsymbol{u}_k^* = -(R + p_0^*)^{-1}(p_u^* \bar{\boldsymbol{u}}_{k-1,k-N+1} + p_y^* \bar{\boldsymbol{y}}_{k,k-N+1}) \tag{5.18}$$

Equation (5.18) gives the optimal control input for the optimal regulation problem in form of a dynamic polynomial autoregressive moving average (ARMA) controller that generates current control input using past input/output data [123]. For systems with unknown dynamics or variations, reinforcement learning approaches to the OPFB LQR problem can then be developed using function approximations and the iterative forward-in-time methods (VI or PI) to solve the corresponding Bellman optimality equations using only the input/output data. Consequently, extension of the OPFB control of dynamical systems is provided in the next section for the tracking control problem introduced in Chapter 4 for systems where measurements of the complete state vectors may be unavailable or the design of state estimators is difficult.

5.2 Output-feedback tracking with integral control using reinforcement learning

An output-feedback (OPFB) RL solution to the discrete-time (DT) infinite-horizon linear quadratic tracking (LQT) problem is presented for systems with unknown dynamics or variations. The tracking problem is transformed to one of regulation by forming an augmented system consisting of the original system dynamics and the integral of the error between the reference input and the tracked output. Similar to the regulation control problem, the augmented system states and the corresponding value function are expressed in terms of the available input/output data, eliminating the need to have state estimators which may be difficult to design for systems with the unknown dynamics or variations. In contrast to existing OPFB RL techniques for the tracking control that make restrictive assumptions on the reference model dynamics or discounted performance cost, the proposed approach makes no such assumptions and guarantees zero steady-state tracking error. The next section presents the LQT problem for which output-feedback RL solutions are later developed for.

5.2.1 The LQT problem

For the development of the LQT problem, consider the general DT system with the following dynamics:

$$\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k; \;\; \boldsymbol{y}_k = C\boldsymbol{x}_k \tag{5.19}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$, $\boldsymbol{u}_k \in \mathbb{R}^m$ and $\boldsymbol{y}_k \in \mathbb{R}^p$ are respectively the system states, inputs and outputs. It is assumed that the pairs (A, B) and (A, C) are respectively controllable and observable for any finite initial condition. An associated infinite-horizon performance cost for the tracking control for a given reference trajectory $\boldsymbol{r}_k$ is:

$$J(\boldsymbol{x}_k) = \sum_{n=k}^{\infty} \mathcal{L}(\boldsymbol{x}_n, \boldsymbol{u}_n) \tag{5.20}$$

where the reward signal is given as the quadratic energy function $\mathcal{L}(\boldsymbol{x}_k, \boldsymbol{u}_k) = (\boldsymbol{y}_k - \boldsymbol{r}_k)^\top Q_T(\boldsymbol{y}_k - \boldsymbol{r}_k) + \boldsymbol{u}_k^\top R\boldsymbol{u}_k$, $Q_T \geq 0$ and $R > 0$. The aim of the optimal tracking control problem is therefore to determine the control policy $\boldsymbol{u} = \pi(\boldsymbol{x})$ that minimises the tracking cost (5.20) and guarantees the system stability such that the system output tracks the desired reference. Using the calculus of variations, it can be shown that the standard solution to the LQT problem is given as:

$$\boldsymbol{u}_k = -K^x \boldsymbol{x}_k + K^v \boldsymbol{v}_{k+1} \tag{5.21}$$

where:

$$\begin{aligned} K^x &= (B^\top P B + R)^{-1} B^\top P A; \;\; K^v = (B^\top P B + R)^{-1} B^\top \\ \boldsymbol{v}_k &= (A - BK^x)^\top \boldsymbol{v}_{k+1} + C^\top Q_T \boldsymbol{r}_k \end{aligned} \tag{5.22}$$

and $P = P^\top > 0$ is the solution to the algebraic Riccati equation [4]. It is however noted that the given standard solution is *non-causal* [75] as it is dependent on a *backwards-in-time* recursion of the vector sequence $\boldsymbol{v}_k$. A direct implication of this is that the standard solution to the tracking problem can only be computed offline and with full knowledge of the system dynamics. Causal solution strategies have thus been proposed to enable online computation of the tracking control input. These strategies make use of state augmentation thereby transforming the tracking problem into one of regulation and are presented next.

5.2.1.1 State augmentation with reference dynamics

This approach assumes that the reference dynamics is governed by $\boldsymbol{r}_{k+1} = F\boldsymbol{r}_k$, where F is Hurwitz [38]. An augmented system is then formed by using the system dynamics (5.19) and that of the reference as follows:

$$\boldsymbol{X}^r_{k+1} = \begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{r}_{k+1} \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & F \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{r}_k \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \boldsymbol{u}_k \tag{5.23}$$

Following this, the tracking cost of (5.20) becomes:

$$\begin{aligned} J(\boldsymbol{X}^r_k) &= \sum_{n=k}^{\infty} \left[(C\boldsymbol{x}_n - \boldsymbol{r}_n)^\top Q_T (C\boldsymbol{x}_n - \boldsymbol{r}_n) + \boldsymbol{u}_n^\top R \boldsymbol{u}_n \right] \\ &= \sum_{n=k}^{\infty} \left[\boldsymbol{X}^r_n Q_r \boldsymbol{X}^r_n + \boldsymbol{u}_n^\top R \boldsymbol{u}_n \right] \end{aligned} \tag{5.24}$$

where $Q_r = \begin{bmatrix} C^\top Q_T C & -C^\top Q_T \\ -C^\top Q_T & Q_T \end{bmatrix}$. This way, the tracking problem is converted into a regulation problem without the need of any backwards-in-time variable recursion.

Remarks

- By assuming that F is Hurwitz, then $\boldsymbol{r}_k \to 0$ as $k \to \infty$; this is a restrictive assumption that limits extension of the approach to any practical reference tracking problem.

- The Hurwitz assumption on F can be relaxed by using a discount factor in the tracking cost (5.24) to ensure that the value remains finite if $\boldsymbol{r}_k \nrightarrow 0$ as proposed in [76] and [75]. However, by introducing a discount factor, the approaches cannot guarantee zero steady-state tracking error and are restrictive to the class of reference generators that can be used.

- Measurements or knowledge of the entire state vector is needed.

5.2.1.2 State augmentation with integral control

In the following, an augmented formulation for the optimal tracking problem with integral control is developed. Consider a new state $\dot{\boldsymbol{z}}$ defined as the integral of the difference between the desired reference and the tracked output as $\dot{\boldsymbol{z}}(t) = \int \big(\boldsymbol{r}(t) - \boldsymbol{y}(t)dt\big)$, where $\boldsymbol{z} \in \mathbb{R}^p$. Using Euler's approximation, an equivalent DT state with sampling time t_s is given as:

$$z_{k+1} = z_k + t_s\big(\boldsymbol{r}_k - C\boldsymbol{x}_k\big) \tag{5.25}$$

An augmented system can then be formulated using the new integral state as follows:

$$\begin{bmatrix} \boldsymbol{x}_{k+1} \\ z_{k+1} \end{bmatrix} = \begin{bmatrix} A & 0 \\ -t_sC & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ z_k \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \boldsymbol{u}_k + \begin{bmatrix} 0 \\ t_sI \end{bmatrix} \boldsymbol{r}_k \tag{5.26}$$

At steady-state, the augmented system (5.26) becomes:

$$\begin{bmatrix} \boldsymbol{x}_{\infty} \\ z_{\infty} \end{bmatrix} = \begin{bmatrix} A & 0 \\ -t_sC & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{\infty} \\ z_{\infty} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \boldsymbol{u}_{\infty} + \begin{bmatrix} 0 \\ -t_sI \end{bmatrix} \boldsymbol{r}_{\infty} \tag{5.27}$$

For a constant reference signal i.e. $\boldsymbol{r}_{\infty} = \boldsymbol{r}_k$, subtracting (5.27) from (5.26) gives:

$$\begin{aligned} \boldsymbol{X}_{k+1} &= \begin{bmatrix} A & 0 \\ -t_sC & I \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k - \boldsymbol{x}_{\infty} \\ z_k - z_{\infty} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} (\boldsymbol{u}_k - \boldsymbol{u}_{\infty}) \\ &= A_1\boldsymbol{X}_k + B_1\tilde{\boldsymbol{u}}_k \end{aligned} \tag{5.28}$$

where $\boldsymbol{X}_k = \begin{bmatrix} \boldsymbol{x}_k - \boldsymbol{x}_\infty \\ \boldsymbol{z}_k - \boldsymbol{z}_\infty \end{bmatrix}$ and $\tilde{\boldsymbol{u}}_k = (\boldsymbol{u}_k - \boldsymbol{u}_\infty)$ with cost:

$$J(\boldsymbol{X}_k) = \sum_{n=k}^{\infty} \left(\boldsymbol{X}_n^\top Q_x \boldsymbol{X}_n + \tilde{\boldsymbol{u}}_n^\top R \tilde{\boldsymbol{u}}_n \right) \tag{5.29}$$

where $Q_x \in \mathbb{R}^{(n+p)\times(n+p)}$. The tracking problem is thus converted to one of regulation such that the control input for a minimum of (5.29) eliminates the steady-state error by ensuring that $\boldsymbol{x}_k \rightarrow \boldsymbol{x}_\infty$ and $\boldsymbol{z}_k \rightarrow \boldsymbol{z}_\infty$ as $\boldsymbol{X}_k \rightarrow 0$.

Remark

- A RL framework using the integral state augmentation that does not impose any restrictive assumptions on the reference dynamics or the use of discounted tracking cost has been proposed in [118]. However, similar to the existing RL strategies, measurements or knowledge of the entire state vector is needed.

Using the frameworks developed in [118] and [123], an OPFB RL solution to the LQT problem that guarantees zero steady-state tracking error and does not require measurements of the entire state vector is presented next.

5.2.2 OPFB solution to the LQT problem

Using the state augmentation framework with integral control presented in Chapter 4.2, an OPFB solution using only the measured input/output data is developed as follows. The output dynamics of system (5.28) is assumed to be given as:

$$\boldsymbol{Y}_k = \begin{bmatrix} C & I \end{bmatrix} \boldsymbol{X}_k = C_1 \boldsymbol{X}_k \tag{5.30}$$

The tracking cost (5.29) is therefore redefined in terms of the input/output dynamics as:

$$\begin{aligned} J(\boldsymbol{X}_k) &= \sum_{n=k}^{\infty} \left(\boldsymbol{X}_n^\top Q_x \boldsymbol{X}_n + \tilde{\boldsymbol{u}}_n^\top R \tilde{\boldsymbol{u}}_n \right) \\ &= \sum_{n=k}^{\infty} \left(\boldsymbol{X}_n^\top C_1^\top Q C_1 \boldsymbol{X}_n + \tilde{\boldsymbol{u}}_n^\top R \tilde{\boldsymbol{u}}_n \right) \\ &= \sum_{n=k}^{\infty} \left(\boldsymbol{Y}_n^\top Q \boldsymbol{Y}_n + \tilde{\boldsymbol{u}}_n^\top R \tilde{\boldsymbol{u}}_n \right) \end{aligned} \tag{5.31}$$

where $Q_x = C_1^\top Q C_1$ and $Q = Q^\top \geq 0$. It was shown in [118] that the value function for the tracking cost with the integral state augmentation is quadratic for some matrix $P_1 = P_1^\top > 0$ given as:

$$V(\boldsymbol{X}_k) = \boldsymbol{X}_k^\top P_1 \boldsymbol{X}_k \tag{5.32}$$

where (5.32) is expressed in terms of the augmented state dynamics. Given that real systems often lack full state observability, the following derives the state dynamics and the quadratic value function in terms of only the measured input/output data.

5.2.2.1 Augmented state dynamics using measured input/output data

Consider the time horizon between time steps k and N as $[k-N,k]$. The augmented state dynamics of system (5.28) can be expressed recursively over the horizon as:

$$\boldsymbol{X}_k = A_1^N \boldsymbol{X}_{k-N} + U_{N_1} \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} \tag{5.33}$$

where $U_{N_1} = \begin{bmatrix} B_1 & A_1 B_1 & \cdots & A_1^{N-1} B_1 \end{bmatrix} \in \mathbb{R}^{(n+p)\times mN}$ is the controllability matrix and $\bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} = \begin{bmatrix} \tilde{\boldsymbol{u}}_{k-1} & \tilde{\boldsymbol{u}}_{k-2} & \cdots & \tilde{\boldsymbol{u}}_{k-N} \end{bmatrix}^\top \in \mathbb{R}^{mN}$ is the measured input data over the interval $[k-N, k-1]$. Using the recursive relationship, the augmented output dynamics (5.30) become:

$$\boldsymbol{Y}_k = C_1 \boldsymbol{X}_k = C_1 A_1^N \boldsymbol{X}_{k-N} + C_1 U_{N_1} \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} \tag{5.34}$$

which can be similarly expressed in terms of the measured output data over the interval $[k-N, k-1]$ as:

$$\bar{\boldsymbol{Y}}_{k-1,k-N} = V_{N_1} \boldsymbol{X}_{k-N} + T_{N_1} \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} \tag{5.35}$$

where $\bar{\boldsymbol{Y}}_{k-1,k-N} = \begin{bmatrix} \boldsymbol{Y}_{k-1} & \boldsymbol{Y}_{k-2} & \cdots & \boldsymbol{Y}_{k-N} \end{bmatrix}^\top \in \mathbb{R}^{pN}$ is the measured output data, $V_{N_1} = \begin{bmatrix} C_1 A_1^{N-1} & \cdots & C_1 A_1 & C_1 \end{bmatrix}^\top \in \mathbb{R}^{pN\times(n+p)}$ is the observability matrix and $T_{N_1} = \begin{bmatrix} 0 & C_1 B_1 & \cdots & C_1 A_1^{N-2} B_1 \\ 0 & 0 & \cdots & C_1 A_1^{N-3} B_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{(n+p)N\times mN}$ is the Toeplitz matrix of Markov parameters. Since the pair (A, C) of the original system is observable and of full rank n; it follows that if and only if all the elements of matrix C_1 are non-zero, then the pair (A_1, C_1) is also observable and of full rank $n+p$. Therefore, there exists an observability index $\mathcal{O}_I$ such that V_{N_1} has a full column rank $rank(V_{N_1}) = n + p \,|_{N \geq \mathcal{O}_I}$ with left inverse $V_{N_1}^{+} = (V_{N_1}^{\top} V_{N_1})^{-1} V_{N_1}^{\top}$. Consequently, let $N \geq \mathcal{O}_I$, there exists a matrix $M \in \mathbb{R}^{(n+p)\times pN}$ such that:

$$A_1^N = MV_{N_1}; \;\; M = A_1^N V_{N_1}^{+} \tag{5.36}$$

Substituting for (5.35) and (5.36) in (5.33), the augmented state dynamics becomes:

$$\begin{aligned} \boldsymbol{X}_k &= MV_{N_1}\boldsymbol{X}_{k-N} + U_{N_1}\bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} \\ &= M(\bar{\boldsymbol{Y}}_{k-1,k-N} - T_{N_1}\bar{\tilde{\boldsymbol{u}}}_{k-1,k-N}) + U_{N_1}\bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} \\ &= M\bar{\boldsymbol{Y}}_{k-1,k-N} + (U_{N_1} - MT_{N_1})\bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} && (5.37) \\ \therefore \boldsymbol{X}_k &= \begin{bmatrix} M_u & M_y \end{bmatrix} \begin{bmatrix} \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} \\ \bar{\boldsymbol{Y}}_{k-1,k-N} \end{bmatrix} && (5.38) \end{aligned}$$

where $M_u = (U_{N_1} - MT_{N_1})$ and $M_y = M$. Equation (5.38) expresses the augmented system dynamics in terms of the measured input/output data, albeit knowledge of the system dynamics (A_1, B_1, C_1) is needed to compute M_u and M_y. The next section develops the Bellman optimality equations to solve the augmented integral tracking control problem.

5.2.2.2 Bellman optimality equations using measured input/output data

By defining the vector $\bar{\boldsymbol{Z}}_{k-1,k-N} = \begin{bmatrix} \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N} \\ \bar{\boldsymbol{Y}}_{k-1,k-N} \end{bmatrix} \in \mathbb{R}^{(m+p)N}$ and substituting for $\boldsymbol{X}_k$, the value function (5.32) can be expressed as:

$$\begin{aligned} V(\boldsymbol{X}_k) &= \boldsymbol{X}_k^{\top} P_1 \boldsymbol{X}_k \\ &= \bar{\boldsymbol{Z}}_{k-1,k-N}^{\top} \begin{bmatrix} M_u^{\top} P_1 M_u & M_u^{\top} P_1 M_y \\ M_y^{\top} P_1 M_u & M_y^{\top} P_1 M_y \end{bmatrix} \bar{\boldsymbol{Z}}_{k-1,k-N} && (5.39) \\ \therefore V(\boldsymbol{X}_k) &\equiv \bar{\boldsymbol{Z}}_{k-1,k-N}^{\top} \bar{P}_1 \bar{\boldsymbol{Z}}_{k-1,k-N} && (5.40) \end{aligned}$$

where $\bar{P}_1 \in \mathbb{R}^{(m+p)N\times(m+p)N}$. Using the Bellman principle of optimality, the optimum value becomes:

$$\begin{aligned} V^*(\boldsymbol{X}_k) &= \min_{\tilde{\boldsymbol{u}}_k} \left(\boldsymbol{Y}_k^{\top} Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^{\top} R \tilde{\boldsymbol{u}}_k + V^*(\boldsymbol{X}_{k+1}) \right) \\ &= \min_{\tilde{\boldsymbol{u}}_k} \left(\boldsymbol{Y}_k^{\top} Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^{\top} R \tilde{\boldsymbol{u}}_k + \bar{\boldsymbol{Z}}_{k,k-N+1}^{\top} \bar{P}_1^* \bar{\boldsymbol{Z}}_{k,k-N+1} \right) && (5.41) \end{aligned}$$

Equation (5.41) gives the DT HJB equation for the integral state augmentation for the tracking control problem using input/output data from which the optimal tracking control input can be obtained as:

$$\pi^*(\boldsymbol{X}_k) = \tilde{\boldsymbol{u}}_k^* = \arg\min_{\tilde{\boldsymbol{u}}_k} \left(\boldsymbol{Y}_k^\top Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + \bar{\boldsymbol{Z}}_{k,k-N+1}^\top \bar{P}_1^* \bar{\boldsymbol{Z}}_{k,k-N+1} \right) \tag{5.42}$$

Let $\bar{\boldsymbol{Z}}_{k,k-N+1}^\top \bar{P}_1 \bar{\boldsymbol{Z}}_{k,k-N+1}$ be partitioned as:

$$\begin{bmatrix} \tilde{\boldsymbol{u}}_k \\ \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N+1} \\ \bar{\boldsymbol{Y}}_{k,k-N+1} \end{bmatrix}^\top \begin{bmatrix} P_0 & P_u & P_y \\ P_u^\top & P_{xx}^{(1)} & P_{xx}^{(2)} \\ P_y^\top & P_{xx}^{(3)} & P_{xx}^{(4)} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{u}}_k \\ \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N+1} \\ \bar{\boldsymbol{Y}}_{k,k-N+1} \end{bmatrix} \tag{5.43}$$

where $P_0 \in \mathbb{R}^{m\times m}$, $P_u \in \mathbb{R}^{m\times(m(N-1))}$, $P_y \in \mathbb{R}^{m\times pN}$, $P_{xx}^{(1)} \in \mathbb{R}^{(m(N-1))\times(m(N-1))}$, $P_{xx}^{(2)} \in \mathbb{R}^{(m(N-1))\times pN}$, $P_{xx}^{(3)} \in \mathbb{R}^{pN\times(m(N-1))}$ and $P_{xx}^{(4)} \in \mathbb{R}^{pN\times pN}$. Equating the derivative of (5.42) with respect to $\tilde{\boldsymbol{u}}_k$ to zero and simplifying gives:

$$\begin{aligned} \tilde{\boldsymbol{u}}_k^* &= -(R + P_0^*)^{-1}(P_u^* \bar{\tilde{\boldsymbol{u}}}_{k-1,k-N+1} + P_y^* \bar{\boldsymbol{Y}}_{k,k-N+1}) \\ &= -K^* \times \left[(\bar{\tilde{\boldsymbol{u}}}_{k-1,k-N+1})^\top \quad (\bar{\boldsymbol{Y}}_{k,k-N+1})^\top \right] \end{aligned} \tag{5.44}$$

where $K^* \in \mathbb{R}^{m+pN}$. Equation (5.44) gives the optimal tracking control input that generates current input based on past input/output data and known as a dynamic polynomial autoregressive moving average (ARMA) controller [124]. The next section presents RL frameworks that make use of function approximations to iteratively solve the OPFB optimal tracking problem without knowledge of the system dynamics.

5.2.3 RL framework for the OPFB tracking problem

Model-free RL approaches are enabled by iterative methods that utilize the Bellman optimality equations to develop value and policy update equations which are solved at each time step. One of such iterative methods is policy iteration (PI) which requires an initially admissible policy (i.e. stabilising policy and with a finite cost $V(\cdot)$) and successively alternates between the following equations:

$$V_{k+1}(\boldsymbol{X}_k) = \boldsymbol{Y}_k^\top Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + V_{k+1}(\boldsymbol{X}_{k+1}) \tag{5.45}$$

$$\tilde{\boldsymbol{u}}_{k+1} = \arg\min_{\tilde{\boldsymbol{u}}} \left(\boldsymbol{Y}_k^\top Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + V_{k+1}(\boldsymbol{X}_{k+1}) \right) \tag{5.46}$$

Given an admissible policy $\pi(\boldsymbol{X})$, the value is evaluated by solving (5.45) till convergence while an improved policy is computed using (5.46) and both respectively constitute the policy evaluation and policy update steps of the PI. The PI method is justified in [43] by showing that the improved policy ensures that

$V_{k+1}(\boldsymbol{X}_k) \leq V_k(\boldsymbol{X}_k)$ associated with the monotonicity property of the fixed-point equations. This way, the PI recursion computes a strictly improved policy and convergence to the optimal policy and value under the controllability/observability conditions has been shown in [125]. For practical implementation of the algorithm, the value function (5.40) is approximated as:

$$V(\boldsymbol{X}_k) \approx \Theta^\top \Phi(\bar{\boldsymbol{Z}}_{k-1,k-N}) \tag{5.47}$$

where $\Theta \in \mathbb{R}^{p_c}$ are the value function parameters with basis function $\Phi(\cdot)$. Equation (5.45) becomes:

$$\Theta^\top \Phi(\bar{\boldsymbol{Z}}_{k-1,k-N}) = \boldsymbol{Y}_k^\top Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + \Theta^\top \Phi(\bar{\boldsymbol{Z}}_{k,k-N+1}) \tag{5.48}$$

The value function parameters are updated in the policy evaluation step by generating a temporal difference (TD) error as:

$$\begin{aligned} e_k = \boldsymbol{Y}_k^\top Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k - \Theta_{k+1}^\top \big(\Phi(\bar{\boldsymbol{Z}}_{k-1,k-N}) \\ -\Phi(\bar{\boldsymbol{Z}}_{k,k-N+1})\big) \end{aligned} \tag{5.49}$$

with $\big(\Phi(\bar{\boldsymbol{Z}}_{k-1,k-N}) - \Phi(\bar{\boldsymbol{Z}}_{k,k-N+1})\big)$ as the regressor vector. Data from multiple time steps can then be obtained to determine the least squares solution for the value function parameters in a batch least squares procedure. Alternatively, standard recursive parameter estimation techniques such as the recursive least squares (RLS) can be run till convergence to determine the best fit for the parameters that minimise the generated TD error.

For convergence of the parameter estimates, it is required that the regressor vector be linearly independent over time and satisfies a persistence of excitation (PE) condition given as $\alpha I \leq \sum_{i=k}^{k+M} \Gamma_i \Gamma_i^\top \leq bI \ \forall i$ where Γ is the regressor vector and with $M > 0, a > 0, b > 0$ [19]. In the RL framework, this is achieved by adding an exploration signal ϵ to the control inputs as $\tilde{\boldsymbol{u}}_k = \pi(\boldsymbol{X}_k) + \epsilon$. Following the update of the value function parameters, the policy for the control input is updated using (5.46) in the policy improvement step. The described OPFB tracking control strategy is represented schematically in Figure 5.1 where the RL framework continually adapts the control parameters of an ARMA controller to optimal values, subject to the unknown or varying system dynamics. Algorithm 5.1 describes the OPFB RL tracking with integral control using PI.

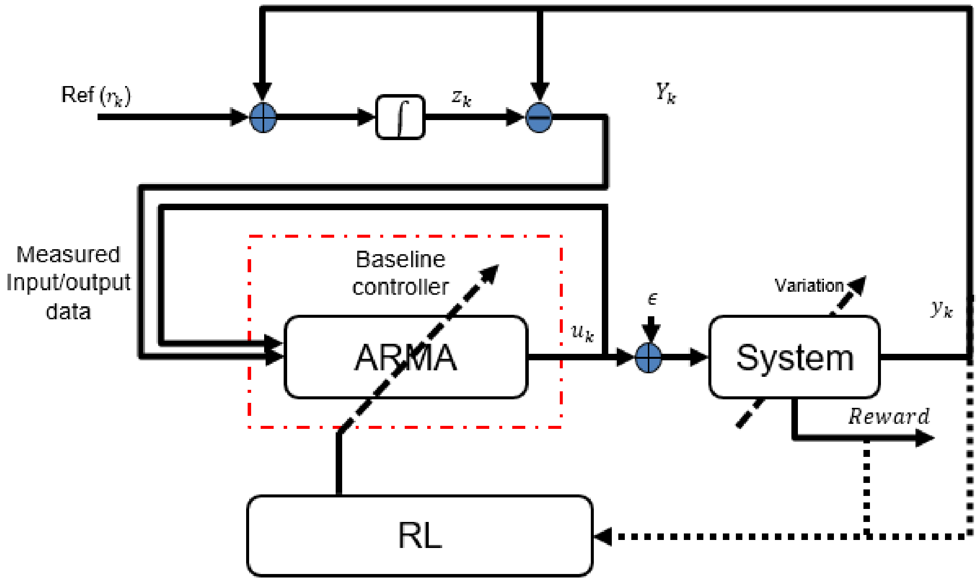


Figure 5.1: Block diagram of the output-feedback tracking with integral control using reinforcement learning that continually updates the parameters of an auto-regressive moving-average (ARMA) controller.

Remarks

- Since the OPFB method uses only the input/output data without any state measurements, the introduction of ϵ to satisfy the PE condition introduces bias in the parameter estimates. It was argued in [123] that using a discount factor in the cost decays the bias and the effects of improper initial conditions.

- Zero steady-state tracking error is however guaranteed with or without the use of the discount factor through the regulation of the augmented integral states to zero.

5.3 Simulation of the condition-based output feedback RL tracking control framework

The proposed OPFB tracking with integral control is demonstrated on the 2-state system of Equation (3.16) with an initially unstable dynamics for which both a baseline model-based and the proposed online RL tracking using augmented formulation with integral control solutions have been provided respectively in Sec-

Algorithm 5.1 OPFB RL tracking with integral control

```
Initialise $V(\boldsymbol{X}) \approx \Theta_k^\top \Phi(\cdot)$ at $k = 0$ for some stabilising initial control policy $\pi_0(\boldsymbol{X})$ and do till convergence: **Value function update step**
for $j = 0$ till parameter convergence do
    At $\boldsymbol{X}_j$, compute the control input $\tilde{\boldsymbol{u}}_j$ with exploration signal $\epsilon$ as $\tilde{\boldsymbol{u}}_j = \pi(\boldsymbol{X}_j) + \epsilon$.
    Compute the least squares solution for $\Theta_{j+1}$ using input/output measurements $\bar{\tilde{\boldsymbol{u}}}_{j,j-N}$ and $\bar{\boldsymbol{Y}}_{j,j-N}$ as:
        $\Theta_{c,j+1}^\top\big(\Phi(\bar{\boldsymbol{Z}}_{j-1,j-N}) - \Phi(\bar{\boldsymbol{Z}}_{j,j-N+1})\big) = \boldsymbol{Y}_j^\top Q \boldsymbol{Y}_j + \tilde{\boldsymbol{u}}_j^\top R \tilde{\boldsymbol{u}}_j$
    $j = j + 1$.
end for
        **Policy update step**
Require: Set $\Theta_{k+1} = \Theta_{j+1}$
Update the control policy using partitioning (5.43) as:
    $\pi_{k+1}(\boldsymbol{X}) = \arg\min_{\tilde{\boldsymbol{u}}} \big(\boldsymbol{Y}_k^\top Q \boldsymbol{Y}_k + \tilde{\boldsymbol{u}}_k^\top R \tilde{\boldsymbol{u}}_k + \Theta_{k+1}^\top \Phi(\bar{\boldsymbol{Z}}_{k,k-N+1})\big)$
Increment time step $k = k + 1$.
```

tions 4.1.1 and 4.3.1. Using Euler's discretisation with a sampling time of $t_s = 0.5s$, the discrete-time dynamics become:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \begin{bmatrix} 1.21 & 1.53 \\ 1.68 & 3.27 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 1.46 \\ 2.09 \end{bmatrix} \boldsymbol{u}_k \\ y_k &= \begin{bmatrix} 1 & 1 \end{bmatrix} \boldsymbol{x}_k \end{aligned} \tag{5.50}$$

It is assumed that measurements of the state variables $\boldsymbol{x}$ are unavailable and only measurements of the input/output data are obtained at the discrete time steps k. The tracking control problem is then to track a time-varying step reference input from any finite initial condition y_0 using only the input/output measurements. Using the proposed augmented formulation with integral control, an augmented system using (5.28) and (5.30) is thus formed as:

$$\begin{aligned} \boldsymbol{X}_{k+1} &= \underbrace{\begin{bmatrix} 1.21 & 1.53 & 0 \\ 1.68 & 3.27 & 0 \\ -.5 & -.5 & 1 \end{bmatrix}}_{A_1} \boldsymbol{X}_k + \underbrace{\begin{bmatrix} 1.46 \\ 2.09 \\ 0 \end{bmatrix}}_{B_1} \tilde{\boldsymbol{u}}_k \\ Y_k &= \underbrace{\begin{bmatrix} 1 & 1 & 1 \end{bmatrix}}_{C_1} \boldsymbol{X}_k \end{aligned} \tag{5.51}$$

Parameters for the tracking cost (5.29) are considered as $Q_x = C_1^\top C_1$ and $R = 1$. With $n = 2$ states and $p = 1$ output, the observability index of the augmented system is computed as $\mathcal{O}_I = n + p = 3$. In order to make the observability matrix V_{N_1} to be full rank, N is selected as the observability index $N = \mathcal{O}_I = 3$.

5.3.1 Model-based OPFB solution using the augmented formulation with integral control

As benchmark solutions, knowledge of the system dynamics were initially assumed to be known and allowing the OPFB solution in Section 5.2.2 to be used to compute the optimal value of the cost. P_1 is computed from (4.49) as:

$$P_1 = \begin{bmatrix} 3.24 & 3.61 & -2.05 \\ 3.61 & 4.36 & -1.92 \\ -2.05 & -1.92 & 6.84 \end{bmatrix} \tag{5.52}$$

The controllability, observability and Toeplitz matrices are computed from Section 5.2.2.1 as:

$$\begin{aligned} U_{N_1} &= \begin{bmatrix} 1.46 & 4.96 & 20.20 \\ 2.09 & 9.29 & 38.74 \\ 0.00 & -1.78 & -8.90 \end{bmatrix} \\ V_{N_1} &= \begin{bmatrix} 9.62 & 17.22 & 1.00 \\ 2.39 & 4.30 & 1.00 \\ 1.00 & 1.00 & 1.00 \end{bmatrix} \\ T_{N_1} &= \begin{bmatrix} 0.00 & 3.55 & 12.48 \\ 0.00 & 0.00 & 3.55 \\ 0.00 & 0.00 & 0.00 \end{bmatrix} \end{aligned} \tag{5.53}$$

Therefore,

$$\begin{aligned} M_y = M = A_1^N V_{N_1}^{+} &= \begin{bmatrix} 2.44 & -3.36 & 0.92 \\ 4.65 & -6.19 & 1.54 \\ -1.60 & 3.64 & -1.04 \end{bmatrix} \\ M_u = (U_{N_1} - M T_{N_1}) &= \begin{bmatrix} 1.46 & -3.72 & 1.63 \\ 2.09 & -7.20 & 2.75 \\ 0.00 & 3.91 & -1.86 \end{bmatrix} \end{aligned} \tag{5.54}$$

The optimal value in terms of matrix $\bar{P}_1^*$ is thus computed using (5.40) as:

$$\bar{P}_1^* = 10^2 \times \begin{bmatrix} .48 & -1.77 & .73 & 1.08 & -1.56 & .41 \\ -1.77 & 7.36 & -3.08 & -4.33 & 6.54 & -1.73 \\ .73 & -3.08 & 1.29 & 1.80 & -2.74 & .73 \\ 1.08 & -4.33 & 1.80 & 2.58 & -3.84 & 1.01 \\ -1.56 & 6.54 & -2.74 & -3.84 & 5.81 & -1.54 \\ .41 & -1.73 & .73 & 1.01 & -1.54 & .41 \end{bmatrix} \tag{5.55}$$

with ARMA gain $K^* = [3.61, \ -1.48, \ -2.21, \ 3.18, \ -0.83]$.

5.3.2 Model-free RL OPFB solution using the augmented formulation with integral control

However, in practice the system dynamics maybe unknown or time varying. The proposed RL framework for the OPFB tracking control is then used to compute the optimal value and controller gains online. The value function is approximated according to (5.47) with $\Theta^\top = stk(\bar{P}) \in \mathbb{R}^{21}$ and $\Phi(\bar{Z}_{k-1,k-N}) = \bar{Z}_{k-1,k-N} \otimes \bar{Z}_{k-1,k-N}$ where $stk(\cdot)$ is the column stacking operator and $\otimes$ is the Kronecker product with the redundant quadratic terms combined. Using Algorithm 5.1, the value function parameters converged to:

$$\hat{\bar{P}}_1 = 10^2 \times \begin{bmatrix} .46 & -1.77 & .71 & 1.05 & -1.56 & .40 \\ -1.77 & 7.45 & -3.02 & -4.33 & 6.57 & -1.70 \\ .71 & -3.02 & 1.23 & 1.75 & -2.67 & .71 \\ 1.05 & -4.33 & 1.75 & 2.53 & -3.81 & .99 \\ -1.56 & 6.57 & -2.67 & -3.81 & 5.80 & -1.51 \\ .40 & -1.70 & .71 & .99 & -1.51 & .39 \end{bmatrix} \tag{5.56}$$

with ARMA gain $K_{RL}^* = [3.78, \ -1.52, \ -2.25, \ 3.32, \ -0.86]$.

Figure 5.2 shows the performance of the ARMA controller given step reference inputs after k training steps while Figure 5.3 shows norm of the difference between the model-based optimal OPFB controller parameters and the computed RL OPFB parameters.

Since the general OPFB RL methods use only the input/output data without any state measurements, the use of ϵ to satisfy the PE condition introduces bias in the parameter estimates as noted in [123]. However, through the regulation

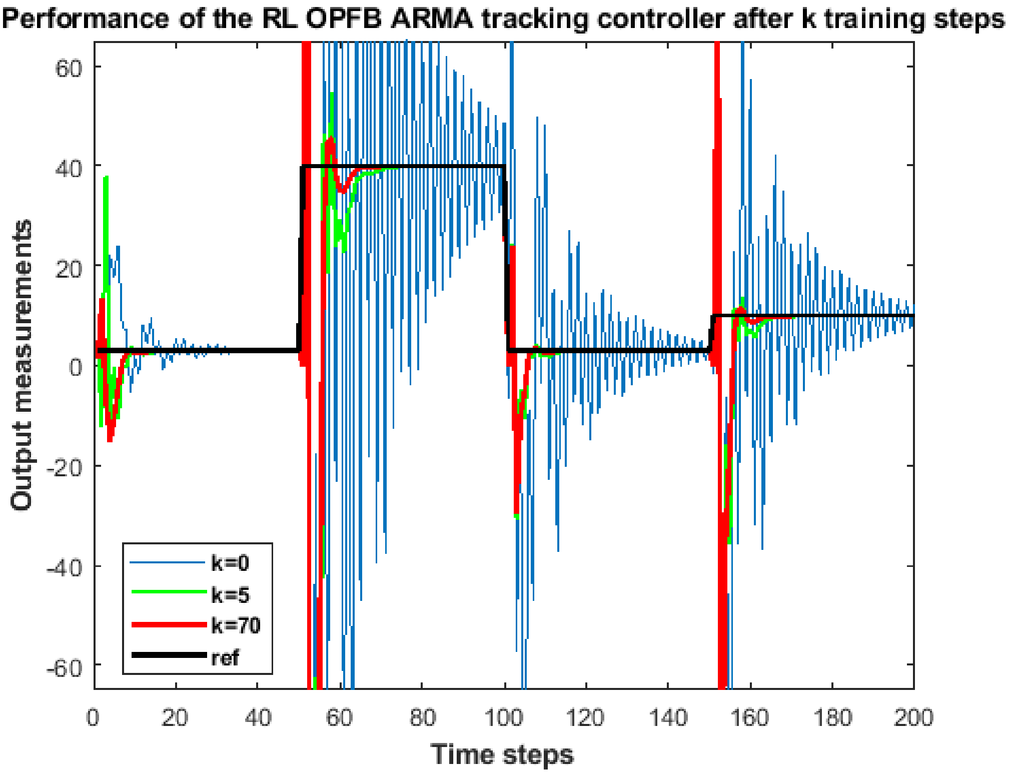


Figure 5.2: Performance of the reinforcement learning output-feedback auto-regressive moving-average tracking controller after k training steps given step reference inputs.

of the augmented integral states, the proposed approach eliminates zero steady-state tracking error and does not impose any restrictions on the reference model dynamics. Figure 5.4 compares the tracking performance and error from using the converged online OPFB RL gains with that of the model-based optimal gains.

Summary

This chapter has proposed and demonstrated an OPFB tracking with integral control using RL for systems with unknown or varying dynamics. In contrast to existing OPFB tracking methods, this framework does not impose any restrictive assumptions on the reference model dynamics or discounted costs, and guarantees zero steady-state tracking error. Simulation results showed that the resulting auto-regressive moving-average (ARMA) tracking controller achieve zero steady-state tracking error on convergence of the RL adaptations.

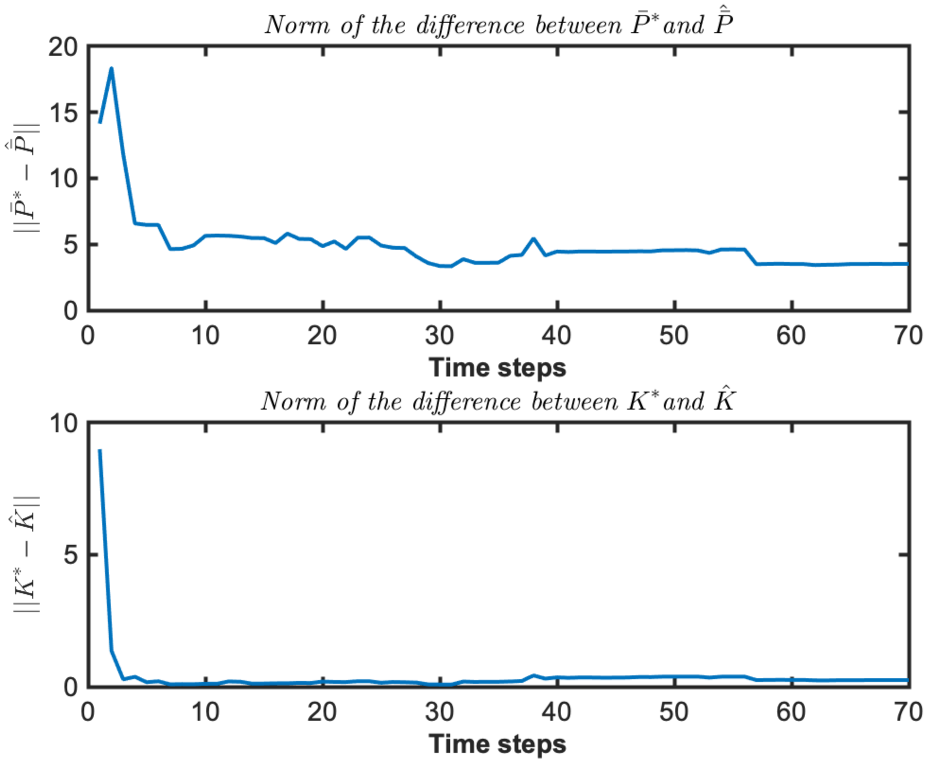


Figure 5.3: Norm of the difference between the model-based optimal output-feedback tracking controller parameters and the computed reinforcement learning output-feedback tracking parameters using Algorithm 5.1.

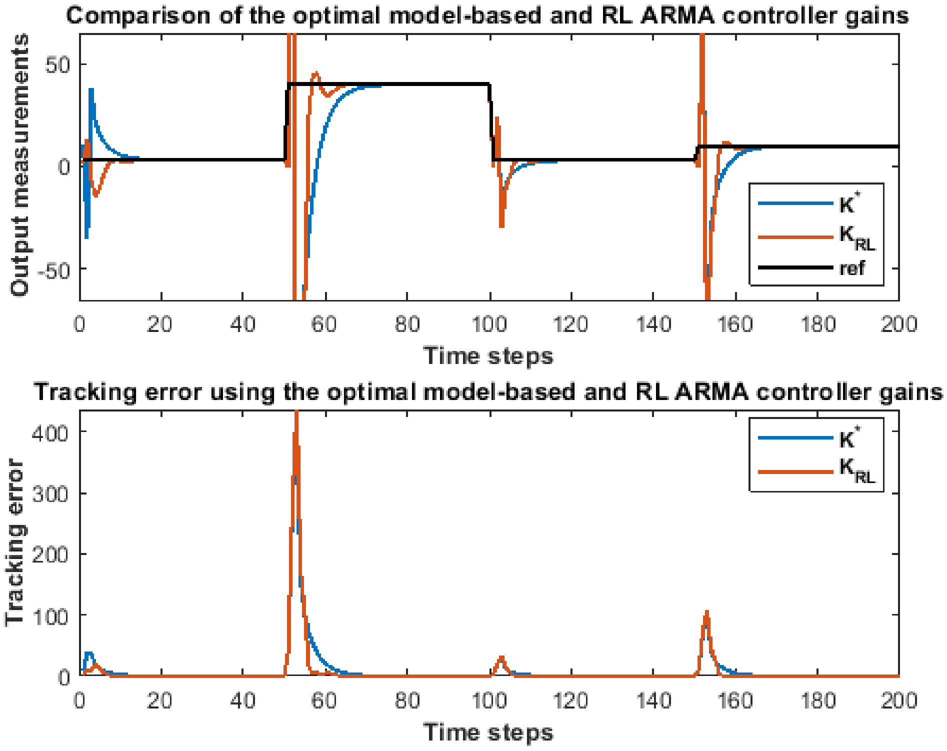


Figure 5.4: Comparison of the optimal model-based auto-regressive moving-average (ARMA) controller and the computed reinforcement learning ARMA gains.

Chapter 6

Power management optimisation for hybrid systems using condition-based reinforcement learning

Previous chapters have shown the development of reinforcement learning (RL) frameworks for the control of time-varying dynamical systems that can effectively accommodate the system variations which are difficult to model and deal with via conventional model-based methods. Hybrid systems are shown in this chapter to benefit from the developed reinforcement learning frameworks which are extended to the power management optimisation problem. Current methods for the power management optimisation problem are conservative and unable to fully account for the variations in the hybrid systems due to changes in the health and operational conditions. These conservative schemes result in less efficient use of the available hybrid power sources, increasing the overall system costs and heightening the risk of failure due to the variations.

Consequently, this chapter presents the development of online condition-based RL frameworks for the power management of hybrid propulsion and electrical power generation systems to compensate for the gradual system variations and learn online the optimal power management strategy between the hybrid power sources. The proposed condition-based power management RL scheme is able to compensate for modelling uncertainties and the gradual system variations resulting from degradation by adapting a model of the performance function online using the observed system measurements as reinforcement signals, and given the future load predictions. The strategies and results discussed in this chapter are based on the author's work in [126]. A summary of the main contributions presented in this chapter are as follows:

- Current state-of-the-art power management optimisation strategies are either based on pre-defined rule based power schedules, exhaustive model-based optimisation or dynamic programming approach [127], [128], [129]. These strategies assume accurate system models for the power management optimisation and are therefore limited in their ability to account for system variations. In contrast to these approaches, this chapter proposes and demonstrates a new online learning scheme based on RL and adaptive dynamic programming (ADP) that is able to compensate for the gradual system variations due to changes in the system health or operating conditions.
- A condition-based online RL framework is proposed, which is composed of a planning/scheduling phase to determine the power management control sequence using dynamic programming (DP) and an iterative adaptation of the system performance functions using Q-learning in a receding horizon manner.

In the following, Section 6.1 introduces the hybrid electric systems and formulates the power management optimisation problem. An overview of the current power management optimisation strategies is given in Section 6.2 along with the proposed RL ADP strategy that overcomes the limitations of the existing strategies by compensating for modelling uncertainties and gradual system variations. Lastly, Section 6.3 provides a representative simulation case study using the proposed power management strategy for the condition-based control of an autonomous hybrid system which shows improved system performance as compared with a conventional dynamic programming power management approach.

6.1 Power management of hybrid electric systems

Hybrid electric systems such as those deployed on unmanned aerial vehicle (UAV) often have architectures which support two or more power sources [130]. The power sources typically consist of joint propulsion and electrical generation systems such as the gas turbine engines (GTEs), and one or more energy storage devices e.g fuel cells, supercapacitors and batteries [129]. With limited energy resources on-board the hybrid systems, power management strategies have been identified as key enabling technologies to support enhanced capabilities of the systems such as longer operational times and increased endurance [130], [131].

The enhanced capabilities are envisaged to be associated with increased power requirements, mission risks and overall system costs. It is therefore the aim of the power management strategies to reduce the risks and overall system costs whilst providing an effective way to support the system power requirements.

The operation of an autonomous vehicle can be divided into phases, for example a car or aircraft may have pre-planned routes or missions (e.g hill climbing or aircraft radar sweeps) associated with varying power demands [130]. There is an energy interdependency between the operation phases as the power drawn from a source for a duration of a phase may become unavailable for the remaining phases. This is the case for the energy storage devices where the available power for a phase is dependent on previous charge/discharge energy cycles at the other phases. Current industry-standard approaches for the power management are therefore based on pre-defined rule based power schedules between the multiple power sources [132]. These approaches follow a series of *if-then* rules designed for the worst-case peak power requirements. As such, they are usually conservative and unable to adapt to dynamic changes in the systems. Over the years, research trends have favoured optimisation based power management approaches to optimise the desired power requirements and constraints of the hybrid systems [127], [128].

In [133], the hybrid system power management was formulated as a mixed-integer nonlinear multi-objective optimisation problem and solved using a differential evolutionary fuzzy scheme. The proposed solution is however non-deterministic and does not provide any solution guarantees to be suited for real-time implementation. Consequently, an intelligent power management system (PMS) that guarantees at least a feasible solution was proposed in [131] using a three level optimisation strategy. Both approaches are, however, unable to account for unmodelled variations in the system resulting from degradation or changes in the system operating conditions. Furthermore, the energy interdependency between the sources is considered in a heuristic rule based manner that is suboptimal in both schemes.

Other approaches have considered the DP technique which is well suited to handle the energy interdependency by solving the optimisation problem as a sequence of operations [41]. The DP technique uses the Bellman's optimality principle to limit the optimisation search to the potentially optimal trajectories as discussed in Section 2.2.1 and on which RL and ADP strategies are based on. In [134], DP was used to develop a hydroelectric scheduling technique between thermal and hydro power sources to minimise the system generation cost while satisfying the system load requirements. Likewise, [135] proposed an optimal dispatch of direct load control using DP to minimise the system production cost. Related works on power management optimisation using DP include [136], [137] for optimal charge/discharge of energy storage devices; [127], [129] and [138] for optimal energy management for hybrid electric vehicles. All of these works depend on accurate system models and are therefore limited in their ability to account for system variations and modelling uncertainties.

Extension of the DP techniques to provide adaptation and self-learning capabilities are enabled using frameworks based on RL and ADP [37], [41], [47], [50]. Using ADP, an adaptive power management scheme was developed for residential load management in both [95] and [96]. Both of these approaches applied a heuristic approach in the online management scheme by limiting the control inputs to one of three choices as *charge, discharge* and *idle*, greatly reducing the optimality of the solutions. In [97], a dual Q-learning scheme was proposed as an extension to the residential load management optimisation and considers the optimisation horizon over future load predictions. This scheme is however restricted to problems involving repeated known cycles over the predicted load horizon and system costs as obtaining a function approximation over arbitrary load horizons is infeasible.

In contrast to the above approaches, this chapter proposes and demonstrates a new learning scheme based on reinforcement learning and adaptive dynamic programming (RL-ADP) that computes optimal power control sequences online given future load predictions, does not assume repeated known load cycles, and is able to compensate for both modelling uncertainties and gradual variability due to changes in the system health or operating conditions. The system learns by using reinforcement signals in the form of the system measurements to adapt the system performance function, which is then used to determine the best power control strategy online in a receding horizon manner. The next section introduces the hybrid propulsion and electrical generation system under consideration.

6.1.1 Hybrid propulsion and electrical power generation systems

An autonomous hybrid electric system consisting of a GTE propulsion system and an energy storage device in form of a battery is considered. The propulsion system provides the necessary thrust (FN) needed by the system whilst also providing electrical power to the on-board system loads. Electrical power is generated from the propulsion system through two sets of generators coupled to the rotating engine core and propeller shafts respectively as P_{core} and P_{prop} as shown in Figure 6.1. This additional load on the propulsion system results in higher fuel burn at peak load requirements. A hybrid battery integration therefore promotes feasibility of power scheduling for efficient system operation and increased system capability.

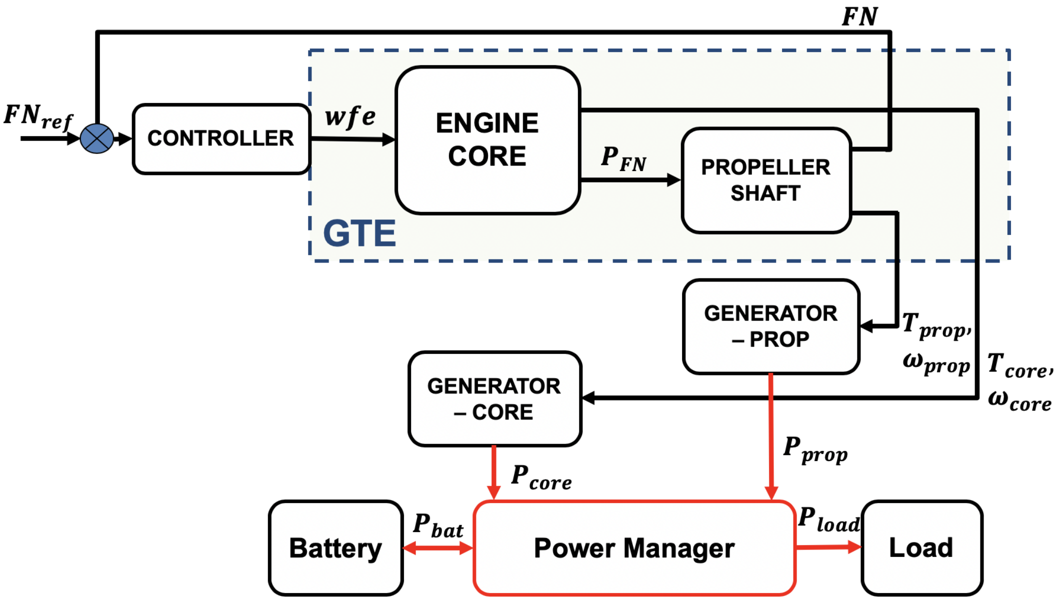


Figure 6.1: Block diagram of a hybrid electric system consisting of a gas turbine engine (GTE) with battery integration. The GTE produces thrust (FN) for a given amount of fuel flow (wfe) whilst also providing electric power via two sets of generators coupled to both the propeller and core shafts.

This chapter focuses on computing the best power delivery strategy by the power manager that optimises some desired performance/efficiency cost and makes the following assumptions:

Assumptions

1. It is assumed that the GTE is pre-stabilised in a thrust control loop with an existing tracking controller that computes the required amount of fuel flow (wfe) needed to generate the needed thrust (FN) and engine power given a thrust reference demand (FN_{ref}).

2. Given gradual variations in the GTE dynamics due to degradation or changing operating conditions, the tracking controller parameters can be com-

pensated for using the proposed condition-based online RL tracking control frameworks of Chapters 4 and 5.

A formulation for the power management optimisation problem is thus given in the next section.

6.1.2 Problem formulation for the power management optimisation

The governing power equation for the power management system is considered as:

$$P_{eng} = P_{FN} + P_{prop} + P_{core} \tag{6.1}$$

where P_{eng} is the total engine power from the GTE, P_{FN} is the propulsive power needed for thrust generation while P_{prop} and P_{core} are respectively the electrical power from the propeller and core shafts. For the load demand side, the power balance equation is given by:

$$P_{prop} + P_{core} = P_{load} - P_{bat} \tag{6.2}$$

where P_{load} is the required load power and P_{bat} is the battery power output. $P_{bat} > 0$ indicates that the battery is discharging, and charging when $P_{bat} < 0$. Based on Assumption 1, the thrust and load power requirements are always satisfied by the thrust control loop. Thus combining (6.1) and (6.2) gives:

$$'P_{eng} = P_{load} - P_{bat} \tag{6.3}$$

where $'P_{eng} = P_{eng} - P_{FN}$. Figure 6.2 shows a sample power demand profile for a hybrid electric system and the discrete time steps k considered for the optimisation. The change in energy between the time steps k is defined as:

$$\Delta E_{k+1} := {'P_{eng,k}} \Delta t = (P_{load,k} - P_{bat,k}) \Delta t \tag{6.4}$$

The dynamics for the battery state of charge (SOC) consistent with [95] and [97] is given as:

$$SOC_{k+1} = SOC_k - sign(P_{bat,k}) \cdot \eta(P_{bat,k}) \Delta t \tag{6.5}$$

where $sign(P_{bat})$ indicates discharging (+) or charging (-) of the battery while $\eta(P_{bat})$ gives the battery efficiency. The power management optimisation problem therefore aims to find the control strategy for P_{bat} that will optimise a desired performance cost for a given load profile P_{load}. The state equations are thus defined as follows:

$$x_{k+1} = F(x_k, u_k) = \begin{bmatrix} (P_{load,k} - u_k)\Delta t \\ x_{2,k} - sign(u_k) \cdot \eta(u_k)\Delta t \end{bmatrix}$$

$$\textit{subject to: } x \in \mathbb{X}, \quad u \in \mathbb{U} \tag{6.6}$$

where $x_k = \begin{bmatrix} \Delta E_k & SOC_k \end{bmatrix}^\top$, $u_k = P_{bat,k}$ and $\mathbb{X}, \mathbb{U}$ are sets of constraints on the state and input respectively. The desired cost to be optimised at the discrete time steps k is given as:

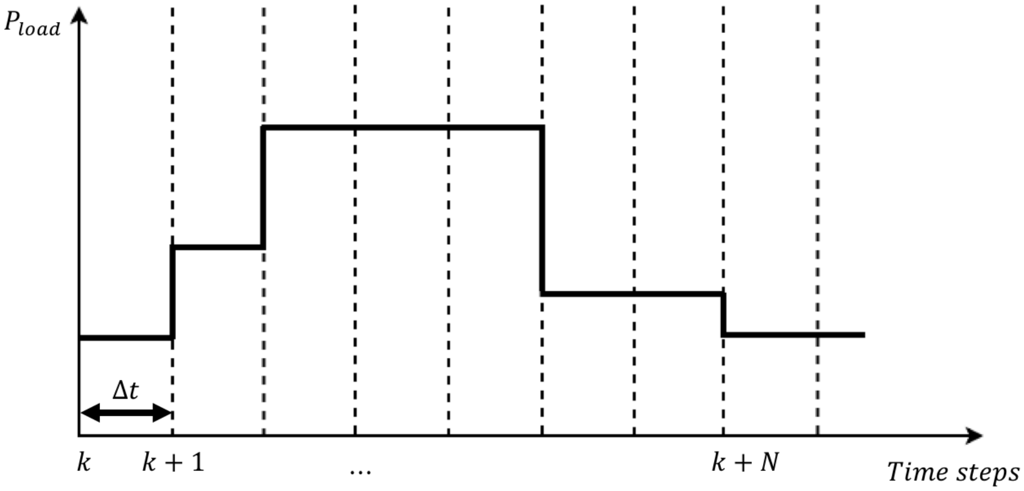


Figure 6.2: Sample operational phases and power requirements for the autonomous hybrid electric system in time steps $k, k+1, \cdots, k+N$.

$$Q(x_k, u_k) = \sum_{n=k}^{N} \gamma^{n-k} R(x_n, u_n) \tag{6.7}$$

where $R(x_k, u_k) = \eta_{GTE} = \overline{TSFC(x_{1,k})}^2 + \overline{u}_k^2$ is the system efficiency function which is assumed to be directly measurable with $\overline{TSFC(x_{1,k})}$ as the normalised GTE fuel consumption and $\overline{u}_k$ as the normalised battery power control input; N is the length of the load demand profile and $\gamma \in [0, 1]$ is a forgetting factor. Analytical solution to the formulated optimisation problem will require knowledge of the system dynamics and reward functions as discussed in Section 2.2.1.2 using calculus of variations. Furthermore, the given nonlinear state and efficiency functions of (6.6) and (6.7) will result in the nonlinear Hamilton-Jacobi Bellman (HJB) equations which are known to be difficult and often impossible to solve analytically

[9]. Candidate state-of-the-art approaches to the formulated power management optimisation problem are provided in the next section.

6.2 Candidate power management optimisation strategies

Industry standard approaches for solving the power management optimisation problem are typically based on rule-based power schedules, optimised for the worst-case peak power requirements [132] or on exhaustive optimisation-based methods [131]. Consequently, the power management optimisation controllers can be classified according to Salmasi [139] and Wirasingha and Emadi [140] as follows:

1. Rule-based controllers: make use of pre-defined sets of rules and logics based on the system power requirements and efficiency charts [140]. These can be further classified into deterministic rule-based methods that make use of state-machine models and transition logics [141], [142]; and fuzzy rule-based methods that provide improved fuel economy over the simple rule-based methods for time-varying nonlinear systems [143]. Limitations of the rule-based methods include:

    - Dependence on known system dynamics and deterministic modes of operations limits the rule-based approaches in compensating for unmodelled variations in the system dynamics and power requirements.
    - Considerations for performance optimisation to include more design parameters such as emissions and life lead to increasingly complex set of rules which may become intractable.

    Consequently, researchers have considered other power management strategies based on mathematical optimisation methods that are able to optimise over large design parameter space [127].

2. Optimisation-based controllers: make use of mathematical optimisation algorithms such as genetic algorithms (GA) or exhaustive search methods to compute the best power management strategy for a given operational cycle using models/functions of the system efficiencies [139]. These can be further classified into global or acausal optimisation methods that make use of historical data or offline system models/functions for the optimisation [133], [138], [144], [145], [146]; and real-time or causal optimisation methods that are able to adapt to the systems variations using real-time data [147]. The optimisation-based approaches, however, may become difficult to solve

analytically for cases involving nonlinear and non-convex optimisation problems. As discussed in Section 2.2.1, dynamic programming provides a systematic approach for solving complex optimisation problems and has been identified as a powerful tool for providing globally optimal solutions to the power management optimisation problem whilst able to handle constraints and nonlinearities [138], [139], [146], [147].

6.2.1 Power management optimisation using dynamic programming

DP provides a systematic approach for solving complex optimisation problem and is well suited to handle the energy interdependency of the power management optimisation problem by solving as a sequence of operations [140], [139]. DP considers the recursive form for the cost function of (6.7) as:

$$Q(\boldsymbol{x}_k, u_k) = R(\boldsymbol{x}_k, u_k) + \gamma Q(\boldsymbol{x}_{k+1}, u_{k+1}) \tag{6.8}$$

Equation (6.8) is solved at every time step k using the Bellman's principle of optimality as discussed in Section 2.2.1 on dynamic programming. DP assumes that the system model and efficiency functions are known, and discretises the system states into levels with associated cost Q. DP therefore uses the Bellman's principle of optimality to limit the optimisation search to only the optimal trajectories, and starting from a terminal cost $Q(\boldsymbol{x}_N, u_N)$, the optimal power control sequence can be determined as follows:

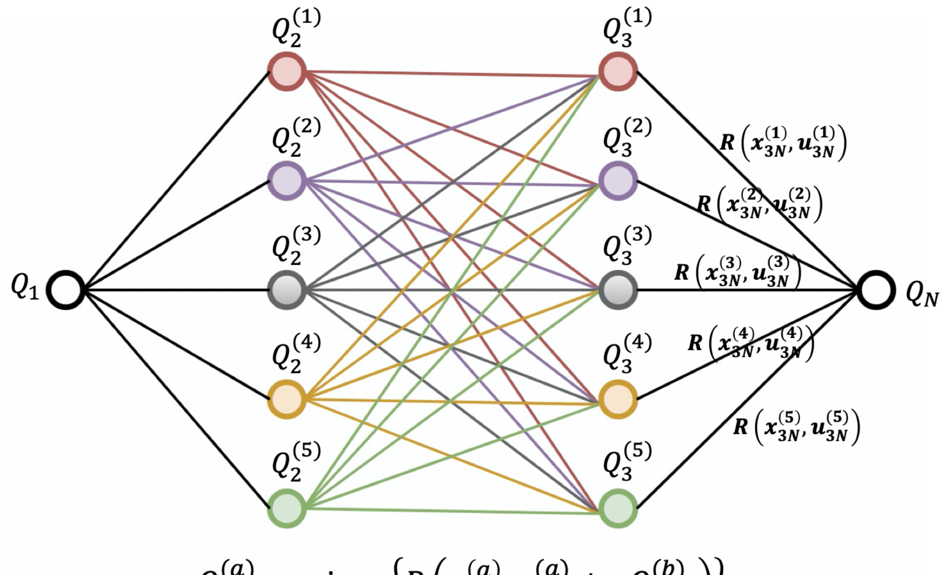
$$\textit{Solve backwards from} \quad n = N - 1 : -1 : k$$

$$\begin{aligned} Q^*(\boldsymbol{x}_k, u_k) &\leftarrow \min_{u_k} \left\{ R(\boldsymbol{x}_k, u_k) + \gamma Q^*(\boldsymbol{x}_{k+1}, u_{k+1}) \right\} \\ \text{subject to: } \boldsymbol{x}_{k+1} = F(\boldsymbol{x}_k, u_k) &= \begin{bmatrix} (P_{load,k} - u_k)\Delta t \\ x_{2,k} - sign(u_k) \cdot \eta(u_k)\Delta t \end{bmatrix} \\ \boldsymbol{x} \in \mathbb{X}, &\quad u \in \mathbb{U} \end{aligned} \tag{6.9}$$

A schematic representation of the DP power management optimisation routine is shown in Figure 6.3.

Remarks

- The problem space for DP is known to increase with increased number of states and actions. This is known as the DP curse of dimensionality as discussed in Section 2.2.1. Although, known to limit its practicality, DP has been


$Q_2^{(1)}$
$Q_3^{(1)}$
$Q_2^{(2)}$
$Q_3^{(2)}$
$Q_2^{(3)}$
$Q_3^{(3)}$
Q_1
Q_N
$Q_2^{(4)}$
$Q_3^{(4)}$
$Q_2^{(5)}$
$Q_3^{(5)}$
$R\left(x_{3N}^{(1)}, u_{3N}^{(1)}\right)$
$R\left(x_{3N}^{(2)}, u_{3N}^{(2)}\right)$
$R\left(x_{3N}^{(3)}, u_{3N}^{(3)}\right)$
$R\left(x_{3N}^{(4)}, u_{3N}^{(4)}\right)$
$R\left(x_{3N}^{(5)}, u_{3N}^{(5)}\right)$

$$Q_k^{(a)} = \min_{u_k}\left\{R\left(x_{ab}^{(a)}, u_{ab}^{(a)} + \gamma Q_{k+1}^{(b)}\right)\right\}$$

where a, b are the discretised energy levels at optimisation stages $k \ and \ k+1$ respectively

Figure 6.3: Schematic representation of the dynamic programming power management routine showing sample discrete energy levels with associated cost Q from which the optimal control sequence can be determined

shown to scale well with problems involving hundreds of states and actions [41] and suited for the power management optimisation problem [140], [139].

- A major drawback of DP however is its dependence on known analytical state and efficiency models (i.e. $F(x,u)$ and $R(x,u)$). For the formulated power management optimisation problem, the state equations, i.e. $F(x,u)$, are given by the system energy requirements and are known from equation (6.6). However, analytical models to accurately describe the changes in the system health or operational conditions are typically unknown. These changes are assumed to reflect in the measured reward signals, i.e. gradual changes in the measured GTE and battery efficiencies. Consequently, the standard DP framework assumes a fixed analytical model for $R(x,u)$ and is unable to cope with variations in the system conditions.

Given the limitations of the standard DP approach to the power management optimisation problem, an online framework based on RL and ADP is therefore proposed to compensate for both modelling uncertainties and gradual variations in the system by recursively solving the sequence of power delivery control decisions using dynamic programming and an iterative adaptation of the system efficiency functions.

6.2.2 Power management optimisation using reinforcement learning and adaptive dynamic programming

Motivated by the Bellman optimality equations, RL-ADP algorithms make use of iterative equations that are known to successively lead to improved policies [42]. The iterative equations involve both value and policy update steps as discussed in Section 2.2.2 and are respectively given as:

$$Q_{k+1}(\boldsymbol{x}_k, u_k) = R(\boldsymbol{x}_k, u_k) + \gamma Q_k(\boldsymbol{x}_{k+1}, u_{k+1}) \tag{6.10}$$

$$u_{k+1} = \arg\min_{u_k} \left(R(\boldsymbol{x}_k, u_k) + \gamma Q_{k+1}(\boldsymbol{x}_{k+1}, u_{k+1})\right) \tag{6.11}$$

These are implemented *forward-in-time* without requiring models of the system. Convergence of the iterative updates has been proven by showing that interleaving (6.10) and (6.11) leads to the contraction map (2.14) and (2.15) associated with the DP [9].

Given a load profile $P_{load,k}|k = 0, 1, \cdots, N$, we wish to solve online the best power delivery strategy (i.e. control sequence $\boldsymbol{U}_N = [u_0, u_1, \cdots, u_N]$) that minimises the desired cost (6.7). Mathematically, this can be formulated as:

$$\begin{aligned} \boldsymbol{U}_N &= \min_{u} Q^*(\boldsymbol{x}_k, u_k) \\ &= \min_{u_k} \Bigg\{ R(\boldsymbol{x}_k, u_k) + \gamma \min_{u_{k+1}} \Bigg\{ R(\boldsymbol{x}_{k+1}, u_{k+1}) + \cdots \\ &+ \gamma \min_{u_{k+j-1}} \left\{ R(\boldsymbol{x}_{k+j-1}, u_{k+j-1}) + \gamma \min_{u_{k+j}} Q^*(\boldsymbol{x}_{k+j}, u_{k+j}) \right\} \Bigg\} \Bigg\} \\ & \textit{for } j = 1, 2, \cdots, N \end{aligned} \tag{6.12}$$

Conventional RL-ADP algorithms however require that the optimal Q-function strictly follows the one-step Bellman optimality equation that explicitly approximates the cost dependence on future load predictions as:

$$Q^*(\boldsymbol{x}_{N-1}, u_{N-1}) = R(\boldsymbol{x}_{N-1}, u_{N-1}) + \gamma \min_{u_N} Q^*(\boldsymbol{x}_N, u_N) \tag{6.13}$$

Clearly, the power management optimisation problem (6.12) involves varying Q-functions due to the dependence of the state variables x on the varying load requirements, P_{load} and does not conform with (6.13). Furthermore, to be able to compensate for unmodelled dynamics and gradual system variations, a novel condition-based approach is therefore to consider the online power management optimisation problem as being composed of:

- A planning/scheduling phase to determine the control sequence $\boldsymbol{U}_N$ using algorithms such as DP which takes into account the future load predictions.

- Iterative learning/adaptation of the system efficiency functions using the system reward measurements to compensate for modelling uncertainties and system variations. This can be achieved by making use of Q-function approximations and temporal difference (TD) error as discussed in Section 3.1.2 as follows:

$$\bar{Q}(\boldsymbol{x}, u) \approx \beta^\top \Psi(\boldsymbol{x}, u) \tag{6.14}$$

$$\therefore e_k = R(\boldsymbol{x}_k, u_k) + \gamma \beta_k^\top \Psi(\boldsymbol{x}_{k+1}, u_{k+1}) - \beta_k^\top \Psi(\boldsymbol{x}_k, u_k) \tag{6.15}$$

  where $\bar{Q}(\boldsymbol{x}, u)$ is the approximated system efficiency function, $\Psi(\boldsymbol{x}, u)$ is a set of basis function and β are the function weights. Equation (6.15) is solved for $e_k = 0$ at each time step to yield the least squares approximation to the TD error equation. This way, only the measured data (i.e $R(\boldsymbol{x}_k, u_k)$, $\boldsymbol{x}_{k+1}$ and u_k) are used to compute the optimal control inputs without knowledge of the system models or variations.

Adaptation of the system efficiency function is achieved by defining a cost E_k based on the TD error (6.15) as follows:

$$E_k = \frac{1}{2} e_k^2 \tag{6.16}$$

Therefore:

$$\begin{aligned} \beta_{k+1} &= \beta_k - l_c \frac{\partial E_k}{\partial \beta_k} \\ &= \beta_k - l_c \left[\frac{\partial E_k}{\partial \bar{Q}(\boldsymbol{x}_k, u_k)} \frac{\partial \bar{Q}(\boldsymbol{x}_k, u_k)}{\partial \beta_k} \right] \end{aligned} \tag{6.17}$$

where $l_c > 0$ is the learning rate. The adapted system efficiency function is then used to generate reward signals and used in an online planning/scheduling scheme to determine the optimal control sequence $\boldsymbol{U}_N$. Following the computed control sequence, only the first control input is applied to the system online in a receding horizon manner, and the process is repeated. Algorithm 6.1 gives the template for the proposed procedure.

Algorithm 6.1 Online RL-ADP framework for power management optimisation

```
1: Initialise $\bar{Q}(\boldsymbol{x}, u) \approx \beta_0^\top \Psi(\boldsymbol{x}, u)$ and obtain the control sequence $\boldsymbol{U}_N = [u_0, u_1, \cdots, u_N]$ from dynamic programming routine of (6.9) with $R(\boldsymbol{x}_n, u_n) = \beta_0^\top \Phi(\boldsymbol{x}_n, u_n) \mid_{n=N:-1:k}$
Online computation: for $k = 0 : N$
2: Apply the first control input $u_k$.
3:     Q-function update step
4: Obtain real-time measurements for the reward signal $R(\boldsymbol{x}_k, u_k)$, the states $\boldsymbol{x}_{k+1}$ and the control input $u_k$.
5: Compute the TD error from (6.15), and adapt the system efficiency function using (6.16) and (6.17).
6:     Online planning/scheduling step
7: Perform online dynamic programming using the updated efficiency function with $R(\boldsymbol{x}_n, u_n) = \beta_{k+1}^\top \Psi(\boldsymbol{x}_n, u_n) \mid_{n=N:-1:k+1}$ and determine the optimal control sequence $\boldsymbol{U}_{k \to N} = [u_{k+1}, u_{k+2}, \cdots, u_N]$.
8: Repeat steps 2 to 5 till $k = N$.
```

Remarks

- Obtaining a Q-function approximation that spans the entire state space in (6.12) may be infeasible with increased number of future load predictions and discrete stages for optimisation. This negates the use of traditional Q-learning algorithms but favours the iterative adaptation of the varying system efficiency function ($\bar{Q}(\boldsymbol{x}, u)$) at each optimisation stage ($n = k$) using the received reward signals as:

$$
\begin{aligned}
\therefore \bar{Q}(\boldsymbol{x}_k, u_k) \approx \beta_k^\top \Psi(\boldsymbol{x}_k, u_k) &= \sum_{n=k}^{k} \gamma^{n-k} R(\boldsymbol{x}_n, u_n) \\
&= R(\boldsymbol{x}_k, u_k) \qquad (6.18)
\end{aligned}
$$

- Consequently, the adapted system efficiency function ($\bar{Q}(\boldsymbol{x}, u)$) gives the instantaneous reward signals from (6.18) in place of a conventional fixed analytical model for $R(\boldsymbol{x}, u)$ and is used in a standard online DP routine of (6.9) which then converges to the optimal Q-function ($Q^*(\boldsymbol{x}, u)$).

6.3 Simulation studies

The proposed RL-ADP framework for power management optimisation is demonstrated on a representative autonomous hybrid electric system model to compensate for both modelling uncertainties and variations in the system efficiency. The electrical power from the GTE and battery are constrained between $30KW \leq {}'P_{eng} \leq 150KW$ and $-60KW \leq P_{bat} \leq 60KW$ respectively i.e. representing the constraint sets $\mathbb{X}, \mathbb{U}$, while the battery SOC is expressed as a percentage between $0 - 100\%$. The reward signal is assumed given by the GTE efficiency, η_{GTE} which is the measured pounds of fuel flow per hour per unit thrust. The intervals between the discrete time steps k, i.e Δt for the optimisation are considered to be fixed and determined by changes in the load demand as shown in Figure 6.2.

Given a load profile $P_{load,k}$, the aim of the power management optimisation framework is then to determine the best power control strategy that optimises the cost function of (6.7) subject to variations in the systems.

6.3.1 Algorithm implementation

A preliminary test was first carried out to determine suitable basis function that can model the search space complexities of the system efficiency function for the power management optimisation problem involving the different load demands and the system energy constraints. The test data consist of randomly sampled ${}'P_{eng}$, P_{bat} and SOC levels with the reward signals as the measured η_{GTE} from the system, penalised with large values for violations of the system energy constraints. Approximation of the system efficiency function using the test data with some choice of basis function is then carried out and the results shown below:

Table 6.1: Cross-validated mean-squared error (MSE)

Model	**Polynomial**	**2-layer neural network**		
Complexity	2^{nd} order	5 hidden	20 hidden	50 hidden
MSE	206.46	0.44	0.26	0.18

Results from Table 6.1 indicate that the approximation function is more complex than a second order and use of higher order polynomials may lead to overfitting. Neural networks however offer better approximation to cope with the nonlinearities with considerations for the trade-off between model complexity and the cross-validated MSE. Consequently, a 2-layer neural network for the system efficiency function is trained as follows:

$$\bar{Q}(x,u) \approx \beta^{(2)^\top} \Psi(x,u) \tag{6.19}$$

where

$$\begin{aligned}\Psi(\boldsymbol{x},u) = \Psi(\underline{\boldsymbol{x}}) &= \begin{bmatrix}1 & \frac{e^{\beta^{(1)^\top}\underline{\boldsymbol{x}}}-e^{-\beta^{(1)^\top}\underline{\boldsymbol{x}}}}{e^{\beta^{(1)^\top}\underline{\boldsymbol{x}}}+e^{-\beta^{(1)^\top}\underline{\boldsymbol{x}}}}\end{bmatrix}\\ &= \begin{bmatrix}1 & \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}\end{bmatrix}\\ &= \begin{bmatrix}1 & \boldsymbol{a}\end{bmatrix}\end{aligned} \tag{6.20}$$

$\underline{\boldsymbol{x}} = \begin{bmatrix}1 & x_1 & x_2 & u\end{bmatrix}^\top \in \mathbb{R}^{1\times 4}$, $\boldsymbol{z} = \beta^{(1)^\top}\underline{\boldsymbol{x}} \in \mathbb{R}^{n_h \times 1}$, $\boldsymbol{a} = tanh(\boldsymbol{z}) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \in \mathbb{R}^{n_h \times 1}$, n_h is the number of hidden nodes, and $\beta^{(1)} \in \mathbb{R}^{4\times n_h}$, $\beta^{(2)} \in \mathbb{R}^{n_{h+1}\times 1}$ are respectively the inner and outer layer weights. The update sequence for the function weights follows from (6.16) and (6.17):

Outer layer

$$\beta^{(2)}_{k+1} = \beta^{(2)}_{k} - l_c\left[\frac{\partial E_k}{\partial \bar{Q}(\boldsymbol{x}_k,u_k)}\frac{\partial \bar{Q}(\boldsymbol{x}_k,u_k)}{\partial \beta^{(2)}_k}\right] \tag{6.21}$$

where $\frac{\partial E_k}{\partial \bar{Q}(\boldsymbol{x}_k,u_k)} = \gamma e_k$ and $\frac{\partial \bar{Q}(\boldsymbol{x}_k,u_k)}{\partial \beta^{(2)}_k} = \Psi(\boldsymbol{x}_k,u_k)$

Inner layer

$$\beta^{(1)}_{k+1} = \beta^{(1)}_{k} - l_c\left[\frac{\partial E_k}{\partial \bar{Q}(\boldsymbol{x}_k,u_k)}\frac{\partial \bar{Q}(\boldsymbol{x}_k,u_k)}{\partial \boldsymbol{a}}\frac{\partial \boldsymbol{a}}{\partial z}\frac{\partial z}{\partial \beta^{(1)}_k}\right] \tag{6.22}$$

where $\frac{\partial \bar{Q}(\boldsymbol{x}_k,u_k)}{\partial \boldsymbol{a}} = \sum_{i=2}^{n_h+1}\beta^{(2)}_{(i)}$, $\frac{\partial \boldsymbol{a}}{\partial z} = 1 - tanh(\boldsymbol{z})^2$ and $\frac{\partial z}{\partial \beta^{(1)}_k} = \underline{\boldsymbol{x}}$. The parameters for the neural network implementation are selected as follows: $\gamma = 1$, $n_h = 20$ and $l_c = 0.3e^{-4}$. There are no stability guarantees for this choice of weight update, but strategies to limit divergence such as the use of target networks discussed in [36] proved successful in the provided simulations. Two scenarios are considered to demonstrate the effectiveness of the proposed approach:

6.3.2 Performance of offline power schedule vs proposed RL-ADP algorithm

As discussed in Section 6.2.1, algorithms such as DP can be used to construct offline power schedules for the power management optimisation problem. Typically, these are designed for fixed nominal system models for the worst-case peak power requirements and are usually suboptimal by being unable to adapt to the actual system conditions. A DP algorithm as described in (6.9) was used to compute feasible offline power schedules for the hybrid system and serves as the baseline.

Given the system mismatch and other uncertainties at design time between the nominal and actual (but unknown) GTE efficiency, the computed offline power schedules will be suboptimal and result in reduced system performance. Figure 6.4 and 6.5 show the given load profile and the results from using Algorithm 6.1 compared with the baseline DP power management strategy. Whilst both power management strategies were able to satisfy the system load requirements, Algorithm 6.1 was able to compensate for the system mismatch by using the actual system measurements as reward signals to adapt the system efficiency functions and deliver improved performance as shown by the reduced average fuel consumed during the simulation.

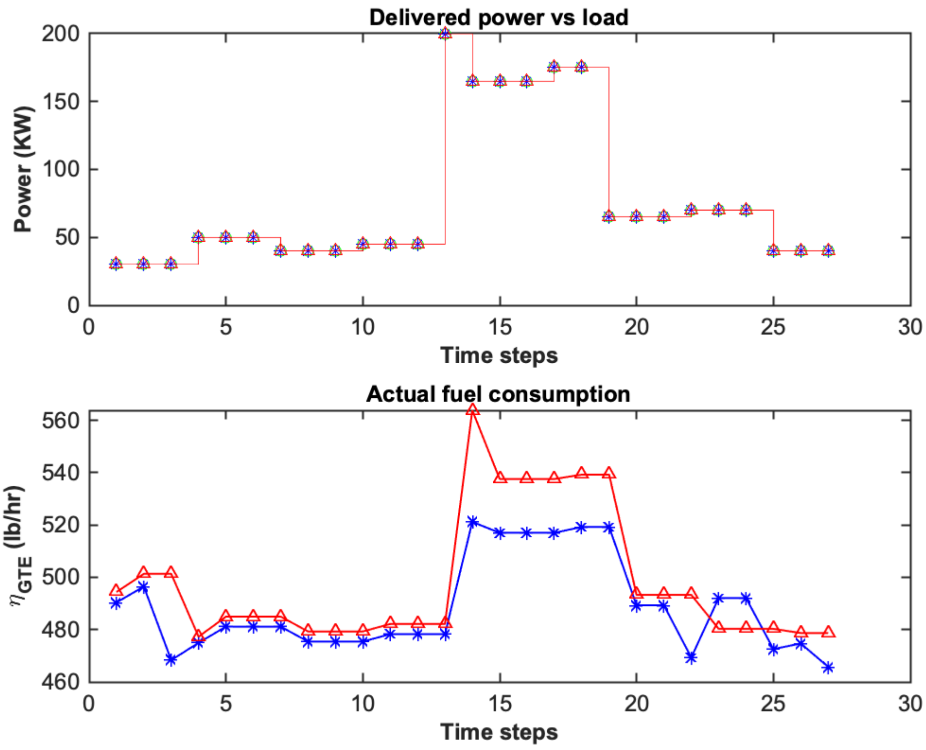


Figure 6.4: **TOP**: Offline dynamic programming (DP) power scheduling (red) and Algorithm 6.1 (blue) vs the load demand profile (green). The load demand profile is overlaid as both algorithms satisfied the requirements. **BOTTOM**: Fuel consumption using offline DP power scheduling with average fuel: 498.04 lb/hr (red) vs Algorithm 6.1 with average fuel: 488.54 lb/hr (blue).

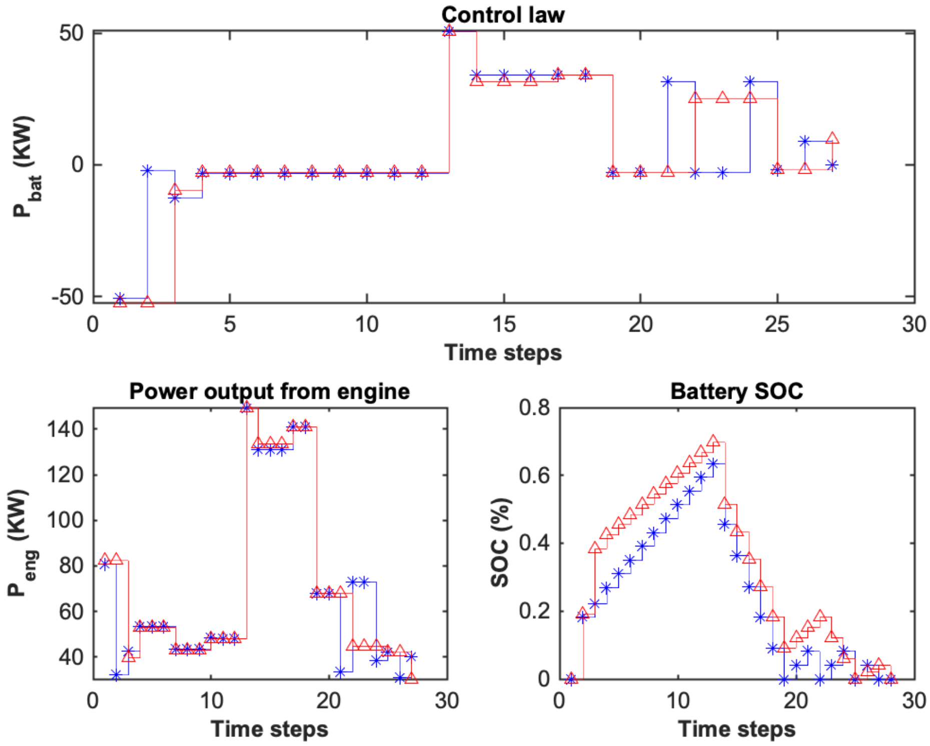


Figure 6.5: **TOP**: Control law from applying offline dynamic programming (DP) power scheduling (red) vs Algorithm 6.1 (blue). **BOTTOM**: Gas turbine engine power output and battery state-of-charge, *SOC* from implementation of both power management strategies.

6.3.3 Variation in system objectives and load requirements

The use of the offline (pre-defined) power schedules heightens the risk of failure due to system variations. Variations can occur from changes in system operation objectives which may result in a change in the future load demand profile [131]. Consider a load demand change at time steps $k = 19$ to $k = 20$ in Figure 6.6. The offline power schedule becomes infeasible as it is unable to adapt the battery power to the event change and compensate for the engine running at maximum power, thus failing to satisfy the load requirements at all times. Algorithm 6.1 is however able to satisfy the load requirements by fully delivering the required load power, given the information about the future load demand online. The RL-ADP scheme is therefore able to determine the best power strategy by computing the best charging/discharging cycles for the battery SOC in anticipation of the load change as shown in Figures 6.6 and 6.7.

Summary

This chapter has proposed and demonstrated an online power management optimisation scheme based on reinforcement learning and adaptive dynamic programming. Current power management strategies are heuristic and thus suboptimal, and are unable to compensate for modelling uncertainties and variations in system conditions. The proposed scheme computes online the optimal control strategies by using system measurements as reinforcement signals to adapt the system efficiency functions and deliver improved system performance. Simulation results using representative data sets showed that improved fuel consumption was achieved using the proposed online power management strategy compared to the conventional strategies, whilst satisfying the changing future load requirements.

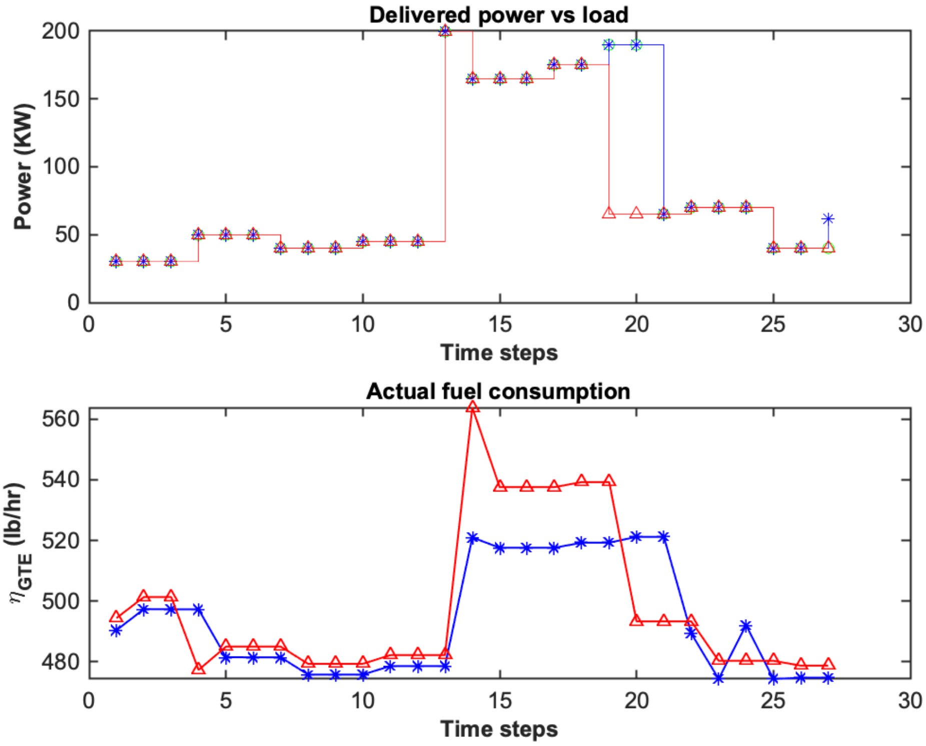


Figure 6.6: **TOP**: Offline dynamic programming (DP) power scheduling (red) and Algorithm 6.1 (blue) vs the load demand profile (green). The load demand profile is overlaid by the output of Algorithm 6.1 indicating that the requirements are fully satisfied but not with the Offline DP. **BOTTOM**: Fuel consumption using offline DP power scheduling with average fuel: 498.04 lb/hr (red) vs Algorithm 6.1 with average fuel: 493.47 lb/hr (blue).

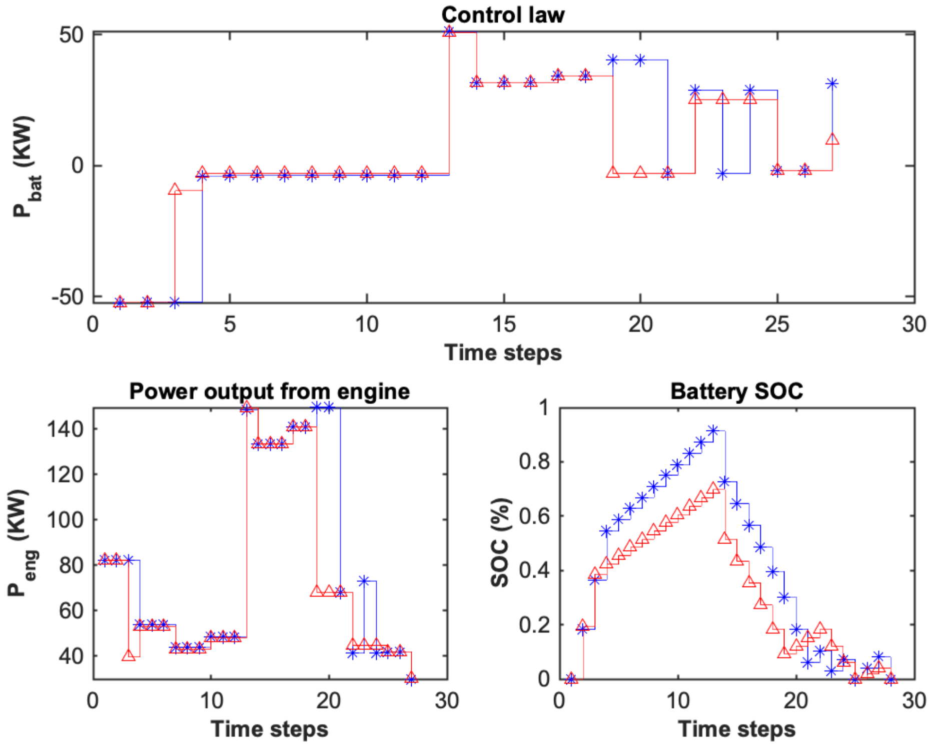


Figure 6.7: **TOP**: Control law from applying offline dynamic programming (DP) power scheduling (red) vs Algorithm 6.1 (blue). **BOTTOM**: Gas turbine engine power output and battery state-of-charge, *SOC* from implementation of both power management strategies.

Chapter 7

Conclusions and recommendations

7.1 Conclusions

Adaptive controllers are designed to use the system measurements to learn and modify the behaviour of the controller in response to changes in the system dynamics and operating conditions. However, the conventional adaptive control techniques design a controller against an identified system model that is assumed to characterise the desired performance metrics for all the possible system variations and are said to be indirect. In contrast, direct adaptive schemes explicitly adjust control actions to optimise a desired performance cost such as minimum fuel consumption, system durability and life without the need to learn the system model or assume characterisation of the performance metrics. Techniques that exploit emerging diagnostic technologies and enable the direct use of complex performance metrics to deliver self-optimising control systems in the face of disturbances and system variations are termed in this thesis as condition-based. This thesis has thus shown the development of direct optimal and adaptive condition-based control (CBC) frameworks using reinforcement learning for time-varying dynamical systems that do not require explicit mathematical models to characterise the varying system states due to degradation but systematically optimise the desired system performance.

In Chapter 3, the development of online reinforcement learning (RL) frameworks that are designed to be both adaptive and optimal for the control of time-varying dynamical systems was presented. In contrast to existing reinforcement learning techniques in the literature for the control of time-varying dynamical systems that use analytical energy functions as reward signals, the developed frameworks enable the direct use of complex measures of system performance such as fuel consumption, efficiency or life as reward signals. This is because the analytical models or expressions for these reward signals are difficult to derive due to system variations resulting from degradation or engine-to-engine differences. In introducing such complex performance metrics as reward signals for the control of time-varying dynamical systems, new ways to guarantee the system safety and reliability are required. This was achieved by proposing a CBC framework that integrates the RL adaptations into existing (pre-stabilised/certified) controller structures, thereby maintaining the system safety and reliability. Consequently, application of the proposed RL framework was shown for the CBC of gas turbine engines (GTEs) that makes use of the complex performance metrics to directly learn and adapt the system control. Simulation results on representative engine data sets showed improved fuel consumption in the GTEs as compared to their conventional control scheme, and provide potential for a significant reduction in operating costs across fleet-wide engines.

In Chapter 4, the development of an online optimal reinforcement learning tracking control framework for time-varying dynamical systems that uses an augmented formulation with integral control was presented. Existing tracking reinforcement learning techniques in the literature either assume the use of a predetermined feedforward control input, use restrictive assumptions on the reference model dynamics or use discounted tracking costs. By using a discounted tracking cost, the existing reinforcement learning tracking techniques are unable to guarantee zero steady-state error. In contrast, the proposed augmented formulation with integral control enables the development of reinforcement learning frameworks that do not make any restrictive assumptions of the existing techniques on the reference model dynamics and guarantees zero steady-state tracking error. Simulation results of the proposed online optimal tracking control framework on representative case studies showed the use of RL in computing the optimal tracking controller gains for the unknown or varying systems, and achieving zero steady-state tracking error. Furthermore, the proposed online RL tracking control is able to continually adapt the controller gains to optimal values, thus providing a through-life adaptation strategy.

In Chapter 5, the development of an online output-feedback reinforcement learning framework for time-varying dynamical systems that uses an augmented formulation with integral control was presented. In contrast to the proposed technique in Chapter 4 that requires full state measurements, the proposed output-feedback approach make use of only input/output data for systems in which full state measurements may be unavailable or the design of state-estimators is difficult. Furthermore, the proposed output-feedback approach does not make any restrictive assumptions of the existing reinforcement learning techniques on the reference model dynamics and equally guarantees zero steady-state tracking error whilst integrating the adaptations into existing controller structures for a through-life adaptation strategy. Simulation results showed that the resulting auto-regressive moving-average (ARMA) tracking controller achieve zero steady-state tracking error on convergence of the RL adaptations.

Finally in Chapter 6, the development of an online power management optimisation scheme for hybrid systems that uses dynamic programming and an iterative Q-learning adaptation of the system performance function in a receding horizon manner to compensate for gradual system variations or uncontrolled system disturbances was presented. Current state-of-the-art power management optimisation schemes are either based on pre-defined rule based power schedules or exhaustive model-based optimisation that assume known system models and efficiency functions. Consequently, the existing approaches are unable to account for unmodelled system variations and disturbances in the hybrid systems resulting in less efficient use of the available power sources. In contrast, the proposed power management optimisation scheme is able to learn and compensate for the gradual system variations and learn online the optimal power management strategy between the hybrid power sources given future load predictions. This way, improved system performance is delivered and providing a through-life adaptation strategy. Simulation results using representative data sets showed that improved fuel consumption was achieved using the proposed online power management strategy compared to the conventional strategies, whilst satisfying the changing future load requirements.

Overall, this thesis has proposed the design of a class of direct adaptive controllers for time-varying dynamical systems that is able to learn online, the optimal controller solutions to some desirable performance costs without the need to learn the system model and using only the measured system data. Furthermore, extension of this class of direct adaptive controllers to complex propulsion and power systems such as the gas turbine engines whose performances are affected by a vast nuber of varying factors has been proposed. To conclude, the contributions of this thesis are now summarised:

- The design of control architectures and algorithms that incorporate reinforcement learning approaches into existing controller structures for complex propulsion and power systems has been proposed. The innovative architectures advance the state-of-the-art to allow for direct optimisation of desired system performance measures whilst satisfying the system safety and stability constraints.

- The design of two new online optimal reinforcement learning tracking control frameworks for time-varying dynamical systems that guarantee zero steady-state tracking error and which unlike prior art do not make any restrictive assumptions on reference model dynamics or use of discounted tracking costs has been proposed. The first proposed online optimal RL tracking framework uses state and input measurements, while the second uses only the input/output data for systems where full state measurements may be unavailable.

- The design of a new online power management optimisation scheme for hybrid systems that uses dynamic programming and an iterative Q-learning adaptation of the system performance function in a receding horizon manner has been proposed. The proposed power management scheme advances the state-of-the-art by compensating for gradual system variations, extracting improved system performance and iteratively learning online, the optimal power management strategy between the hybrid power sources given the future load predictions.

7.2 Recommendations for future work

The proposed techniques and methods in this thesis can be extended in the following directions for future work:

- The proposed reinforcement learning framework in Chapter 3 can be extended in gas turbine engines to provide lifing performance optimisation using reward measurements that are only available at the end of each flight cycle. This extended framework would further exploit the delayed reward concept in reinforcement learning which takes into account future control actions that will improve the system life far into the future. Envisaged complexities would be in training large function networks that would be flexible and sensitive to the limited/sparse lifing reward signals, whilst guaranteeing the convergence of the network parameters.

- The proposed online optimal reinforcement learning tracking control frameworks in Chapters 4 and 5 have only been shown on discrete-time linear time-varying systems. Recommendations for future work is to provide extension of the frameworks to discrete-time nonlinear time-varying systems and also to continuous-time domains. Envisaged complexities would be in providing rigorous stability proofs for the nonlinear system adaptations and convergence of the overall scheme.

- Lastly, the proposed online power management optimisation scheme for hybrid systems as introduced an iterative and receding horizon dynamic programming routine whilst adapting the system efficiency functions to compensate for system variations. Dynamic programming was introduced to avoid learning the entire energy state space with function approximations given large horizon future load predictions. However, due to inherent curse-of-dimensionality limitations of the dynamic programming routine, possible recommendations for future work would be to consider a Q-function approximation over finite/limited future load prediction horizon. Envisaged complexities would be in training the Q-function approximation networks for varying future load demands over the finite/limited prediction horizon.

Bibliography

[1] H.A. Spang III and H. Brown. Control of jet engines. *Control Engineering Practice*, 7(9):1043–1059, 1999.

[2] A. Linke-Diesinger. *Systems of commercial turbofan engines: an introduction to systems functions*. Springer Science & Business Media, 2008.

[3] J.S. Litt, D.L. Simon, S. Garg, T.H. Guo, C. Mercer, R. Millar, A. Behbahani, A. Bajwa, and D.T. Jensen. A survey of intelligent control and health management technologies for aircraft propulsion systems. *Journal of Aerospace Computing, Information, and Communication*, 1(12):543–563, 2004.

[4] F.L. Lewis and V.L. Syrmos. *Optimal control*. John Wiley & Sons, 1995.

[5] J.B. Rawlings and D.Q. Mayne. *Model predictive control: Theory and design*. Nob Hill Pub., 2009.

[6] M. Czubenko, Z. Kowalczuk, and A. Ordys. Autonomous driver based on an intelligent system of decision-making. *Cognitive computation*, 7(5):569–581, 2015.

[7] W.J. Rugh and J.S. Shamma. Research on gain scheduling. *Automatica*, 36 (10):1401–1425, 2000.

[8] K.J. Åström and B. Wittenmark. *Adaptive control*. Courier Corporation, 2013.

[9] F.L. Lewis, D. Vrabie, and K.G. Vamvoudakis. Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. *IEEE Control Systems*, 32(6):76–105, 2012.

[10] R. Kurz and K. Brun. Degradation in gas turbine systems. *Journal of Engineering for Gas Turbines and Power*, 123(1):70–77, 2001.

[11] Rolls-Royce. *The Jet Engine*. Rolls-Royce Technical Publications, London, 5th edition, 2005.

[12] B.D. Anderson et al. Failures of adaptive control theory and their resolution. *Communications in Information & Systems*, 5(1):1–20, 2005.

[13] B.D. Anderson and A. Dehghani. Challenges of adaptive control–past, permanent and future. *Annual Reviews in Control*, 32(2):123–135, 2008.

[14] D.A. Bristow, M. Tharayil, and A.G. Alleyne. A survey of iterative learning control. *IEEE Control Systems Magazine*, 26(3):96–114, 2006.

[15] G. Tao. Multivariable adaptive control: A survey. *Automatica*, 50(11):2737–2764, 2014.

[16] M. Benosman. Model-based vs data-driven adaptive control: An overview. *International Journal of Adaptive Control and Signal Processing*, 32(5):753–776, 2018.

[17] B. Kiumarsi, K.G. Vamvoudakis, H. Modares, and F.L. Lewis. Optimal and autonomous control using reinforcement learning: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 29(6):2042–2062, 2018.

[18] Z.S. Hou and Z. Wang. From model-based control to data-driven control: Survey, classification and perspective. *Information Sciences*, 235:3–35, 2013.

[19] L. Ljung. System identification. *Wiley Encyclopedia of Electrical and Electronics Engineering*, pages 1–19, 1999.

[20] H. Kaufman, I. Barkana, and K. Sobel. *Direct adaptive control algorithms: theory and applications*. Springer Science & Business Media, 2012.

[21] K.S. Narendra and J. Balakrishnan. Adaptive control using multiple models. *IEEE Transactions on Automatic Control*, 42(2):171–187, 1997.

[22] J. Jäschke, Y. Cao, and V. Kariwala. Self-optimizing control–a survey. *Annual Reviews in Control*, 43:199–223, 2017.

[23] G.B. Gilyard and J.S. Orme. Performance seeking control: program overview and future directions. *NASA Technical Memorandum*, 4531, 1993.

[24] D.E. Viassolo, S. Adibhatla, B.J. Brunell, J.H. Down, N.S. Gibson, A. Kumar, H.K. Mathews, and L.D. Holcomb. Advanced estimation for aircraft engines. In *Proceedings of the 2007 American Control Conference*, pages 2807–2821. IEEE, 2007.

[25] M.L. Darby, M. Nikolaou, J. Jones, and D. Nicholson. Rto: An overview and assessment of current practice. *Journal of Process Control*, 21(6):874–884, 2011.

[26] A. Marchetti, B. Chachuat, and D. Bonvin. Modifier-adaptation methodology for real-time optimization. *Industrial & Engineering Chemistry Research*, 48(13):6022–6033, 2009.

[27] J.S. Orme and G.B. Gilyard. Subsonic flight test evaluation of a propulsion system parameter estimation process for the f100 engine. *NASA Technical Memorandum*, 4426, 1992.

[28] D. Viassolo, A. Kumar, and B. Brunell. Advanced controls for fuel consumption and time-on-wing optimization in commercial aircraft engines. In *ASME Turbo Expo 2007: Power for Land, Sea, and Air*, pages 539–548. American Society of Mechanical Engineers, 2007.

[29] S.G. Khan, G. Herrmann, F.L. Lewis, T. Pipe, and C. Melhuish. Reinforcement learning and optimal adaptive control: An overview and implementation examples. *Annual Reviews in Control*, 36(1):42–59, 2012.

[30] G. Battistelli, E. Mosca, M.G. Safonov, and P. Tesi. Stability of unfalsified adaptive switching control in noisy environments. *IEEE Transactions on Automatic Control*, 55(10):2424–2429, 2010.

[31] J.C. Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE Transactions on Automatic Control*, 45(10):1839–1853, 2000.

[32] H. Hjalmarsson. Iterative feedback tuning—an overview. *International Journal of Adaptive Control and Signal Processing*, 16(5):373–395, 2002.

[33] M.C. Campi, A. Lecchini, and S.M. Savaresi. Virtual reference feedback tuning: a direct method for the design of feedback controllers. *Automatica*, 38(8):1337–1346, 2002.

[34] K.B. Ariyur and M. Krstic. *Real-time optimization by extremum-seeking control*. John Wiley & Sons, 2003.

[35] C. Zhang and R. Ordóñez. *Extremum-seeking control and applications: a numerical optimization-based approach*. Springer Science & Business Media, 2011.

[36] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[37] R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[38] B. Kiumarsi-Khomartash, F.L. Lewis, M.B. Naghibi-Sistani, and A. Karimpour. Optimal tracking control for linear discrete-time systems using reinforcement learning. In *52nd IEEE Conference on Decision and Control*, pages 3845–3850. IEEE, 2013.

[39] R. Hafner and M. Riedmiller. Reinforcement learning in feedback control. *Machine learning*, 84(1-2):137–169, 2011.

[40] R. Bellman. A markovian decision process. Technical report, DTIC Document, 1957.

[41] W.B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[42] D.P. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995.

[43] D.P. Bertsekas. *Dynamic programming and optimal control*, volume 2. Athena scientific Belmont, MA, 1995.

[44] P.J. Werbos, W. Miller, and R. Sutton. A menu of designs for reinforcement learning over time. *Neural Networks for Control*, pages 67–95, 1990.

[45] D.V. Prokhorov and D.C. Wunsch. Adaptive critic designs. *IEEE Transactions on Neural Networks*, 8(5):997–1007, 1997.

[46] S. Ferrari and R.F. Stengel. An adaptive critic global controller. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, volume 4, pages 2665–2670. IEEE, 2002.

[47] D.P. Bertsekas and J.N. Tsitsiklis. Neuro-dynamic programming: an overview. In *Proceedings of the 34th IEEE Conference on Decision and Control*, volume 1, pages 560–564. IEEE, 1995.

[48] R.A. Howard. Dynamic programming and markov processes.. 1960.

[49] S. Bhasin. *Reinforcement learning and optimal control methods for uncertain nonlinear systems*. University of Florida, 2011.

[50] P.J. Werbos. Approximate dynamic programming for real-time control and neural modeling. *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, 15:493–525, 1992.

[51] R.J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[52] J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

[53] J. Peters, K. Mulling, and Y. Altun. Relative entropy policy search. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[54] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

[55] I. Grondman. *Online Model Learning Algorithms for Actor-Critic Control*. TU Delft, Delft University of Technology, 2015.

[56] R.S. Sutton, D.A. McAllester, S.P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1063, 2000.

[57] A. Al-Tamimi, F.L. Lewis, and M. Abu-Khalaf. Model-free q-learning designs for linear discrete-time zero-sum games with application to h-infinity control. *Automatica*, 43(3):473–481, 2007.

[58] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. 2014.

[59] L.C. Baird III and A.W. Moore. Gradient descent for general reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 968–974, 1999.

[60] S.J. Bradtke, B.E. Ydstie, and A.G. Barto. Adaptive linear quadratic control using policy iteration. In *Proceedings of the 1994 American Control Conference*, volume 3, pages 3475–3479. IEEE, 1994.

[61] L.C. Baird. Reinforcement learning in continuous time: Advantage updating. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 4, pages 2448–2453. IEEE, 1994.

[62] S. Ten Hagen and B. Kröse. Linear quadratic regulation using reinforcement learning. 1998.

[63] T. Landelius. *Reinforcement learning and distributed local model synthesis*. PhD thesis, Linköping University Electronic Press, 1997.

[64] Q. Zhao, H. Xu, and J. Sarangapani. Finite-horizon near optimal adaptive control of uncertain linear discrete-time systems. *Optimal Control Applications and Methods*, 36(6):853–872, 2015.

[65] D. Vrabie, O. Pastravanu, M. Abu-Khalaf, and F.L. Lewis. Adaptive optimal control for continuous-time linear systems based on policy iteration. *Automatica*, 45(2):477–484, 2009.

[66] T. Bian and Z.P. Jiang. Value iteration and adaptive dynamic programming for data-driven adaptive optimal control design. *Automatica*, 71:348–360, 2016.

[67] T. Dierks and S. Jagannathan. Online optimal control of affine nonlinear discrete-time systems with unknown internal dynamics by using time-based policy update. *IEEE Transactions on Neural Networks and Learning Systems*, 23 (7):1118–1129, 2012.

[68] Y. Lv, J. Na, Q. Yang, X. Wu, and Y. Guo. Online adaptive optimal control for continuous-time nonlinear systems with completely unknown dynamics. *International Journal of Control*, 89(1):99–112, 2016.

[69] D. Vrabie and F. Lewis. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Networks*, 22(3):237–246, 2009.

[70] K.G. Vamvoudakis, D. Vrabie, and F.L. Lewis. Online adaptive algorithm for optimal control with integral reinforcement learning. *International Journal of Robust and Nonlinear Control*, 24(17):2686–2710, 2014.

[71] M. Abu-Khalaf and F.L. Lewis. Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network hjb approach. *Automatica*, 41(5):779–791, 2005.

[72] H. Modares, M.B.N. Sistani, and F.L. Lewis. A policy iteration approach to online optimal control of continuous-time constrained-input systems. *ISA transactions*, 52(5):611–621, 2013.

[73] X. Yang, D. Liu, and D. Wang. Reinforcement learning for adaptive optimal control of unknown continuous-time nonlinear systems with input constraints. *International Journal of Control*, 87(3):553–566, 2014.

[74] H. Modares, F.L. Lewis, and M.B. Naghibi-Sistani. Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems. *Automatica*, 50(1): 193–202, 2014.

[75] B. Kiumarsi, F.L. Lewis, H. Modares, A. Karimpour, and M.B. Naghibi-Sistani. Reinforcement q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. *Automatica*, 50(4):1167–1175, 2014.

[76] B. Kiumarsi, F.L. Lewis, M.B. Naghibi-Sistani, and A. Karimpour. Optimal tracking control of unknown discrete-time linear systems using input-output measured data. *IEEE Transactions on Cybernetics*, 45(12):2770–2779, 2015.

[77] H. Zhang, Q. Wei, and Y. Luo. A novel infinite-time optimal tracking control scheme for a class of discrete-time nonlinear systems via the greedy hdp iteration algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):937–942, 2008.

[78] T. Dierks and S. Jagannathan. Optimal tracking control of affine nonlinear discrete-time systems with unknown internal dynamics. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 6750–6755. IEEE, 2009.

[79] D. Wang, D. Liu, and Q. Wei. Finite-horizon neuro-optimal tracking control for a class of discrete-time nonlinear systems using adaptive dynamic programming approach. *Neurocomputing*, 78(1):14–22, 2012.

[80] Z. Ni, H. He, and J. Wen. Adaptive learning in tracking control based on the dual critic network design. *IEEE Transactions on Neural Networks and Learning Systems*, 24(6):913–928, 2013.

[81] B. Kiumarsi and F.L. Lewis. Actor–critic-based optimal tracking for partially unknown nonlinear discrete-time systems. *IEEE Transactions on Neural Networks and Learning Systems*, 26(1):140–151, 2014.

[82] Y.J. Liu, L. Tang, S. Tong, C.P. Chen, and D.J. Li. Reinforcement learning design-based adaptive tracking control with less learning parameters for nonlinear discrete-time mimo systems. *IEEE Transactions on Neural Networks and Learning Systems*, 26(1):165–176, 2014.

[83] B. Kiumarsi, F.L. Lewis, and D.S. Levine. Optimal control of nonlinear discrete time-varying systems using a new neural network approximation structure. *Neurocomputing*, 156:157–165, 2015.

[84] L. Liu, Z. Wang, and H. Zhang. Adaptive fault-tolerant tracking control for mimo discrete-time systems via reinforcement learning algorithm with less

learning parameters. *IEEE Transactions on Automation Science and Engineering*, 14(1):299–313, 2016.

[85] Q. Lin, Q. Wei, and D. Liu. A novel optimal tracking control scheme for a class of discrete-time nonlinear systems using generalised policy iteration adaptive dynamic programming algorithm. *International Journal of Systems Science*, 48(3):525–534, 2017.

[86] C. Qin, H. Zhang, and Y. Luo. Online optimal tracking control of continuous-time linear systems with unknown dynamics by using adaptive dynamic programming. *International Journal of Control*, 87(5):1000–1009, 2014.

[87] H. Modares and F.L. Lewis. Linear quadratic tracking control of partially-unknown continuous-time systems using reinforcement learning. *IEEE Transactions on Automatic control*, 59(11):3051–3056, 2014.

[88] H. Modares, F.L. Lewis, and Z.P. Jiang. H-infinity tracking control of completely unknown continuous-time systems via off-policy reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 26(10): 2550–2562, 2015.

[89] H. Modares and F.L. Lewis. Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning. *Automatica*, 50(7):1780–1792, 2014.

[90] A.Y. Ng, S. Sastry, H.J. Kim, and M.I. Jordan. Autonomous helicopter flight via reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 799–806, 2004.

[91] A.Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang. Autonomous inverted helicopter flight via reinforcement learning. In *Experimental Robotics IX*, pages 363–372. Springer, 2006.

[92] P. Abbeel, A. Coates, M. Quigley, and A.Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *Advances in Neural Information Processing Systems*, pages 1–8, 2007.

[93] D. O'Neill, M. Levorato, A. Goldsmith, and U. Mitra. Residential demand response using reinforcement learning. In *First IEEE International Conference on Smart Grid Communications*, pages 409–414. IEEE, 2010.

[94] B. Jiang and Y. Fei. Dynamic residential demand response and distributed generation management in smart microgrid with hierarchical agents. *Energy Procedia*, 12:76–90, 2011.

[95] T. Huang and D. Liu. A self-learning scheme for residential energy system control and management. *Neural Computing and Applications*, 22(2):259–269, 2013.

[96] M. Boaro, D. Fuselli, F. De Angelis, D. Liu, Q. Wei, and F. Piazza. Adaptive dynamic programming algorithm for renewable energy scheduling and battery management. *Cognitive Computation*, 5(2):264–277, 2013.

[97] Q. Wei, D. Liu, and G. Shi. A novel dual iterative q-learning method for optimal battery management in smart residential environments. *IEEE Transactions on Industrial Electronics*, 62(4):2509–2518, 2014.

[98] X. Qi, G. Wu, K. Boriboonsomsin, M.J. Barth, and J. Gonder. Data-driven reinforcement learning–based real-time energy management system for plug-in hybrid electric vehicles. *Transportation Research Record*, 2572(1):1–8, 2016.

[99] Y. Zou, T. Liu, D. Liu, and F. Sun. Reinforcement learning-based real-time energy management for a hybrid tracked vehicle. *Applied energy*, 171:372–382, 2016.

[100] A. Sheikhi, M. Rayati, and A. Ranjbar. Dynamic load management for a residential customer; reinforcement learning approach. *Sustainable Cities and Society*, 24:42–51, 2016.

[101] A. Anvari-Moghaddam, A. Rahimi-Kian, M.S. Mirian, and J.M. Guerrero. A multi-agent based energy management solution for integrated buildings and microgrid system. *Applied Energy*, 203:41–56, 2017.

[102] R. Xiong, J. Cao, and Q. Yu. Reinforcement learning-based real-time power management for hybrid energy storage system in the plug-in hybrid electric vehicle. *Applied energy*, 211:538–548, 2018.

[103] Y. Zhu, D. Zhao, and X. Li. Using reinforcement learning techniques to solve continuous-time non-linear optimal tracking problem without system dynamics. *IET Control Theory & Applications*, 10(12):1339–1347, 2016.

[104] H. Zhang, Y. Luo, and D. Liu. Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints. *IEEE Transactions on Neural Networks*, 20(9):1490–1503, 2009.

[105] J. Kober, J.A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[106] A.K. Akametalu, J.F. Fisac, J.H. Gillula, S. Kaynama, M.N. Zeilinger, and C.J. Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control*, pages 1424–1431. IEEE, 2014.

[107] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause. Safe model-based reinforcement learning with stability guarantees. In *Advances in Neural Information Processing Systems*, pages 908–918, 2017.

[108] J. Garcıa and F. Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[109] I. Sanusi, A. Mills, P. Trodden, V. Kadirkamanathan, and T. Dodd. Reinforcement learning for condition-based control of gas turbine engines. In *Proceedings of the 18th European Control Conference (ECC)*, pages 3928–3933. IEEE, 2019.

[110] M. Bohlin, K. Doganay, P. Kreuger, R. Steinert, and M. Warja. Searching for gas turbine maintenance schedules. *AI Magazine*, 31(1):21–36, 2010.

[111] V.V. Silva, W. Khatib, and P.J. Fleming. Performance optimization of gas turbine engine. *Engineering Applications of Artificial Intelligence*, 18(5):575–583, 2005.

[112] C. Bringhenti and J. Barbosa. Methodology for gas turbine performance improvement using variable-geometry compressors and turbines. *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, 218(7):541–549, 2004.

[113] S. Garg. Fundamentals of aircraft turbine engine control. Technical report, 2010.

[114] S. Garg. Aircraft turbine engine control research at nasa glenn research center. *Journal of Aerospace Engineering*, 26(2):422–438, 2013.

[115] H. Richter. *Advanced control of turbofan engines*. Springer Science & Business Media, 2011.

[116] C. Eckert and O. Isaksson. Safety margins and design margins: A differentiation between interconnected concepts. *Procedia CIRP*, 60:267–272, 2017.

[117] P. Venkataraman. *Applied optimization with MATLAB programming*. John Wiley & Sons, 2009.

[118] I. Sanusi, A. Mills, T. Dodd, and G. Konstantopoulos. Online optimal and adaptive integral tracking control for varying discrete-time systems using reinforcement learning. *International Journal of Adaptive Control and Signal Processing*, 2020.

[119] R.E. Precup, S. Preitl, I.J. Rudas, M.L. Tomescu, and J.K. Tar. Design and experiments for a class of fuzzy controlled servo systems. *IEEE/ASME Transactions on Mechatronics*, 13(1):22–35, 2008.

[120] K.F. Krommydas and A.T. Alexandridis. Nonlinear stability analysis for ac/dc voltage source converters driven by pi current-mode controllers. In *Proceedings of the 2014 European Control Conference (ECC)*, pages 2774–2779. IEEE, 2014.

[121] G.C. Konstantopoulos and Q.C. Zhong. Current-limiting dc/dc power converters. *IEEE Transactions on Control Systems Technology*, 27(2):855–863, 2018.

[122] I. Sanusi, A. Mills, and G. Konstantopoulos. Output-feedback tracking with integral control using reinforcement learning. 2020.

[123] F.L. Lewis and K.G. Vamvoudakis. Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(1):14–25, 2010.

[124] F.L. Lewis and K.G. Vamvoudakis. Optimal adaptive control for unknown systems using output feedback by reinforcement learning methods. In *Proceedings of the 8th IEEE International Conference on Control and Automation (ICCA)*, pages 2138–2145. IEEE, 2010.

[125] G. Hewer. An iterative technique for the computation of the steady state gains for the discrete optimal regulator. *IEEE Transactions on Automatic Control*, 16(4):382–384, 1971.

[126] I. Sanusi, A. Mills, G. Konstantopoulos, and T. Dodd. Power management optimisation for hybrid electric systems using reinforcement learning and adaptive dynamic programming. In *Proceedings of the 2019 American Control Conference (ACC)*, pages 2608–2613. IEEE, 2019.

[127] H. Lee, J. Jeong, Y.i. Park, and S.W. Cha. Energy management strategy of hybrid electric vehicle using battery state of charge trajectory information. *International Journal of Precision Engineering and Manufacturing-Green Technology*, 4(1):79–86, 2017.

[128] J. Hoelzen, Y. Liu, B. Bensmann, C. Winnefeld, A. Elham, J. Friedrichs, and R. Hanke-Rauschenbach. Conceptual design of operation strategies for hybrid electric aircraft. *Energies*, 11(1):217, 2018.

[129] L.V. Pérez, G.R. Bossio, D. Moitre, and G.O. García. Optimization of power management in an hybrid electric vehicle using dynamic programming. *Mathematics and Computers in Simulation*, 73(1-4):244–254, 2006.

[130] L. Karunarathne, J.T. Economou, and K. Knowles. Power and energy management system for fuel cell unmanned aerial vehicle. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 226 (4):437–454, 2012.

[131] M.M. Mansor, I. Giagkiozis, D. Wall, A.R. Mills, R.C. Purshouse, and P.J. Fleming. Real-time improved power management for autonomous systems. *IFAC Proceedings Volumes*, 47(3):2634–2639, 2014.

[132] B. Morley and D. Wall. Intelligent power management: the development of a functional architecture. In *5th SEAS DTC Technical Conference*, 2010.

[133] S. Abedi, A. Alimardani, G. Gharehpetian, G. Riahy, and S. Hosseinian. A comprehensive method for optimal power management and design of hybrid res-based autonomous energy systems. *Renewable and Sustainable Energy Reviews*, 16(3):1577–1587, 2012.

[134] S.C. Chang, C.H. Chen, I.K. Fong, and P.B. Luh. Hydroelectric generation scheduling with an effective differential dynamic programming algorithm. *IEEE Transactions on Power Systems*, 5(3):737–743, 1990.

[135] Y.Y. Hsu and C.C. Su. Dispatch of direct load control using dynamic programming. *IEEE Transactions on Power Systems*, 6(3):1056–1061, 1991.

[136] X. Dong, G. Bao, Z. Lu, Z. Yuan, and C. Lu. Optimal battery energy storage system charge scheduling for peak shaving application considering battery lifetime. In *Informatics in Control, Automation and Robotics*, pages 211–218. Springer, 2011.

[137] Y. Riffonneau, S. Bacha, F. Barruel, and S. Ploix. Optimal power flow management for grid connected pv systems with batteries. *IEEE Transactions on Sustainable Energy*, 2(3):309–320, 2011.

[138] C.C. Lin, H. Peng, J.W. Grizzle, and J.M. Kang. Power management strategy for a parallel hybrid electric truck. *IEEE Transactions on Control Systems Technology*, 11(6):839–849, 2003.

[139] F.R. Salmasi. Control strategies for hybrid electric vehicles: Evolution, classification, comparison, and future trends. *IEEE Transactions on Vehicular Technology*, 56(5):2393–2404, 2007.

[140] S.G. Wirasingha and A. Emadi. Classification and review of control strategies for plug-in hybrid electric vehicles. *IEEE Transactions on Vehicular Technology*, 60(1):111–122, 2010.

[141] A.M. Phillips, M. Jankovic, and K.E. Bailey. Vehicle system controller design for a hybrid electric vehicle. In *Proceedings of the 2000. IEEE International Conference on Control Applications. (Cat. No. 00CH37162)*, pages 297–302. IEEE, 2000.

[142] H. Banvait, S. Anwar, and Y. Chen. A rule-based energy management strategy for plug-in hybrid electric vehicle (phev). In *Proceedings of the 2009 American Control Conference*, pages 3938–3943. IEEE, 2009.

[143] E.S. Koo, H.D. Lee, S.K. Sul, and J.S. Kim. Torque control strategy for a parallel hybrid vehicle using fuzzy logic. In *Conference Record of 1998 IEEE Industry Applications Conference. Thirty-Third IAS Annual Meeting (Cat. No. 98CH36242)*, volume 3, pages 1715–1720. IEEE, 1998.

[144] E.D. Tate and S.P. Boyd. Finding ultimate limits of performance for hybrid electric vehicles. Technical report, SAE Technical Paper, 2000.

[145] S. Delprat, J. Lauber, T.M. Guerra, and J. Rimaux. Control of a parallel hybrid powertrain: optimal control. *IEEE Transactions on Vehicular Technology*, 53(3): 872–881, 2004.

[146] C.C. Lin, H. Peng, and J. Grizzle. A stochastic control strategy for hybrid electric vehicles. In *Proceedings of the 2004 American Control Conference*, volume 5, pages 4710–4715. IEEE, 2004.

[147] M.P. O'Keefe and T. Markel. Dynamic programming applied to investigate energy management strategies for a plug-in hev. Technical report, National Renewable Energy Lab.(NREL), Golden, CO (United States), 2006.