Online learning algorithms

For passivity-based and distributed control

Nageshrao, Subramanya

DOI
10.4233/uuid:9f3a2496-7851-40f6-a947-102080bdd5fd

Publication date
2016

Document Version
Final published version

Citation (APA)
Nageshrao, S. (2016). *Online learning algorithms: For passivity-based and distributed control*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:9f3a2496-7851-40f6-a947-102080bdd5fd

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

ONLINE LEARNING ALGORITHMS

FOR PASSIVITY-BASED AND DISTRIBUTED CONTROL

ONLINE LEARNING ALGORITHMS

FOR PASSIVITY-BASED AND DISTRIBUTED CONTROL

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K.C.A.M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op

maandag 18 April 2016 om 12:30 uur

door

Subramanya Prasad NAGESHRAO

Master of Science, Technische Universität Hamburg-Harburg, Duitsland,
geboren te Mandya, India.

This dissertation has been approved by the

promotor: Prof. dr. R. Babuška
copromotor: Dr. G.A.D Lopes

Composition of the doctoral committee:

Rector Magnificus,	Chairman
Prof. dr. R. Babuška,	Technische Universiteit Delft
Dr. G.A.D. Lopes,	Technische Universiteit Delft
Independent members:	
Prof. dr. S. Bhatnagar,	Indian Institute of Science
Dr. L. Buşoniu,	Technische Universiteit Cluj Napoca
Prof. dr.ir. J.M.A. Scherpen,	Rijksuniversiteit Groningen
Prof. dr.ir. N. van de Wouw,	Technische Universiteit Delft
Other member:	
Dr. D. Jeltsema,	Technische Universiteit Delft
Reserve member:	
Prof. dr.ir. B. De Schutter,	Technische Universiteit Delft

This dissertation has been completed in partial fulfillment of the requirements of the Dutch Institute of Systems and Control (DISC) for graduate studies.

Email: subramanyanageshrao@gmail.com

ISBN 9789461866219

An electronic version of this dissertation is available at
`http://repository.tudelft.nl/`
Cover design by: Reshu Gupta e-mail:reshurgupta@gmail.com

Printed in the Netherlands

Dedicated to my grandparents.

ACKNOWLEDGEMENTS

This is a rather interesting section for me to write. Considering this is the last section that I am writing for the thesis, owing to structure of the book, this would probably be the first one to be read. I always felt acknowledgments is a way to humanize the thesis. It is farthest from all the mathematical equations that follow. Writing this gave me an opportunity to look back at the last few years. Assuming this will be the last thesis I am ever going to write, I have taken the liberty of words to thank various individuals who have helped, inspired and influenced me, albeit, with a near certainty that I might miss someone important.

Probably it was during my high school that I came across this story, the gist of which goes like this: A student at the start of academic life is more like a raw material, say a piece of rock. It takes a sculptor to work on it relentlessly and make something good out of that potential. When I look back at the last four years I could not have asked for a better sculptor than my promotor Prof. dr. ir. Robert Babuška.

Dear Robert, thank you for believing in me, for giving me this wonderful opportunity to pursue doctoral education. I always admired your sincerity to help students. Your attention to details, curiosity to learn and immense enthusiasm are few of the qualities I actively tried to imitate and often failed miserably.

This thesis would have stayed a mote idea if it was not for my co-promotor and daily supervisor Dr. Gabriel Lopes. Dear Gabriel, while you had to take care of little kids at home, you had to endure grown up kid(s) at the office. Your patience and constant encouragement made my last four years exceptional. A special thanks for always keeping your office door open to discuss any (non) technical issues. You have always supported me both on professional and personal front, helped and pushed me to achieve higher goals while simultaneously giving me enough freedom to conduct the research, Thank you!

I would also like to thank my other (unofficial) supervisor Dr. Dimitri Jeltsema, his willingness to collaborate, his patience in teaching me the basics of port-Hamiltonian systems and nonlinear control greatly helped me in my work. I would also like to thank ir. Olivier Sprangers, for being the first person to work on this idea.

During the initial four months of 2015, I visited University of Texas at Arlington (UTA) as a research visitor. I am grateful to Prof. dr. Frank Lewis for providing this valuable opportunity and for introducing me to multi-agent systems and distributed control. During my stay at UTA, I got the opportunity to work with Dr. Hamidreza Modares, it is extremely rare to come across a person who works relentlessly, is highly talented and adores his job. Reza, my friend you are destined for greatness.

I express my gratitude to the members of PhD committee, Prof. dr. Shalabh Bhatnagar, Dr. Lucian Buşoniu, Prof. dr. ir. Jacquelien Scherpen, Prof. dr. ir. Nathan van de Wouw and Prof. dr. ir. Bart de Schutter, your suggestions and comments greatly helped in improving the quality of this thesis.

There are many people who have guided me at various stages of my academic life. A few unforgettable names are, Prof. Herbert Werner, for introducing me to the world of advance control, Prof. Radhakrishna Rao and Athmanand sir, for being a perfect role model.

I would also express my gratitude to all my masters students, Jan-Willem, Yudha and Anshuman - you all did a great job. I am thankful to always smiling and wonderful staff at DCSC: Marieke, Kitty, Heleen, Saskia, Esther, Martha, Kiran and Ditske – you are the rock stars of our department. Experimental study would have been impossible without timely help from Will, Kees, Ron and Fankai, your support is much appreciated.

Thanks to my fabulous colleagues, life at DCSC was never dull. The numerous biking trips, badminton, football, volleyball, fussball, frisbee games were always fun and I always played a significant part in those games. Actually in the hindsight the teams I played for always ended up losing, hmm, lets keep this discussion for another day. The interaction I had with numerous past and current colleagues of DCSC has greatly helped in my thesis. Particularly, I am thankful to Ivo, for helping me with RL concepts in the initial days. Amol you have been a great mentor and guide, you have helped me so often I sincerely lack words to thank you. Past and current colleagues at DCSC, Alfredo, Zhe, Amir, Mohammad, Esmaeil, Farid, Yiming, Vishal, Anqi, Le, Shaui, Jia, Max, Yihuai, Ilya, Kim, Shuai, Yashar, Vahab, Ana, Jan-Marteen, Laura, Jens, Manuel, Jan-Willem, Simone, Hans, Michel, Alex, Samira, Ilya, Marco, Dieky, John and Cornelis – thanks to each of you for many interesting discussions, talks and a quick 'Hi' in the hallway or near the coffee machine. A special thanks to Tim and Cess for translating my summary into Dutch, dank je wel jongens.

Life in Delft wouldn't have been fun without regular badminton on Tuesdays and Saturdays with Mohammad, Yue, Jia, Le and Sadegh. The road (and ski) trips with Noortje, Sachin, Renshi, Hans, Hildo are few of the most cherishable moments of the last four years. My board game buddies, Edwin, Paolo, Bart, Laurens, Dean, Reinier, Baptiste and Pieter, thank you for organizing fabulous game nights and even more fabulous wine!

Andra, you are one of the sweetest person I have ever met-keep smiling. Ellie – where to start, we have taught courses together, planned epic road trip together, scared even a charging elephant! Well it was only you, I was just screaming for my life. I am indebted for your patience, wisdom and encouragement, you were always there for me whenever I was in need of help or a cup of coffee! Thank you.

My amazing roommates over the years, Mukunda, Susana, Omar, Mathew and Anna, thank you for giving me a home away from home. A special thanks to Muks, there is never a dearth of laughter when you are around. In the beginning of 2014, I moved to 'dharmshala' also known as AvS for official purpose. Living here is an experience in itself. At times it consists of various creatures and occasional thieves. Having said that, the core of the 'dharmshala' is led by the able bodied Tiggy Patel, Aditya, Sachin, Chocka, Reshu (cover page designer!), Kalsi, Sid, Srijith, Yash and Kaa aka Chiku, thank you all. My stay in Delft would't have been memorable if it was not for my friends from subcontinent, Girish, Sairam, Zubir, Krishna, Jai, Adi, Rajat, Sumit, Vijay, Murali and Seshan.

My thesis writing was relatively stress free mainly due to one person, Tiggy's mom Mrs. Amita Patel, thank you for taking care of me.

My band of brothers, Kirana, Lohi, Gopala, Deepu, Chandra and Sri - thank you for

being there for me at my lowest of ebb and defining some of the finest moments of my life. My undergrad classmates 'E&C batch of 2005 from PESCE', my batchmates at TUHH, my colleagues at Bosch, thank you.

If it was not for my family, I would not have reached this far. Only because of your care, love and encouragement, no goal seems unreachable, thank you for being there for me at each and every step of my life. Lastly, my wonderful girl, Navya, because of you *life is beautiful*.

Subramanya Prasad Nageshrao
Delft, March 2016

CONTENTS

1

INTRODUCTION

In this chapter the motivation behind the thesis is provided. It also introduces reinforcement learning as a key concept that is prominently used in this work. Following the introduction, an overview on the focus and contributions of the thesis is given. Subsequently a detailed outline of the thesis, both verbal and pictorial, is presented.

1.1. MOTIVATION

The need to control a (man-made) physical system is arguably as ancient as our civilization. However, the genesis of *modern control* can be traced to the onset of industrial revolution, which ushered the beginning of our technological era [1]. Control and technology form a positive feedback loop: control enhances the technical performance of a system and advances in technology enable the realization of better control algorithms. Over time, this mechanism results in significant improvement in performance [1, 2].

Humanity is surrounded by man-made machines. The accessibility of technology in daily life such as, advanced medical facilities, transportation, etc. is often used as a measure for the quality of human development. The technology and machines generally enhance the comfort and ease of life. Many of the dynamic machines require a valid controller in order to function in a safe and efficient manner. Even though control is the functioning brain in various applications, its ubiquitous presence is not readily evident. Control applications range from a simple coffee maker to an awe inspiring space shuttle. The available control methods are as diverse as the systems to which they are applied. Most of the control methods can be broadly classified either as linear or nonlinear control [3]. Linear control is prominently used and it is a rich field with a wide-variety of methods. As it is beyond the scope of this thesis, these methods are not discussed in detail, for further information see [4, 5]. The desirability of linear control can be attributed to the elegant design methods and the ease of implementation. Although it is widely used, linear control may be inefficient when the control objective is demanding or when the dynamic system under consideration does not obey the superposition principle. This may be due to hard nonlinearities such as hysteresis, saturation, nonlinear

friction, etc. In these scenarios, linear control may be used in a relatively small operating range. Typically, this implies a rather small part of operating space where the controller can achieve the desired stability and performance criterion. Some of these difficulties can be addressed by using nonlinear control. Moreover nonlinear control will also be extremely useful when the system under consideration is complex, multi-domain and has a large operating range or is driven by stringent control objectives [6, 7].

Although nonlinear control is highly desired, designing and implementing a nonlinear controller can be a challenging and extremely hard task. In many instances it is stymied by the need for full state information, high sensitivity to the system parameters and dependence on the system model. Even if these requirements are addressed, for example, by using a nonlinear observer and by performing a precise system identification, the nonlinear control synthesis problem is still difficult as it often involves a set of partial-differential, algebraic equations or differential equations [8, 9]. Solving these equations can be both cumbersome and time consuming. In an abstract sense, the issue faced can be attributed to the structure of the feedback controller itself. This is because the control can be viewed as a fixed map from plant states to the plant input that is obtained offline. When encountering an uncertain and imprecise model, a pre-fixed map might not be able to achieve the desired control objectives. This is also evident when only a limited prior system information is available. In these scenarios it is nearly impossible to consider all the uncertainties during the feedback control synthesis. Additionally, while designing a nonlinear controller, there is no general mechanism to incorporate a performance measure. This is because most of the techniques are devised only to achieve regulation or tracking without alluding to the performance criterion. By using learning techniques, instead of model-based nonlinear methods, at least some of the mentioned issues can be addressed [10].

Animals and humans have the ability to share, explore, act or respond, memorize the outcome and repeat the task to achieve better result when they encounter the same or similar scenario. This is called learning from interaction [11, 12]. A learning system is characterized by its ability to improve the performance, use the past experience or information and adapt to changes in the environment. Thus learning algorithms, in principle, are capable of controlling a poorly modeled nonlinear dynamic system. As will be explained in the following sections, in this thesis an instance of learning algorithm called *Reinforcement Learning* (RL) is used to solve a particular family of nonlinear control and distributed control problems [13].

1.2. REINFORCEMENT LEARNING

Reinforcement learning is a subclass of machine learning techniques. It is a collection of algorithms that can be used to solve sequential decision making problems [14]. The learning objective is to control a system so as to maximize some predefined performance criterion. The system is assumed to be a Markov decision process (MDP), which is a default framework for sequential decision making problems. The sequential problem generally involves a mechanism to obtain a series of actions in a closed-loop setting. In RL this is addressed by the policy,[1] a map from the system's state to action. The control

[1] In RL literature the terms policy, agent, actor, and controller are used interchangeably.

objective is to find an optimal policy such that some expected measure of performance is maximized [15].

While the system is assumed to be an MDP, RL methods can learn a policy without requiring any a priori information about the system. In an RL algorithm the required knowledge is obtained by directly interacting or experimenting with the system. Based on the experimental data a control law is learned online. Owing to this basic principle of learning from interaction, RL can obtain a local optimal policy even in an uncertain and/or time-varying environment. Thus RL can readily address model and parametric uncertainties [16].

Unlike in supervised learning where an explicit desired output is given to the agent, in RL the only available feedback is an evaluative scalar reward. The reward gives an instantaneous measure of performance. The working of an RL algorithm is as follows: at every time instance the agent senses the system's state. It then calculates an appropriate action using the policy at hand. On applying this action the system transits to a new state and provides a numerical scalar reward. Using this evaluative feedback the policy is modified [17, 18].

A key component for any learning method is its memory. In the RL setting this is represented either by the state-value or the action-value function. These functions approximate the cumulative reward, called the return, starting from a particular state or a state-action pair, respectively. The goal in RL is to find a policy that maximizes the return over the course of interaction.

RL algorithms can be broadly classified into three categories. The *Critic-only* methods first learn an optimal value function from which an optimal policy is computed. The *Actor-only* algorithms directly search for an optimal policy in the policy-space using a gradient-based optimization approach. *Actor-Critic* methods explicitly learn a policy (actor) and also a value-function (critic). The critic provides an evaluation of the actor's performance. As will be explained in the following sections, the main focus of this thesis is to solve the parameterized nonlinear control and distributed control problems. For this purpose the useful class of RL algorithms are Actor-only and Actor-Critic methods. In both of these approaches the learning algorithm can use parameterized policies and the parameters of which can be learned online by using the gradient descent optimization method [19].

1.3. FOCUS AND CONTRIBUTIONS

Although the use of reinforcement learning in control and robotics is gaining traction, its widespread applications are still limited. A prominent factor that adversely affects the broad use of RL is the lack of interpretability of the policy, i.e., no physical meaning can be attributed to the learned control law. This is due to the absence of a general framework to relate an existing parameterized control law to an RL policy. Additionally, RL methods are stymied by the curse of dimensionality, slow and non-monotonic convergence of the learning algorithm, etc. Partially, this can be attributed to the characteristic of RL as it lacks a standard mechanism for incorporating any a priori model information into the RL algorithm [20–22].

This thesis aims to address the mentioned issues such as, interpretability of the learned control law, easier mechanisms to incorporate prior knowledge, enhancing the learn-

ing speed, etc. This is done by first parameterizing a well known nonlinear control design method called passivity-based control. The parameters of the controller are then learned online by using a variant of the standard actor-critic algorithm. Using the elements of stochastic approximation theory the convergence of the policy parameters to a local minimum of the pre-defined cost function is shown. In the last part of the thesis, RL is used to solve the multi-agent output synchronization problem for a network of linear heterogeneous systems [23, 24].

1.3.1. SOLVING PBC USING RL

Passivity-based control (PBC) is a well-known model-based nonlinear control approach. It is prominently used for regulation and tracking control of physical systems. Applications of PBC can be found in various domains such as, electrical, mechanical, electromechanical, etc. [9, 25, 26]. PBC achieves the control objective by rendering the closed-loop passive, hence the name [27]. PBC is a model-based control strategy and it traditionally relies on the system's model. The system's dynamic model is typically represented either in the Euler-Lagrangian [9] or port-Hamiltonian (PH) form [25]. In this thesis, the physical systems modeled in the PH form are used.

Port-Hamiltonian theory provides a novel mechanism to model a complex physical system. By using the port theory for networked systems, the interaction between various components of a complex system can be represented, whereas the dynamics of each component can be derived by using the Hamiltonian framework. This natural way of representing the dynamics of a complex multi-domain system is the most prominent feature in the PH framework. In the control community, PBC is a preferred model-based control approach for systems represented in PH form.

However, the passivity-based control synthesis for a PH system often involves solving a set of under-determined complex partial differential equations (PDEs) [28]. Analytic solutions of these equations can be hard to compute. Additionally, similar to other model-based approaches, the veracity of the solution may not be guaranteed due to parameter and model uncertainties. Even if a feasible solution is found there is no standard mechanism to incorporate a performance criterion. In this thesis instead of obtaining an analytical solution for the PBC control law of a PH system, the controller is first parameterized in terms of an unknown parameter vector. Then, by using a variation of the standard actor-critic learning algorithm the unknown parameters are learned online [18]. Thanks to the online learning capabilities, parameter/model uncertainties and performance criterion can be readily addressed. Thanks to the effectiveness of the developed algorithms, real-time learning on an experimental setup for various systems is possible.

An extensive numerical and experimental evaluation of the developed algorithms is conducted to check for the scalability and performance. Since in the proposed methods prior system knowledge can be easily incorporated in the form of a PH model, a comparison study is done to highlight the advantages w.r.t. the standard model-free actor-critic approach. Additionally, because of learning, the proposed methods can achieve zero steady state error in the presence of model uncertainties. A comparison study is done to highlight the advantages w.r.t. standard passivity-based control. A mechanism to convert the PBC control law into a stochastic policy is given. By using the principles

of stochastic approximation algorithms the proof-of-convergence of the passivity-based actor-critic method is shown [29, 30].

1.3.2. SOLVING MULTI-AGENT TRACKING USING RL

Using the port-based modeling framework PH theory can be used to model a networked system when complete state information is available. Alternatively, a networked system can be considered as a set of individual agents sharing only limited information such as output among the neighbours. In multi-agent systems the control objective is that all the agents are required to reach agreement on certain quantities of interests. If the common value that agents agree on is not specified, then the problem is called leaderless consensus. If all agents follow the trajectories of a leader node, then the problem is known as cooperative tracking (leader-follower) control. A rich body of literature is available on distributed control of multi-agent systems. Generally the available methods for multi-agent tracking of heterogeneous systems assume complete knowledge of the leader as well as the agent's dynamics [24]. However, in practice this assumption is rather unrealistic. Although the existing adaptive methods can address the uncertainty in the agent's dynamics, the optimality of the control law is yet to be explicitly achieved. In Chapter 6, a novel model-free integral reinforcement-learning algorithm is proposed. Unlike the standard methods, the proposed approach does not require either the solution of the output regulator equations or the incorporation of a p-copy, i.e., a model of the leader's dynamics in the controller of each agent. Also because of RL, the learned control is both model-free and optimal. Numerical evaluation demonstrates the effectiveness of the developed method.

1.3.3. CONTRIBUTIONS

The major contributions of this thesis are:

1. A set of actor-critic learning algorithms are proposed to solve the passivity-based control problem for mechanical and electro-mechanical systems.

2. Using the stochastic approximation theory [29] the convergence of the developed RL based passivity methods is shown.

3. A learning algorithm was developed to solve the optimal output tracking control problem for a heterogeneous multi-agent network.

1.4. THESIS OUTLINE

Including the introduction, this thesis in total consists of 7 chapters. The remaining chapters are organized as follows. Chapter 2 introduces two basic entities that are prominently used throughout the thesis, namely, port-Hamiltonian systems and passivity-based control. First, examples are given to illustrate the modeling in PH form. Following this, various port-Hamiltonian system properties are explained. The chapter ends with a brief survey on the standard model-based passivity control methods for PH systems. In Chapter 3 the existing state-of-the-art adaptive and learning methods that explicitly use the PH structure are reviewed. Starting with the need for learning in the PH framework, adaptive control, iterative control, and evolutionary control techniques that use the PH

system properties are reviewed. For each method the essential changes from the general setting due to the PH model are highlighted. This is followed by a detailed presentation of the respective control algorithms. Finally as a concluding remark a brief note on the open research issues is given. Chapter 4 introduces the actor-critic method and its modifications to solve the passivity-based control problems. A detailed numerical and experimental study of the developed algorithms is provided. Furthermore, a comparison study with the standard actor-critic and model-based PBC is given. In Chapter 5 the proof-of-convergence of the developed methods is shown using the stochastic approximation framework. The application of RL to solve the optimal output tracking of a multi-agent heterogeneous system is given in Chapter 6. First a distributed adaptive observer is used to obtain an estimate of the leader's state in each agent. Following this, integral reinforcement-learning (IRL) is used to learn the feedback and feed-forward components of the tracking control law, online and model-free. Detailed numerical study shows the effectiveness of the proposed method. The thesis concludes with Chapter 7 where a summary of major conclusions and recommendations for future research is provided. The list of the publications by author is listed on page 134 of the thesis. The pictorial representation of the thesis outline is given in Figure 1.1. Sequential reading of the thesis is recommended since it gradually introduces all the essential components. However, if the reader has prior knowledge of PH theory and PBC, then Chapter 3 can be skipped.

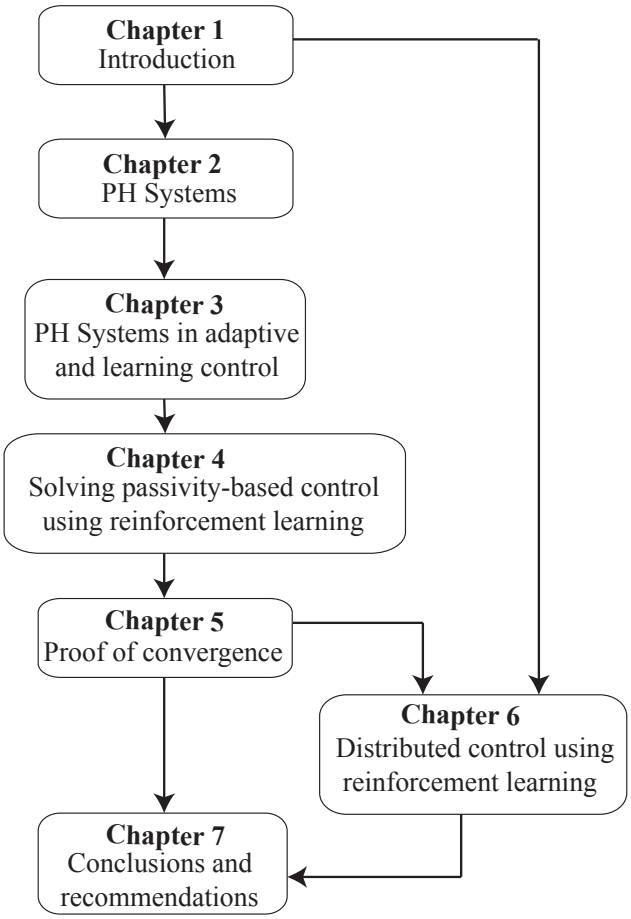


Figure 1.1: Outline of the thesis providing an suggestive roadmap for the reading order.

2

PORT-HAMILTONIAN SYSTEMS

Port-Hamiltonian (PH) theory is a novel, but well established modeling framework for nonlinear physical systems. Due to the emphasis on the physical structure and modular framework, PH modeling has become a prime focus in system theory. This has led to a considerable research interest in the control of PH systems, resulting in numerous nonlinear control techniques. This chapter describes the modelling and model-based control of port-Hamiltonian systems. A variation of the Hamiltonian systems called the controlled-Hamiltonian systems will also be introduced. The methods surveyed here provide a brief overview of the state of the art model-based control of PH systems. These methods are extensively used in later part of the thesis for e.g., in Chapter 3 and 4.

2.1. INTRODUCTION

Port-Hamiltonian (PH) modeling of physical systems [8, 31, 32] has found a wide acceptance and recognition in the systems and control community. Thanks to the underlying principle of system modularity and the emphasis on the physical structure and interconnections, the PH formulation can be efficiently used to model complex multi-domain physical systems [25]. The main advantage of the PH approach is that the Hamiltonian can be used as a basis to construct a candidate Lyapunov function, thus providing valuable insight into numerous system properties like passivity, stability, finite L_2 gain [8], etc. These features have led to a deep research focus on the control of port-Hamiltonian systems. There are numerous interrelated control methods which have been extended or developed specifically for PH systems, namely canonical transformation [33], control by interconnection (CbI) [25, 34, 35] , energy-balancing [34, 36], interconnection and damping assignment passivity-based control (IDA-PBC) [37, 38]. In this chapter these methods are collectively denoted as model-based synthesis methods. A brief overview of the prominent synthesis methods for PH systems is presented. For an in-depth review of the PH control approaches refer to [25, 26] and the references therein. All these methods rely on the PH model of the physical system and generally the controller is obtained by solving a set of partial-differential or algebraic equations.

The chapter is organized as follows. In Section 2, PH systems will be introduced along with examples to illustrate the representation of well-known mechanical and electromechanical systems in PH form. The conversion from PH form to the controlled Hamiltonian representation is also given in Section 2. In Section 3 properties of port and controlled Hamiltonian systems are given. Following this, prominent passivity-based control methods are explained in Section 4, and Section 5 provides the summary.

2.2. HAMILTONIAN SYSTEMS

In this section basic theoretical background on port and controlled-Hamiltonian framework, representation, and examples are given.

2.2.1. PORT-HAMILTONIAN SYSTEMS

Port-Hamiltonian[1] systems are often considered as a generalization of Euler-Lagrangian or Hamiltonian systems. PH modeling stems from the port-based network modeling of multi-domain complex physical systems having distinct energy storage elements (e.g., electrical, mechanical, electro-mechanical, chemical, hydrodynamical and thermodynamical systems). A strong aspect of the port-Hamiltonian formalism is that it emphasizes the physics of the system by highlighting the relationship between the energy storage, dissipation, and the interconnection structures. Additionally, finite-dimensional PH theory can be readily extended to infinite-dimensional (distributed-parameter) systems [39].

A time-invariant PH system in the standard input-state-output form is given as

$$\begin{aligned} \dot{x} &= \big(J(x) - R(x)\big)\frac{\partial H}{\partial x}(x) + g(x)u, \quad x \in \mathbb{R}^n, \\ y &= g^T(x)\frac{\partial H}{\partial x}(x), \end{aligned} \tag{2.1}$$

where $J(x) = -J^T(x) \in \mathbb{R}^{n\times n}$ is the skew-symmetric interconnection matrix, $R(x) = R^T(x) \in \mathbb{R}^{n\times n}$ is the symmetric positive semi-definite dissipation matrix, and $g(x) \in \mathbb{R}^{n\times m}$ is the input matrix. The Hamiltonian $H(x) \in \mathbb{R}$ is the system's total stored energy, obtained by adding the energy stored in all the individual energy-storing elements. Signals $u \in \mathbb{R}^m$ and $y \in \mathbb{R}^m$ are called the port variables and their inner-product forms the supply rate which indicates the power supplied to the system.

Example 1. *PH modeling of a mechanical system:* Some systems have a natural PH representation, for example, a fully actuated mechanical system is described by:

$$\begin{aligned} \begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix} &= \Bigg(\underbrace{\begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}}_{J} - \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & D \end{bmatrix}}_{R}\Bigg)\begin{bmatrix} \frac{\partial H}{\partial q}(x) \\ \frac{\partial H}{\partial p}(x) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{g} u, \\ y &= \begin{bmatrix} 0 & I \end{bmatrix}\begin{bmatrix} \frac{\partial H}{\partial q}(x) \\ \frac{\partial H}{\partial p}(x) \end{bmatrix}, \end{aligned} \tag{2.2}$$

[1] The terminology used in the literature also includes terms like port-controlled Hamiltonian systems (PCH), port-controlled Hamiltonian systems with dissipation (PCHD), generalized Hamiltonian systems, etc.

where the generalized position $q \in \mathbb{R}^{\bar{n}}$ and the momentum $p \in \mathbb{R}^{\bar{n}}$ form the system state $x = [q^T \quad p^T]^T$. The matrix $D \in \mathbb{R}^{\bar{n}\times\bar{n}}$ represents the dissipation, and $2\bar{n} = n$. The Hamiltonian $H(x)$ is the sum of the kinetic and the potential energy,

$$H(x) = \frac{1}{2} p^T M^{-1}(q) p + V(q) = T(x) + V(q), \tag{2.3}$$

where $M(q) \in \mathbb{R}^{\bar{n}\times\bar{n}}$ is the mass-inertia matrix, $T(x) \in \mathbb{R}$ and $V(q) \in \mathbb{R}$ are the kinetic and the potential energy terms, respectively. □

Example 2. *PH modeling of an electro-mechanical system:* The magnetic levitation system of Figure 2.1 consists of two subsystems, namely i) a mechanical system — iron ball of mass M, ii) an electro-magnetic system — a coil of nominal inductance L_0 and resistance Z.

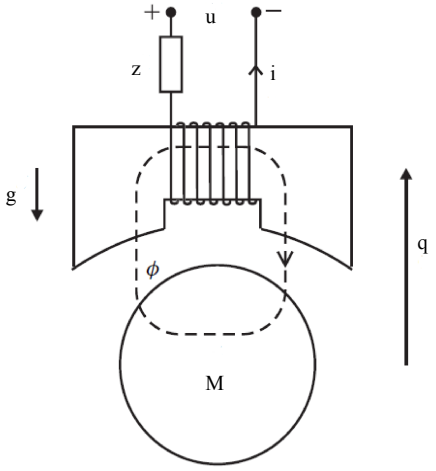


Figure 2.1: Schematic representation of magnetic levitation of an iron ball [32].

The dynamics of the magnetic-levitation system using the first principles are [37]:

$$\begin{aligned} \dot{\phi} &= u - Zi, \\ M\ddot{q} &= F_{\text{emf}} - Mg, \end{aligned} \tag{2.4}$$

where $u \in \mathbb{R}$ and $i \in \mathbb{R}$ are the voltage across and the current through the coil, respectively, q is the position of the ball and F_{emf} is the magnetic force acting on the ball. The effective magnetic flux ϕ linking the coil is a function of the position q, and it can be approximated as $\phi = L(q)i$. Using the approximation for the varying inductance

$$L(q) = \frac{L_0}{1-q}, \tag{2.5}$$

the effective force F_{emf} on the iron ball is

$$F_{\text{emf}} = \frac{1}{2}\frac{\partial L(q)}{\partial q} i^2. \tag{2.6}$$

For the Hamiltonian

$$H(x) = Mgq + \frac{p^2}{2M} + \frac{1}{2L_0}(1-q)\phi^2, \tag{2.7}$$

where $p = M\dot{q}$ is the momentum of the iron ball. By substituting (2.4)-(2.7) in (2.1), the system dynamic equation (2.4) can be represented in the PH form as

$$\begin{bmatrix} \dot{q} \\ \dot{p} \\ \dot{\phi} \end{bmatrix} = \left(\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & Z \end{bmatrix} \right) \begin{bmatrix} \frac{\partial H}{\partial q} \\ \frac{\partial H}{\partial p} \\ \frac{\partial H}{\partial \phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u,$$

$$y = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\partial H}{\partial q} \\ \frac{\partial H}{\partial p} \\ \frac{\partial H}{\partial \phi} \end{bmatrix}, \tag{2.8}$$

where

$$\begin{bmatrix} \frac{\partial H}{\partial q} \\ \frac{\partial H}{\partial p} \\ \frac{\partial H}{\partial \phi} \end{bmatrix} = \begin{bmatrix} Mg - \frac{\phi^2}{2L_0} \\ \frac{p}{M} \\ \frac{(1-q)\phi}{L_0} \end{bmatrix}. \tag{2.9}$$

For more examples and an in-depth theoretical background on PH theory, see [25, 32] and the references therein. □

2.2.2. CONTROLLED-HAMILTONIAN SYSTEMS

A controlled-Hamiltonian system – following the terminology of [40] – is described as

$$\begin{aligned} \dot{x} &= (J-R)\frac{\partial H}{\partial x}(x,u), \\ y &= -\frac{\partial H}{\partial u}(x,u). \end{aligned} \tag{2.10}$$

Note that (2.10) differs from the PH system representation of (2.1) in the input u, output y and constant system matrices J and R. Although the two types of systems are structurally different, there is a subclass of systems that can be written in both forms, as illustrated by the following example.

Example 3. *Continued from Example 2*

For the sake of simplicity, by using the resistance of $Z = 1\,\Omega$ and inductance $L_0 = 1\,\mathrm{H}$, the system Hamiltonian (2.7) can be rewritten as the controlled-Hamiltonian in terms of input u and state $x = [q \quad p \quad \phi]^T$

$$H(x,u) = Mgq + \frac{p^2}{2M} + \frac{1}{2}(1-q)\phi^2 - \frac{\phi}{Z}u. \tag{2.11}$$

Observe that the product $\frac{\phi}{Z}u$ has the unit of energy. Equation (2.1) can be transformed

to a controlled-Hamiltonian form (2.10) as

$$\begin{bmatrix} \dot{q} \\ \dot{p} \\ \dot{\phi} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}}_{(J-R)} \underbrace{\begin{bmatrix} \frac{\partial H}{\partial q}(x,u) \\ \frac{\partial H}{\partial p}(x,u) \\ \frac{\partial H}{\partial \phi}(x,u) \end{bmatrix}}_{\frac{\partial H}{\partial x}(x,u)}. \tag{2.12}$$

Note that due to the structural differences, the transformation from PH to a controlled-Hamiltonian system is not always possible. □

2.3. SYSTEM PROPERTIES

In this section essential system properties of the port-Hamiltonian and the controlled-Hamiltonian systems are explained.

2.3.1. PASSIVITY AND STABILITY

The notion of passivity can be stated using a generic time-invariant input affine nonlinear system

$$\dot{x} = f(x) + g(x)u, \tag{2.13}$$

where $x \in \mathbb{R}^n$ is the system state vector, $f(x) \in \mathbb{R}^n$ is the state-dependent nonlinear function and $g(x) \in \mathbb{R}^{n\times m}$ is the input function and $u \in \mathbb{R}^m$ is the input.

Definition 1. A given nonlinear system (2.13) is said to be passive if there exists a non-negative function $S(x) \in \mathbb{R}^+$ and a system output $y = h(x) \in \mathbb{R}^m$ such that the inequality

$$S(x(T)) - S(x(0)) \leq \int_0^T u(t)^T y(t) dt, \tag{2.14}$$

is satisfied.

For a continuous storage function $S(x)$ the passivity inequality (2.14) can be simplified as

$$\frac{dS}{dt}(x(t)) \leq u(t)^T y(t), \tag{2.15}$$

for an appropriate control input, for example the passive feedback $u(t) = -Ky(t)$ where $K = K^T \geq 0$, the system (2.13) can be rendered stable. The storage function $S(x)$ then becomes a Lyapunov-like function and it can be used to demonstrate the stability of the system. This indicates a strong correlation between passivity and stability, because given a passive system it can be easily stabilized using the passive feedback [41].

The passivity of a system is evaluated in two steps. First, for the given system (2.13) a passive output $y = h(x)$ is found. Then an appropriate positive semi-definite storage function $S(x)$ is formulated so that it satisfies the passivity inequality (2.15).

By assuming that the system's Hamiltonian $H(x)$ of (2.1) is bounded from below it is straightforward to illustrate the passivity property of a given PH system. Consider the

time-derivative of the Hamiltonian $H(x)$:

$$\begin{aligned}\frac{dH}{dt}(x(t)) &= \frac{\partial H^T}{\partial x}\dot{x} \\ &= \frac{\partial H^T}{\partial x}\big(J(x)-R(x)\big)\frac{\partial H}{\partial x} + \frac{\partial H^T}{\partial x}g(x)u \\ &= -\frac{\partial H^T}{\partial x}R(x)\frac{\partial H}{\partial x} + u^T y,\end{aligned} \tag{2.16}$$

where $u^T y$ is the supply rate. It is defined as the product of conjugate variables which is the power supplied to the system, for instance, Voltage × Current, Force × Velocity, etc. For a positive semi-definite dissipation matrix (i.e. $R(x) \geq 0$) the equality (2.16) reduces to

$$\frac{dH}{dt}(x(t)) \leq u^T y. \tag{2.17}$$

Equation (2.17) is called the differential dissipation inequality and it implies that in the presence of dissipation the change in the system's total stored energy is less than or equal to the supply rate with the difference being the dissipated energy [41]. Observe that (2.17) is similar to the derivative of the storage function $S(x)$ in (2.15).

2.3.2. VARIATIONAL SYSTEM

Consider a general nonlinear system operator Σ, acting on an input signal u and resulting in a system output y

$$\Sigma(u): \begin{cases} \dot{x} = f(x,u), \quad x \in \mathbb{R}^n, u \in \mathbb{R}^m \\ y = h(x,u), \end{cases} \tag{2.18}$$

where $f : \mathbb{R}^{n\times m} \to \mathbb{R}^n$ is a system function and $h : \mathbb{R}^{n\times m} \to \mathbb{R}^m$ is an output function. One can linearize (2.18) along an input and a system trajectory, $u(t)$ and $x(t)$, respectively, resulting in a linear time-variant (LTV) system [42]:

$$d\Sigma(u_{\mathrm{v}}): \begin{cases} \dot{x}_{\mathrm{v}} = \dfrac{\partial f(x,u)}{\partial x}x_{\mathrm{v}}(t) + \dfrac{\partial f(x,u)}{\partial u}u_{\mathrm{v}}(t), \\ y_{\mathrm{v}} = \dfrac{\partial h(x,u)}{\partial x}x_{\mathrm{v}}(t) + \dfrac{\partial h(x,u)}{\partial u}u_{\mathrm{v}}(t), \end{cases} \tag{2.19}$$

where $(x_{\mathrm{v}}, u_{\mathrm{v}}, y_{\mathrm{v}})$ are the variational state, input and outputs, respectively. They represent the variation along the trajectories $(x,\ u,\ y)$. For any controlled-Hamiltonian system (2.10) (or the subset of PH systems (2.1)) one can obtain the variational system using (2.19) [43]

$$d\Sigma(u_{\mathrm{v}}): \begin{cases} \dot{x}_{\mathrm{v}} = (J-R)\dfrac{\partial H_{\mathrm{v}}}{\partial x_{\mathrm{v}}}(x,u,x_{\mathrm{v}},u_{\mathrm{v}}), \\ y_{\mathrm{v}} = -\dfrac{\partial H_{\mathrm{v}}}{\partial u_{\mathrm{v}}}(x,u,x_{\mathrm{v}},u_{\mathrm{v}}), \end{cases} \tag{2.20}$$

where $H_{\mathrm{v}}(x,u,x_{\mathrm{v}},u_{\mathrm{v}})$ is the new controlled-Hamiltonian

$$H_{\mathrm{v}}(x,u,x_{\mathrm{v}},u_{\mathrm{v}}) = \frac{1}{2}\begin{bmatrix} x_{\mathrm{v}} & u_{\mathrm{v}} \end{bmatrix}\frac{\partial^2 H(x,u)}{\partial (x,u)^2}\begin{bmatrix} x_{\mathrm{v}} \\ u_{\mathrm{v}} \end{bmatrix}, \tag{2.21}$$

provided there exists a transformation matrix $T \in \mathbb{R}^{n\times n}$ that satisfies

$$\begin{aligned} J &= -TJT^{-1}, \\ R &= TRT^{-1}, \\ \frac{\partial^2 H(x,u)}{\partial(x,u)^2} &= \begin{bmatrix} T & 0 \\ 0 & I \end{bmatrix} \frac{\partial^2 H(x,u)}{\partial(x,u)^2} \begin{bmatrix} T^{-1} & 0 \\ 0 & I \end{bmatrix}. \end{aligned} \tag{2.22}$$

Unfortunately for a generic controlled-Hamiltonian system, obtaining a transformation matrix T so as to satisfy (2.22) is rather difficult. However, for a fully actuated mechanical system a simple trick to circumvent this problem has been demonstrated in [44], this is done by using an internally stabilizing PD controller .

2.3.3. ADJOINT SYSTEM

For a given LTV system

$$\Sigma(u): \begin{cases} \dot{x} = A(t)x(t) + B(t)u(t), \\ y = C(t)x(t) + D(t)u(t), \end{cases} \tag{2.23}$$

the adjoint operator is

$$\Sigma^*(u^*): \begin{cases} \dot{x}^* = -A^T(t)x^*(t) - C^T(t)u^*(t), \\ y^* = B^T(t)x^*(t) + D^T(t)u^*(t). \end{cases} \tag{2.24}$$

and it is related to the original system (2.23) by the vector inner-product [45]

$$\langle y, \Sigma(u)\rangle = \langle \Sigma^*(y), u\rangle. \tag{2.25}$$

The adjoint operator of a given system possesses various interesting properties, namely it can be used for model-order reduction [46], adjoint-based optimal control [47], etc.

Since the variational system (2.19) and (2.20) are in LTV form one can obtain their respective adjoint form. In [43], assuming invertibility of $J-R$, the adjoint of the controlled-Hamiltonian system (2.10) is obtained as

$$d\Sigma^*(u^*): \begin{cases} \dot{x}^* = -(J-R)\dfrac{\partial H^*}{\partial x^*}(x,u,x^*,u^*), \\ y^* = -\dfrac{\partial H^*}{\partial u^*}(x,u,x^*,u^*), \end{cases} \tag{2.26}$$

in terms of the new controlled-Hamiltonian $H^*(x,u,x^*,u^*)$

$$H^*(x,u,x^*,u^*) = \frac{1}{2}\begin{bmatrix} x^* & u^* \end{bmatrix} \frac{\partial^2 H(x,u)}{\partial(x,u)^2} \begin{bmatrix} x^* \\ u^* \end{bmatrix}. \tag{2.27}$$

From (2.20) and (2.26) it is evident that the variational and the adjoint of a Hamiltonian system have similar state-space realization. In [43] it is shown that – under the assumption of non-singularity of $(J-R)$ or the time symmetry of the Hessian of $H(x,u)$ – they are related by a time-reversal operator, i.e.,

$$(d\Sigma(u_v))^* := d\Sigma^*(u_v) = \mathscr{R} \circ d\Sigma(u_v), \tag{2.28}$$

where $\mathscr{R}$ is the time-reversal operator, i.e.,

$$\mathscr{R}(u(t)) = u(T-t)\ \forall\, t \in [0,T]. \tag{2.29}$$

This implies that the adjoint of a variational controlled-Hamiltonian system can be obtained from the variational system itself. Additionally the complexity involved in obtaining the variational system $d\Sigma$ can be avoided by using the local linear approximation [42]

$$d\Sigma(u_v) \approx \Sigma(u+u_v) - \Sigma(u). \tag{2.30}$$

Hence the adjoint output of a controlled-Hamiltonian system can be obtained from the actual system output without any a priori system information [43]. Optimal iterative learning controller of a port-Hamiltonian system by using self-adjoitness property is elaborated in Chapter 3.4.

2.4. CONTROL OF PH SYSTEMS

Passivity-Based control (PBC) is a model-based nonlinear control methodology that exploits the passivity property of a system to achieve various control objectives. In this section a brief overview of prominent static state-feedback PBC methods are given.

2.4.1. STABILIZATION VIA DAMPING INJECTION

Asymptotic stability of a given PH system can be achieved by using (2.17). Consider a negative feedback to the system as

$$u = -K(x)y, \tag{2.31}$$

with $K(x) = K^T(x) > 0 \in \mathbb{R}^{m\times m}$ i.e., $K(x)$ is a symmetric positive definite damping injection matrix, that needs to be designed by the user. Then the dissipation inequality (2.17) becomes

$$\frac{dH}{dt}(x) \leq -y^T K(x) y. \tag{2.32}$$

By assuming zero state detectability, the asymptotic stability of the PH system (2.1) at the origin can be inferred [8]. Stabilizing the system at the origin which corresponds to the open loop minimum energy is not an enticing control problem. Alternatively a wider practical interest is to stabilize the system at a desired equilibrium state, say x_*. In the PH framework this set-point regulation can be achieved by the standard PBC as elaborated below.

2.4.2. STANDARD PASSIVITY BASED CONTROL

ENERGY BALANCING

The energy-balancing (EB) equation is defined as

$$H(x(t)) - H(x(0)) = -\bar{R}(t) + \int_0^t u^T(\tau)y(\tau)d\tau. \tag{2.33}$$

Equation (2.33) is obtained by integrating the dissipation inequality (2.16), where $\bar{R}(t)$ is

$$\bar{R}(t) = \int_0^t \frac{\partial H}{\partial x}^T R(x) \frac{\partial H}{\partial x} d\tau,$$

is the existing dissipation in the system. The objective of EB is to devise a control input $\beta(x)$ so as to stabilize the given system (2.1) at a desired equilibrium. This equilibrium is generally associated with a minimum of the desired closed-loop Hamiltonian, i.e.,

$$x_* = \arg\min H_\mathrm{d}(x), \tag{2.34}$$

which is achieved by adding an external energy $H_\mathrm{a}(x)$ to the existing energy $H(x)$, i.e.,

$$H_\mathrm{d}(x) = H(x) + H_\mathrm{a}(x). \tag{2.35}$$

The desired Hamiltonian $H_\mathrm{d}(x)$ is called energy-balancing if the control input $\beta(x)$ satisfies the equality

$$H_\mathrm{a}(x(t)) = -\int_0^t \beta^T(x(\tau))y(\tau)d\tau, \tag{2.36}$$

or the differential inequality

$$\left(\frac{\partial H_\mathrm{a}}{\partial x}(x)\right)^T \left[(J(x)-R(x))\frac{\partial H}{\partial x} + g(x)\beta(x)\right] = -y^T\beta(x). \tag{2.37}$$

At the equilibrium the controlled system satisfies the equality,

$$(J(x_*)-R(x_*))\frac{\partial H}{\partial x}(x_*) + g(x_*)\beta(x_*) = 0,$$

this implies that the controller will not be able to extract any power at the equilibrium since $y(x_*)^T\beta(x_*) = 0$. Alternatively, this means energy-balancing can be used if and only if the given system (2.1) can be stabilized by extracting a finite amount of energy from the controller.

In energy-shaping the control objective is to find a control input that results in the closed-loop

$$\dot{x} = (J(x)-R(x))\frac{\partial H_\mathrm{d}}{\partial x}(x),$$

where $H_\mathrm{d}(x)$ satisfies (2.35) and (2.36). For the PH system (2.1), the energy-balancing control input $\beta(x)$ is obtained by solving the equality,

$$g(x)\beta(x) = (J(x)-R(x))\frac{\partial H_\mathrm{a}}{\partial x}(x). \tag{2.38}$$

ENERGY BALANCING AND DAMPING INJECTION

A combination of the previous two methods i.e., energy-balancing (2.38) and damping-injection (2.31) (EB-DI) is predominantly used for PH control. For a given PH system (2.1), the EB-DI objective is to obtain a target closed-loop of the form [36]

$$\dot{x} = \big(J(x)-R_\mathrm{d}(x)\big)\frac{\partial H_\mathrm{d}}{\partial x}(x), \tag{2.39}$$

where $R_\mathrm{d}(x)$ is the desired dissipation matrix given as

$$R_\mathrm{d}(x) \;=\; R(x) + g(x)K(x)g^T(x), \tag{2.40}$$

in terms of the damping injection matrix $K(x)$.

The desired closed-loop form of (2.39) can be obtained by using the control input[2]

$$\begin{aligned} u &= u_{\text{eb}}(x) + u_{\text{di}}(x) \\ &= \big(g^T(x)g(x)\big)^{-1} g^T(x)\big(J(x) - R(x)\big)\frac{\partial H_{\text{a}}}{\partial x}(x) \\ &- K(x)g^T(x)\frac{\partial H_{\text{d}}}{\partial x}(x), \end{aligned} \tag{2.41}$$

where the added energy term $H_{\text{a}}(x)$ is a solution of the set of PDE's

$$\begin{bmatrix} g^{\perp}(x)(J(x) - R(x))^T \\ g^T(x) \end{bmatrix} \frac{\partial H_{\text{a}}}{\partial x}(x) = 0, \tag{2.42}$$

with $g^{\perp}(x)$ the full-rank left annihilator matrix of the input matrix $g(x)$, i.e., $g^{\perp}(x)g(x) = 0$. Among the solutions of (2.42) the one satisfying (2.34) is chosen. If the second part of the matching condition (2.42) is satisfied, then $g^T(x)\frac{\partial H_{\text{d}}}{\partial x}(x)$ in (2.41) can be rewritten as

$$\begin{aligned} g^T(x)\frac{\partial H_{\text{d}}}{\partial x}(x) &= g^T(x)\left(\frac{\partial H}{\partial x}(x) + \frac{\partial H_{\text{a}}}{\partial x}(x)\right), \\ &= g^T(x)\frac{\partial H}{\partial x}(x) = y. \end{aligned}$$

By multiplying $(\frac{\partial H_{\text{a}}}{\partial x})^T$ to (2.38),

$$\left(\frac{\partial H_{\text{a}}}{\partial x}\right)^T (J(x) - R(x))\left(\frac{\partial H_{\text{a}}}{\partial x}\right) = \left(\frac{\partial H_{\text{a}}}{\partial x}\right)^T g(x)\beta(x) \tag{2.43}$$

using the second equality of (2.42) in (2.43) results in

$$R(x)\frac{\partial H_{\text{a}}}{\partial x} = 0. \tag{2.44}$$

This roughly implies that the added energy $H_{\text{a}}(x)$ should not depend on the states of the system that have natural damping. This is called the *dissipation obstacle*. A major drawback of the ES-DI approach is the dissipation obstacle, as it constraints the set of achievable equilibria by the controller [3]. This often limits the applicability of the method, whereas the following two additional methods, namely energy-shaping and interconnection and damping assignment (IDA) PBC, explained below, do not suffer from this drawback.

ENERGY SHAPING

The dissipation obstacle can be readily avoided by relaxing the second equality constraint in (2.42). This results in only one matching condition

$$g^{\perp}(x)(J(x) - R(x))\frac{\partial H_{\text{a}}}{\partial x}(x) = 0. \tag{2.45}$$

The relaxed constraint implies that the closed-loop will no longer be passive with respect to the original system output y of (2.1) [36].

[2] Note that $g(x)$ is assumed to be full rank such that the matrix $g^T(x)g(x)$ is always invertible [48].

[3] For in-depth analysis and examples see [36].

2.4.3. INTERCONNECTION AND DAMPING ASSIGNMENT (IDA)-PBC

For a PH system (2.1) the IDA-PBC design objective is to obtain a closed-loop system of the form [37]

$$\dot{x} = \big(J_{\mathrm{d}}(x) - R_{\mathrm{d}}(x)\big)\frac{\partial H_{\mathrm{d}}}{\partial x}(x), \tag{2.46}$$

where the desired interconnection and the damping matrices satisfy skew-symmetry and symmetric positive definiteness respectively, i.e.,

$$J_{\mathrm{d}}(x) = -J_{\mathrm{d}}^{T}(x), \quad R_{\mathrm{d}}(x) = R_{\mathrm{d}}^{T}(x), \quad R_{\mathrm{d}}(x) \geq 0.$$

The closed-loop (2.46) can be achieved by the control input:

$$u = \big(g^{T}(x)g(x)\big)^{-1}g^{T}(x)\left(\big(J_{\mathrm{d}}(x) - R_{\mathrm{d}}(x)\big)\frac{\partial H_{\mathrm{d}}}{\partial x} - \big(J(x) - R(x)\big)\frac{\partial H}{\partial x}\right), \tag{2.47}$$

where the desired Hamiltonian $H_{\mathrm{d}}(x)$ and system matrices $J_{\mathrm{d}}(x)$, $R_{\mathrm{d}}(x)$ are obtained by solving the matching condition

$$g^{\perp}\big(J(x) - R(x)\big)\frac{\partial H}{\partial x} = g^{\perp}\big(J_{\mathrm{d}}(x) - R_{\mathrm{d}}(x)\big)\frac{\partial H_{\mathrm{d}}}{\partial x}, \tag{2.48}$$

such that $H_{\mathrm{d}}(x)$ satisfies the desired equilibrium condition (2.34).

Prior to solving the matching condition (2.48), some facts about the choice of the system matrices of (2.46) need to be highlighted [38].

- The desired interconnection matrix $J_{\mathrm{d}}(x)$ and the dissipation matrix $R_{\mathrm{d}}(x)$ can be freely chosen provided they satisfy the skew-symmetry and positive semi-definiteness, respectively.
- The left-annihilator matrix $g^{\perp}(x)$ can be considered as an additional degree of freedom. Hence for a particular problem it can be appropriately chosen to reduce the complexity of the matching condition (2.48).
- The desired Hamiltonian $H_{\mathrm{d}}(x)$ can be partially or completely fixed to satisfy the desired equilibrium condition (2.34).

Using the combination of the stated options there are three main approaches to solve the PDE (2.48) [38].

- *Non-parameterized IDA-PBC* — In this general form, first introduced in [37], the desired interconnection matrix $J_{\mathrm{d}}(x)$ and the dissipation matrix $R_{\mathrm{d}}(x)$ are fixed and the PDE (2.48) is solved for the energy function $H_{\mathrm{d}}(x)$. Among the admissible solutions the one satisfying (2.34) is chosen.
- *Algebraic IDA-PBC* [33] — The desired energy function $H_{\mathrm{d}}(x)$ is fixed thus making (2.48) an algebraic equation in terms of the unknown matrices $J_{\mathrm{d}}(x)$ and $R_{\mathrm{d}}(x)$.
- *Parameterized IDA-PBC* — Here, the structure of the energy function $H_{\mathrm{d}}(x)$ is fixed. This imposes constraints on the unknown matrices $J_{\mathrm{d}}(x)$ and $R_{\mathrm{d}}(x)$, which need to be satisfied by the PDE (2.48) [38].

2.4.4. CONTROL BY INTERCONNECTION

In CbI the controller is also a PH system in the input-state-output form

$$\begin{aligned}\dot{\xi} &= \big(J_c(\xi) - R_c(\xi)\big)\frac{\partial H_c}{\partial \xi}(\xi) + g_c(\xi)u_c, \quad \xi \in \mathbb{R}^{n_c},\\ y_c &= g_c^T(\xi)\frac{\partial H_c}{\partial \xi}(\xi),\end{aligned} \tag{2.49}$$

where $J_c(\xi) \in \mathbb{R}^{n_c \times n_c}$ is the skew-symmetric interconnection matrix, $R_c(\xi) \in \mathbb{R}^{n_c \times n_c}$ is the symmetric dissipation matrix, and $g_c(\xi) \in \mathbb{R}^{n_c \times m_c}$ is the input matrix. The controller Hamiltonian $H_c(\xi) \in \mathbb{R}$ is the energy available in the controller. Generally, the plant (2.1) and the controller (2.49) are connected by a power preserving feedback interconnection [34],

$$\begin{bmatrix} u \\ u_c \end{bmatrix} = \begin{bmatrix} 0 & -I \\ I & 0 \end{bmatrix} \begin{bmatrix} y \\ y_c \end{bmatrix}, \tag{2.50}$$

the resulting closed-loop can be represented as a PH system

$$\begin{aligned}\begin{bmatrix} \dot{x} \\ \dot{\xi} \end{bmatrix} &= \begin{bmatrix} J(x) - R(x) & -g(x)g_c^T(\xi) \\ g_c(\xi)g^T(x) & J_c(\xi) - R_c(\xi) \end{bmatrix} \begin{bmatrix} \frac{\partial H}{\partial x}(x) \\ \frac{\partial H_c}{\partial \xi}(\xi) \end{bmatrix},\\ \begin{bmatrix} y \\ y_c \end{bmatrix} &= \begin{bmatrix} g^T(x) & 0 \\ 0 & g_c^T(\xi) \end{bmatrix} \begin{bmatrix} \frac{\partial H}{\partial x}(x) \\ \frac{\partial H_c}{\partial \xi}(\xi) \end{bmatrix}.\end{aligned} \tag{2.51}$$

For an appropriate controller Hamiltonian $H_c(\xi)$ the PH system (2.1) can be stabilized at x_*, provided the overall closed-loop Hamiltonian $H(x) + H_c(\xi)$ has a minimum at the desired equilibrium.

This can be achieved by ensuring a static relationship between the controller state ξ and the system state x. Generally, this problem is solved by using an invariant function called the *Casimir function* (or simply Casimir) [49] [26].

Casimir's are the state-dependent conserved quantities that are invariant along the system dynamics. They provide a static relationship between the system states and the controller states. In the literature, a prominent choice for the Casimir function is [34]

$$C(x, \xi) = \xi - S(x) = 0, \tag{2.52}$$

where S is some unknown state-dependent function. Since the invariance condition requires $\dot{C}(x, \xi) = 0$, this results in the following partial differential equation

$$\begin{bmatrix} -\frac{\partial S}{\partial x}(x) & I \end{bmatrix} \begin{bmatrix} J(x) - R(x) & -g(x)g_c^T(\xi) \\ g_c(\xi)g^T(x) & J_c(\xi) - R_c(\xi) \end{bmatrix} = 0, \tag{2.53}$$

which can be further simplified to the following chain of equalities [35]

$$\frac{\partial^T S}{\partial x} J(x) \frac{\partial S}{\partial x} = J_c(\xi), \tag{2.54}$$

$$R(x)\frac{\partial S}{\partial x} = 0, \tag{2.55}$$

$$R_c(x) = 0, \tag{2.56}$$

$$J(x)\frac{\partial S}{\partial x} = -g(x)g_c^T(\xi). \tag{2.57}$$

By solving the equalities (2.54)– (2.57) controller matrices and the state-dependent function $S(x)$ can be obtained. From $S(x)$ the controller Hamiltonian can be formulated as $H_c(\zeta) = H_c(S(x))$, using these in (2.49) along with (2.50) results in control-by-interconnection.

2.5. SUMMARY

The PH framework gives an intuitive approach for modeling complex multi-domain systems. This is thanks to the emphasis on the interconnection and energy exchange between subsystems [26]. The model-based control approaches for PH systems explicitly depends on this feature in order to achieve the control objective. For example, PBC for PH system renders the closed loop passive by energy pumping and damping, so as to ensure the system stability [36]. Compared to generic non-linear control techniques, the model-based approaches that explicitly use the PH framework have various attractive properties, e.g. passivity, Lyapunov stability, finite L_2 gain, etc [8, 25, 26].

3

PORT-HAMILTONIAN SYSTEMS IN ADAPTIVE AND LEARNING CONTROL

Generally a nonlinear control method can be classified in a spectrum from model-based to model-free, where adaptation and learning methods typically lie close to model-based and model-free methods, respectively. Various articles and monographs provide a detailed overview of model-based control techniques that are based on port-Hamiltonian (PH) framework, but no survey is specifically dedicated to the learning and adaptive control methods that can benefit from the PH structure. To this end, in this chapter, a comprehensive review of the current learning and adaptive control methodologies that have been developed specifically to PH systems will be provided. After establishing the need for learning in the PH framework, various general machine learning, iterative learning, and adaptive control techniques and their application to PH systems will be given. For each method the essential changes from the general setting due to PH model will be highlighted, followed by a detailed presentation of the respective control algorithm. In general, the advantages of using PH models in learning and adaptive control are: i) Prior knowledge in the form of PH model will speeds up the learning. ii) In some instances new stability or convergence guarantees are obtained by having a PH model. iii) The resulting control laws can be interpreted in the context of physical systems. The chapter will be concluded with brief notes on open research issues.

3.1. INTRODUCTION

Nonlinear control synthesis methods rely to different degrees on the availability of a system model. Examples include model-based control for input affine systems [7], adaptive control [50], or reinforcement learning (RL) [12]. The absolute reliance on the system model in model-based methods and the lack of system knowledge in the learning algorithms, places these two approaches at two extremes of a spectrum. Adaptive control and model-based learning lie close to the model-based and model-free methods, respectively. Each approach has its trade-offs. Nonlinear model-based methods can be

very sensitive to mismatches in the model. On the other side of the spectrum, learning methods achieve the desired control objective by adapting the control law based on the interactions with the system. Thanks to this approach, design objectives such as robustness against model uncertainties and/or parameter variations can be achieved. However, learning methods suffer from several notable drawbacks, such as: slow and non-monotonous convergence and non-interpretability of the learned control law, often arising from the 'learning from scratch' mind-set. This chapter explores how standard learning and adaptive methodologies can benefit from the rich structure of the port-Hamiltonian models, thus 'moving' into the middle regions of the model-based/pure learning spectrum and mitigating some of the hurdles present in the standard control synthesis methodologies.

A few prominent advantages of incorporating PH models in learning are:

- Prior system information in the form of a PH model can significantly improve the rate of convergence.
- New stability or convergence guarantees are obtained in some instances by the virtue of having a PH model.
- The resulting control laws can have an interpretation in terms of physical quantities, such as energy, power, etc.

Simultaneously, there are various improvements to the synthesis of passivity-based controllers (PBC) for PH systems by having learning or adaptive structures. These advantages are:

- Learning can avoid the need for solving complex mathematical equations (PDE's) analytically.
- Performance criteria can be incorporated via learning.
- Novel design problems can be solved which would otherwise be intractable. For example, optimal control problems for PH systems [44] have not been addressed by using solely model-based synthesis methods.

The combination of learning with PH models opens new avenues for solving complex control problems which otherwise would be intractable using either method in isolation. In this chapter, a comprehensive overview of various learning control methods that extensively use the PH system properties will be provided. When applicable, a simple algorithmic (pseudo-code) representation of the learning method will be presented.

This chapter is organized as follows. The need for adaptive and learning control in the PH framework is explained in Section 2. Sections 3 through 5 describe various learning methods that have been introduced for the control of PH systems. Starting with adaptive methods in Section 3, iterative and repetitive control methods are elaborated on in Section 4. Applications of evolutionary strategies for PH control is given in Sections 5. Section 6 concludes the chapter with a discussion on possible open research areas.

3.2. ADAPTIVE AND LEARNING CONTROL IN PH SYSTEMS: A MOTIVATION

Using the stated model-based synthesis methods of Section 2.4, one can acquire a detailed insight into the closed-loop system properties. However, external disturbances and model uncertainties can result in performance issues as illustrated by the following example.

Example 4. *(Continued from Example 1)* Consider a vessel such as boat floating in a water canal. The simplified dynamics for the lateral movement of the vessel are

$$\dot{x} = J\frac{\partial H}{\partial x} + g(x)(u+d), \tag{3.1}$$

where $g(x) = [0 \quad 1]^T$, $d = A\sin(wt)$ represent the waves in the canal modeled as a sinusoidal disturbance of an unknown amplitude A and a known frequency w. The position q and the momentum p constitutes the state vector $x = [q \quad p]^T$. The propulsion u can be used to position the vessel along the horizontal direction. The system Hamiltonian, input matrix and the interconnection matrix are

$$H = \frac{p^2}{2M},\ g = [0 \quad 1]^T,\ J = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

In the absence of external disturbances, the standard ES-DI control input

$$u = -\frac{\partial V_\mathrm{d}}{\partial q} - K_\mathrm{d} y, \tag{3.2}$$

can stabilize (3.1) at the desired equilibrium $(q_*, 0)$, where

$$V_\mathrm{d}(q) = \frac{1}{2}\tilde{q}^T K_\mathrm{p}\tilde{q} = \frac{1}{2}(q-q_*)^T K_\mathrm{p}(q-q_*)$$

is the desired potential energy. Generally, the proportional gain matrix K_p and the positive definite damping-injection matrix $K_\mathrm{d} = K_\mathrm{d}^T$ are chosen by the user. Note that the control law (3.2) is same as a standard PD compensator to stabilize a system at $(q_*, 0)$. Observe that (3.1) is influenced by the external disturbance d which is generally unknown, possibly resulting in performance degradation [51]. □

A dynamical system can typically experience external disturbances and/or exhibit parameter variations during its operational life span. These variations generally require fine tuning of the control parameters so as to achieve the desired objective [50, 52, 53]. Robustness against disturbances and parameter variations is often addressed either by the robust or adaptive control methods. In robust control, the required behavior is achieved by making the closed-loop insensitive to parameter uncertainties, resulting in varying, but acceptable performance. In adaptive control the parameters are modified online, thus ensuring the desired performance. This makes adaptive control approaches more appealing when performance is of high importance. In robust control, the trade-off between performance and robustness is always present [54]. In Section 3, adaptive methods that use the PH framework are discussed.

Many industrial systems often execute the same task multiple times. For repetitive operations, the idea of using previous time-history to improve the performance of the system has a rich literature spanning over three decades [55]. The general idea behind these methods is to minimize a cost function related to the tracking error during each repetition. This is achieved by using the error information of the previous iteration to generate the control action in the current iteration [56]. Iterative control methods adapt the control signal directly, whereas in adaptive control the parameters of the control law are modified. The main objective of the iterative approach is to generate the desired output trajectory without using a priori system information. In [55] six postulates have been given in order to characterize an approach as iterative. If a given method satisfies all the six postulates (see Chapter 1.1 of [55]) then it is classified as *iterative learning control* (ILC). If the initial condition is different in each iteration then the iterative method is called *repetitive control* (RC). Using a uniform mathematical framework, similarity and differences between ILC and RC have been described in [57]. In Section 4, ILC and RC methods for the port-Hamiltonian systems will be given.

The need to fine-tune the controller parameters online so as to satisfy a given performance criterion arises in various control applications. Iterative Feedback Tuning (IFT) [58] is one such prominent online tuning method. IFT learns the optimal parameters by using the measured data from the system. The application of IFT for PH systems will be discussed in Section 4. In the last few decades both offline and online data-driven techniques have been widely used to solve nonlinear control problems [59–61]. A major advantage of the data-driven methods is that they can learn the optimal control parameters for a given cost-function. One such prominent machine-learning approach is Evolutionary Algorithm (EA) [62]. The use of the Evolutionary Algorithm (EA) for PH system control will be discussed in Section 5.

A few of the well-known control methods for port-Hamiltonian systems are listed in Table 3.1. Some of the model-based methods, like ES-DI, and IDA-PBC are briefly introduced in Section 2.4, for other methods references are given in the table. Along with the system dynamics, if the desired closed-loop Hamiltonian and the desired-system structure are known a priori, then the model-based approaches can be used with a relatively low effort. Table 3.1 also provides the list of adaptive and learning methods that are discussed in detail in this chapter. As it will be explained in the following parts of the chapter (Sections 3.3 – 3.5), these methods can be used to address complex control objectives, model uncertainty, input disturbance and performance requirements, etc.

3.3. ADAPTIVE CONTROL METHODS

As shown in Example 4, the need to adapt or modify the control law often arises due to external disturbance, ageing etc. This issues can be solved by using adaptive control methods [50]. The standard adaptive control framework consists of two components: a parameterized control law (3.3) and a parameter update law (3.4)

$$u = \beta(x, \hat{\theta}), \tag{3.3}$$

$$\dot{\hat{\theta}} = \eta(x, \hat{\theta}, e), \tag{3.4}$$

Table 3.1: port-Hamiltonian control methods.

	Model-based methods (Section 2.4)	Learning methods (Section 3.3–3.5)
Stabilization	Energy shaping and damping injection (ES-DI) [34], Interconnection and damping assignment (IDA)-PBC [37], power based method [63], Canonical transformation (CT) [33], Control by interconnection (CbI) or energy-casimir methods [25, 34, 35]	Iterative feedback tuning (IFT) [64], Evolutionary strategy (ES-IDA-PBC) [65], Adaptive control (AC) [66–69]
Tracking	Modified IDA-PBC [70], Canonical transformation [71]	Iterative learning control (ILC) [44], Repetitive control (RC) [72], Adaptive control [67]

where $e = x - x_*$ is the error between the desired and the actual state, θ is an unknown parameter vector and $\hat{\theta}$ is its estimate. Adaptive control methods are broadly classified as indirect and direct control methods [52, 53]. In indirect adaptive control, the unknown plant parameter vector θ is estimated as $\hat{\theta}$ and an appropriate controller is then devised. In direct adaptive control, $\hat{\theta}$ is the estimate of an unknown parameter vector of the control law. A common principle behind the adaptive methods is to treat the estimate as a true value; this is called as the *certainty equivalence principle* [50, 53].

Since many multi-domain systems can be represented in the port-Hamiltonian form, the self-tuning of the PH controller is of interest. In this section the adaptive control framework for port-Hamiltonian systems will be introduced using a simple example for input disturbance compensation.

Example 5. *(Continued from Example 1 and Example 4)* In order to compensate for the external disturbance in (3.1) one can use the adaptive approach [67]. The unknown disturbance can be approximated as

$$d = A\sin(wt) = \theta^T \phi(t), \tag{3.5}$$

where θ^T is the unknown parameter and $\phi(t)$ is a non-constant, known basis function matrix. The unknown parameter vector θ is estimated as $\hat{\theta}$, resulting in an estimate of the disturbance

$$\hat{d} = \hat{\theta}^T \phi(t), \tag{3.6}$$

which is added to the standard EB-DI input (2.41). To ensure the PH structure for the closed-loop, the parameter update law is constrained to be

$$\dot{\hat{\theta}} = -Q\phi^T(t)\, y, \tag{3.7}$$

where y is the system output of (2.2) and $Q = Q^T$ is the update rate matrix. Combining

(2.41) with (3.3) — (3.7), the closed-loop for (2.2) is

$$\begin{bmatrix} \dot{q} \\ \dot{p} \\ \dot{\tilde{\theta}} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & -K_{\mathrm{d}} & \phi(t)Q \\ 0 & -Q\phi^T(t) & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial \bar{H}}{\partial q} \\ \frac{\partial \bar{H}}{\partial p} \\ \frac{\partial \bar{H}}{\partial \tilde{\theta}} \end{bmatrix}, \tag{3.8}$$

where

$$\bar{H} = \frac{p^2}{2M} + \frac{1}{2}\tilde{q}^T K_{\mathrm{p}} \tilde{q} + \frac{1}{2}\tilde{\theta}^T Q^{-1} \tilde{\theta}, \tag{3.9}$$

is the closed-loop Hamiltonian, with $\tilde{\theta} = \hat{\theta} - \theta$. The closed-loop Hamiltonian can be used for the stability analysis. From (3.9) positive semi-definiteness of $\bar{H}$ is evident, i.e., $\bar{H} \geq 0$ now taking the derivative,

$$\frac{d\bar{H}}{dt} = \dot{\bar{H}} = -K_{\mathrm{d}} \left(\frac{p}{M} \right)^2 \leq 0,$$

hence $\bar{H}$ is a decreasing function. Unfortunately, by using Lyapunov theory neither the asymptotic stability nor the parameter convergence can be demonstrated, i.e., $q \to q_*$ and $\hat{\theta} \to \theta$ cannot be shown. This is because of the negative semi-definiteness of $\dot{\bar{H}}$. However, based on the assumption that $\phi(t)$ is non-constant and by further analysis of the closed-loop dynamics based on the LaSalle's extension it can be shown that indeed the convergence happen, i.e., $q \to q_*$ and $\hat{\theta} \to \theta$. This can be shown by using the closed-loop dynamics at the equilibrium

$$\begin{bmatrix} \dot{q} \\ \dot{p} \\ \dot{\tilde{\theta}} \end{bmatrix} = \begin{bmatrix} 0 \\ -K_{\mathrm{p}}(q - q_*) + \phi(t)(\theta - \hat{\theta}) \\ 0 \end{bmatrix},$$

from this it is clear that the largest set that satisfies $\dot{\bar{H}} = 0$ is $[q \quad p \quad \theta]^T = [q^+ \quad 0 \quad \theta^+]^T$, where q^+, θ^+ are some constants values. From the second equation the following equality can be obtained,

$$K_{\mathrm{p}}(q - q_*) = \phi(t)(\theta - \hat{\theta}).$$

As it is assumed that $\phi(t)$ is time varying and non-constant hence the only possibility to satisfy the equality is $q \to q_*$ and $\hat{\theta} \to \theta$. This ensures both the asymptotic stabilization of the system at $(q_*, 0)$ and the convergence of the parameter error [67]. In the classical sense, the non-constant and boundedness assumptions on the basis function $\phi(t)$ as $t \to \infty$ ensures the required persistency of excitation. □

This is one of the simple and most straightforward implementation of adaptive control for PH systems. Because of its plainness adaptive control was one of the first self-tuning methods used in the PH framework [66]. The schematic representation of adaptive control of a PH system is illustrated in Figure 3.1.

The adaptive control law depends on the design objective for e.g., stabilization, simultaneous stabilization [68], tracking [51], input disturbance rejection [67], etc. The parameter update rule is devised to ensure the PH form for the closed-loop. Normally, a candidate Lyapunov function (3.9) is constructed with a minimum where the desired

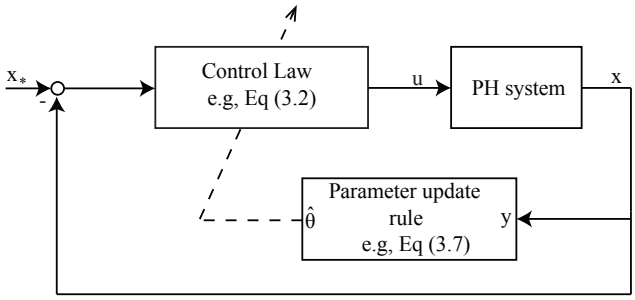


Figure 3.1: Adaptive control schematic for PH systems.

control goal and zero parameter estimation error (i.e., $\theta = \hat{\theta}$) are attained. Adaptive control for port-Hamiltonian system has several advantages:

- As the closed-loop (Figure 3.1) retains the PH structure, various system properties like passivity and finite L_2 gain can be readily inferred;
- As the closed-loop (see (3.8)) is in the PH form it satisfies the inequality (2.17), from which an upper bound on the error signal can be derived [63];
- For the closed-loop (3.8) by using the Hamiltonian (3.9) the asymptotic stability of the equilibrium point x_* was shown by assuming the basis function $\phi(t)$ to be non-constant and bounded as $t \to \infty$. As a generalization of this approach by using the Barbalat's lemma and zero-state detectability, the uniform asymptotic stability of the equilibrium point x_* can be demonstrated, for the proof see Theorem 3 in [67];
- By using the closed-loop Hamiltonian $\bar{H}$ as a Lyapunov function and from the inequality (2.17), the stability of the origin can be shown provided the dissipation matrix R is positive definite (i.e., $R > 0$). For the proof see [68];
- Unlike standard adaptive methods [54, 73], redefinition of the error signal is not required for adaptive control of PH systems [63].

Because of the above advantages, the adaptive framework for PH systems has been used in various applications. Notable examples include power system stabilization [66], adaptive tracking control and disturbance rejection [51, 63, 67], simultaneous stabilization of a set of uncertain PH systems [68, 69], and stabilization of time-varying PH systems [74].

Although the advantages of the adaptive control in the context of port-Hamiltonian systems is evident, its usage is rather limited. This is mainly due to the lack of analysis tools and the detailed study of the closed-loop system properties. For example, the prominent notions in the adaptive control literature such as persistency of excitation, stability analysis using Lyapunov-like functions, adaptive control for identification [53], model-reference adaptive control [50] and extremum seeking control [75] are missing in the context of port-Hamiltonian systems. These missing techniques can be valuable future avenues.

3.4. ITERATIVE AND REPETITIVE CONTROL METHODS

Iterative methods are based on the notion that, for a repetitive task the performance of the system can be improved by using the time-history of the previous executions [56]. The basic working principle of an iterative method is to find an input trajectory $u_\mathrm{d}(t)$ such that it results in the prescribed system output $y_\mathrm{d}(t)$. This objective is achieved by an iterative law

$$u_{i+1}(t) = u_i(t) + \Gamma\underbrace{\left(y_\mathrm{d}(t) - y_i(t)\right)}_{e_i(t)} = u_i(t) + \Gamma\left(e_i(t)\right), \tag{3.10}$$

where $e_i(t)$ is the tracking error in the i^{th} iteration and Γ is an algorithm specific update function [55]. Most of the available iterative methods either use D-Type or its modifications like PD-Type or PID-Type update function. The main advantage of using the standard update rule is that the resulting iterative method can learn the required input without using a priori system information. However, the standard methods for a generic nonlinear system suffer from non-monotonic convergence of the tracking error, i.e., prior to convergence, the tracking error may not decrease in every iteration. Using the adjoint output of the system, a mechanism to ensure monotonic error convergence for repetitive tasks, was proposed in [47]. Unlike the standard iterative methods the adjoint based iterative method requires the complete system information, thus hindering its usability.

As elaborated in Section 2.3.3 the self-adjoint property was shown for controlled-Hamiltonian systems (2.10) [40, 43]. This means that the algorithm-specific update function Γ in (3.10) can be obtained without any a prior system information while guaranteeing monotonic convergence. Using this feature, various iterative methods for controlled-Hamiltonian systems are proposed in [44, 64, 76].

3.4.1. ITERATIVE LEARNING CONTROL (ILC)

To get an ILC law that ensures monotonic error convergence one needs to constrain the update function Γ of equation (3.10) such that the quadratic cost function

$$J(y_i) = \int_{t_0}^{t_1} \left(y_\mathrm{d}(t) - y_i(t)\right)^T Q\left(y_\mathrm{d}(t) - y_i(t)\right) dt, \tag{3.11}$$

is reduced in every iteration, where Q is a positive definite weight matrix [44]. From the principles of functional analysis, Γ needs to be the negative gradient of the cost function

$$\begin{aligned} dJ(y_i) &= -2\langle Q(y_\mathrm{d} - y_i), dy_i\rangle, \\ &= -2\langle Q(y_\mathrm{d} - y_i), d\Sigma(du_i)\rangle, \\ &= -2\langle d\Sigma^*(Q(y_\mathrm{d} - y_i)), du_i\rangle, \end{aligned} \tag{3.12}$$

where $dy_i = d\Sigma(du_i)$ was used for the second identity. This refers to small change in the system output dy_i due to small variation in the system input du_i. For the last identity the equality (2.25) is used. The resulting ILC law is

$$u_{i+1} = u_i + \underbrace{K_i\left(d\Sigma^*(Q(y_\mathrm{d} - y_i))\right)}_{\Gamma}, \tag{3.13}$$

where K_i is a user-defined constant, it can be considered as an extra tuning to ensure monotonic convergence. Although it can be changed after every iteration, in the simulation analysis it was kept constant for all the iterations.

Using (2.28)–(2.30) in (3.13) the update law becomes

$$u_{i+1} = u_i + K_i \left(\mathscr{R} \circ \Sigma(u_i + Q(y_\mathrm{d} - y_i)) - \mathscr{R} \circ \Sigma(u_i) \right), \tag{3.14}$$

as this involves two output trajectories, the update law (3.14) can be split into a two-step iteration law [44] :

$$\begin{aligned} u_{2i+1} &= u_{2i} + \mathscr{R}\left(Q(y_\mathrm{d} - y_{2i})\right), \\ u_{2i+2} &= u_{2i} + K_i \mathscr{R}\left(y_{2i+1} - y_{2i}\right). \end{aligned} \tag{3.15}$$

The working of the iterative control law is illustrated by the following example.

Example 6. Figure 3.2 shows the schematic of a two degree-of-freedom (DOF) manipulator arm. This system can be represented both in the PH form (2.1) and in the controlled-Hamiltonian form (2.10).

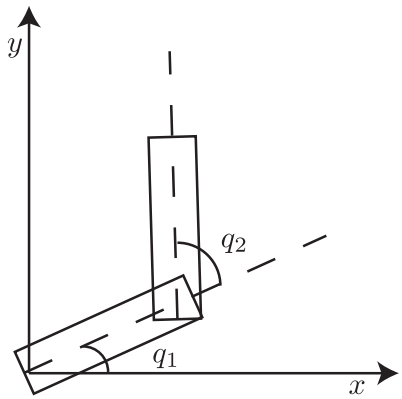


Figure 3.2: A two degree of freedom manipulator arm.

The 2-DOF manipulator arm is characterized by link length l_i, mass of link m_i, center of gravity r_i, and moment of inertia I_i where $i \in \{1,2\}$. The arm's motion is confined to the horizontal plane hence it has no potential energy term. The generalized position of the arm $q = [q_1 \ q_2]^T$ along with the momentum $p = [p_1 \ p_2]^T = M(q)\dot{q}$ constitute the system state $x = [q \quad p]^T$. The mass-inertia matrix is

$$M(q) = \begin{bmatrix} C_1 + C_2 + 2C_3\cos(q_2) & C_2 + C_3\cos(q_2) \\ C_2 + C_3\cos(q_2) & C_2 \end{bmatrix}, \tag{3.16}$$

with the constants C_1, C_2, and C_3 defined as

$$C_1 = m_1 r_1^2 + m_2 l_1^2 + I_1,\ C_2 = m_2 r_2^2 + I_2,\ C_3 = m_2 l_1 r_2.$$

For numerical values of the parameters see [44]. For a desired system output

$$y_{\mathrm{d}}(t) = \begin{bmatrix} 0.5 - 0.5\cos(\pi t) \\ 0.5\cos(\pi t) - 0.5 \end{bmatrix}, \tag{3.17}$$

the ILC law (3.15) is evaluated for 20 iteration, the resulting quadratic cost (3.11) for $Q = I$ is given in Figure 3.3. □

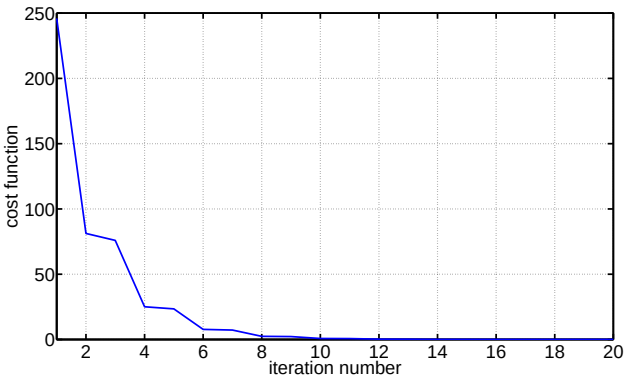


Figure 3.3: Quadratic cost function for ILC.

It must be noted that the ILC update law (3.15) does not depend on the system entities $J, R, H(x,u)$ and the transformation matrix T. They are required to demonstrate the self-adjointness of the given Hamiltonian system, hence the only requirement for the update law (3.15) is that the input-output data samples are obtained from a controlled-Hamiltonian system (2.10). In general ILC for PH system has various advantages, namely,

- The ILC algorithm (3.15) is less sensitive to the measurement noise since it does not require higher order derivatives of the error signal;
- System convergence to a global minimum can be demonstrated, for proof see [44];
- Monotonic convergence of the error signal in the sense of L_2-norm can be shown [44].

Owing to its simplicity and advantages, the iterative law (3.15) has been extended to address different control problems, namely optimal gait generation [77, 78], optimal control [79], optimal control with input constraints [80], and control of nonholonomic Hamiltonian systems [72, 81], etc.

3.4.2. REPETITIVE CONTROL

Using ILC one can readily achieve tracking or disturbance rejection of a periodic signal. However, a major drawback of ILC is that, in every iteration it requires the same initial state of the system. For various applications this requirement can be extremely hard to satisfy. This disadvantage can be overcome by using yet another time-periodic trajectory

learning method called repetitive control (RC). Repetitive control does not rely on the initial condition of the system as it uses a desired trajectory of an infinite time horizon [55].

Repetitive control for a general system relies on the internal model principle, which can be loosely stated as follows: in order to track or reject a periodic signal, a model of this signal must be included in the closed-loop [57]. Almost all the RC algorithms use the internal model principle by having the model of T-periodic signals in the closed-loop [82]. However, repetitive control for PH systems, introduced in [76], does not require the signal model since it is based on the variational symmetry of the Hamiltonian system (2.28)–(2.30). Similar to ILC, repetitive control of a PH system achieves the local minimum for a given quadratic cost function (3.11). The RC framework of [76] is summarized in Algorithm 1. In [76] it is stated that $\Delta\tau_i \to 0$ as $i \to \infty$. Next, this is demonstrated

Algorithm 1 Repetitive control

1: Given the controlled-Hamiltonian system (2.10) and the cost function (3.11)
2: **repeat:** for every iteration i
3: Using the iterative control law (3.15) obtain a suitable control input $u_i(t)$, $t \in [t_0, t_1]$
4: Apply $u_i(t)$ to system (2.10) and observe the system states $x(t)$, $t \in [t_0, t_1]$. For time $t > t_1$ set $u_i(t) = 0$
5: Wait until the system state $x(t)$ converges within a predefined error bound b, for τ_i the shortest time duration s.t. $\|x(\tau_i)\| < b$. Calculate the excess convergence time $\Delta\tau_i = \tau_i - t_1$
6: Evaluate the cost function (3.11)
7: **until the cost function ceases to decrease**

for trajectory tracking of a 2-DOF manipulator arm. The trajectory tracking repetitive control task is evaluated for the desired system output

$$y_d(t) = \begin{bmatrix} 0.5 + 0.5\sin(\pi t) \\ -0.5 - 0.5\sin(\pi t) \end{bmatrix}. \tag{3.18}$$

The resulting cost, using (3.11) with $Q = I$, is given in Figure 3.4.

3.4.3. ITERATIVE FEEDBACK TUNING

Using ILC or RC one can obtain a control input that achieves the tracking of a desired reference. An equally important control objective is online tuning of the feedback control parameters so as to achieve a desired performance criterion irrespective of the model variation or parameter uncertainty. For example, the parameters of a state-feedback controller can be adjusted online via experiments to reduce a quadratic type of cost. One such online tuning method for repetitive task is iterative feedback tuning (IFT). Introduced in [83], IFT has found a wide acceptance as a self-tuning control design method for nonlinear or model-uncertain systems [58, 84]. In [64], by using self-adjointness of the Hamiltonian system, an IFT algorithm has been devised for the systems in controlled-Hamiltonian form.

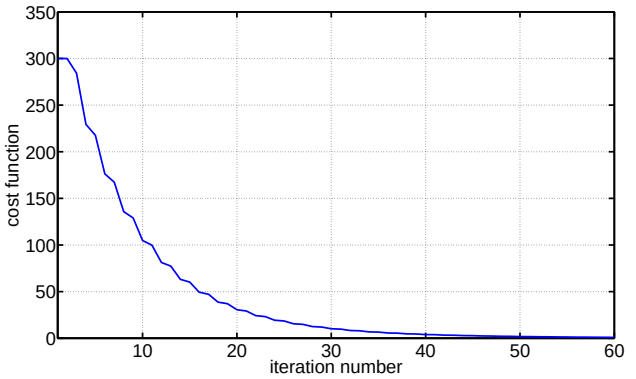


Figure 3.4: Quadratic cost function for RC.

Consider for instance the system (2.10) that has been modified by using energy-shaping feedback as

$$H(x,u) = \frac{1}{2}p^T M^{-1}(q)p + \frac{1}{2}q^T K_p q - u^T q, \tag{3.19}$$

where $x = [q \quad p]^T$ is the system state vector. If the parameter K_p is tunable, then the Hamiltonian (3.19) can be rewritten as a function of the unknown parameter matrix Θ

$$H(x,u,\Theta) = \frac{1}{2}p^T M^{-1}(q)p + \frac{1}{2}q^T \Theta q - u^T q. \tag{3.20}$$

If the design objective is to stabilize the system at the origin by using minimum energy, then the cost function can be formulated as

$$J(q,\Theta) = \sum_{i=1}^{n} \int_0^T \frac{1}{2} Q_1 q_i^2 dt + \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{1}{2} Q_2 \Theta_{i,j}^2, \tag{3.21}$$

where Q_1, Q_2 are weighting constants. The optimal parameter values for $\Theta_{i,j}$ that achieve the minimal cost can be obtained by using the gradient descent method. In [64], a gradient expression has been devised which is similar to (3.12). The gradient of (3.21) also depends on the adjoint of the variational operator. By using the self-adjointness property of the controlled-Hamiltonian, (2.28)–(2.30), an iterative feedback tuning algorithm is given in [64]. The feasibility of the algorithm was demonstrated by stabilizing a 3 DOF mass-spring-damper system at the origin.

3.5. EVOLUTIONARY STRATEGIES

For a nonliner system, obtaining an analytical solution of matching conditions for the IDA-PBC (2.48) (also ES-DI-PBC (2.42)) can be tedious and cumbersome, as illustrated by the following example.

Example 7. *(Continued from Example 1)* For a fully actuated mechanical system (2.2), consider the parameterized IDA-PBC problem. In order to simplify the IDA-PBC problem, as explained in Section 2.4.3, the interconnection and dissipation matrices can be fixed. For this example $J_\mathrm{d} = J$ is chosen and the dissipation matrix is modified as $R_\mathrm{d} = \mathrm{diag}\big([0 \quad D(x)]^T\big)$. Then by using (2.48), the obtained desired Hamiltonian is $H_\mathrm{d} = \phi(q) + \frac{1}{2} p^T M^{-1}(q) p$.

For a valid control action (2.47), the unknown elements $\phi(q), D(x)$ need to be chosen so as to ensure the minimality condition (2.34) and the positive semi-definiteness of R_d. For simple control problems like stabilization of a mass-spring-damper system, the unknown elements $\phi(q)$ and $D(x)$ can be obtained using the approach introduced in [36]. However, as shown in [85] finding the appropriate functions $\phi(q)$ and $D(x)$ for relatively complex complex systems/tasks such as the swing-up and stabilization of pendulum is rather involved. This problem becomes prominent for multi-domain systems or under-actuated systems. □

In order to circumvent the elaborated issues, the authors in [65] parameterized the added energy component $H_\mathrm{a}(x)$ as

$$H_\mathrm{a}(\xi, x) = h^T(P_1, P_2, x) P_0 h(P_1, P_2, x) \tag{3.22}$$

where P_i, for $i \in \{0,1,2\}$, are the unknown symmetric square matrices of dimension $n \times n$. Additionally, P_0 is constrained to be positive definite. A single vector ξ is constructed by vectorizing the unknown elements of P_i. To satisfy the equilibrium condition (2.34), the sigmoid function $h(P_1, P_2, x)$ is chosen such that $h(P_1, P_2, x_*) = 0$. The matrices P_i are assumed to be full rank. The unknown parameters of the added energy function are learnt by using the Evolutionary Algorithms (EA), which is an instance of a stochastic optimization method.

Evolutionary algorithms are inspired by biology. In EA a population of candidate solutions called *parents* are evolved by a *variation operator* and a subset is chosen using *population selection* to form new candidate solutions. The variation operator and the population selection together result in a new *offspring generation*. This process is repeated until a desired objective is achieved. EA can be partitioned in three comparable methods [86], namely, Evolutionary Programming (EP), Genetic Algorithm (GA), and Evolutionary Strategies (ES). The main difference among these methods are in

- problem representation: real-valued vectors, finite state machines, strings of data, etc.
- offspring generation: mutation, cross-over, recombination, elitism, self-adaptation, etc.

For example, in ES the optimization problem is represented by a real-valued vector. The variation operator is often a combination of mutation and self-adaptation of the parents and the population selection mechanism is either described by (λ, μ)-ES or $(\lambda + \mu)$-ES methods, where λ represents the number of offsprings and μ the number of parents. In the first approach, λ offsprings are created by variation of μ parents. Irrespective of the fitness of the new offsprings λ, the original μ parents are discarded prior

to the next iteration. In the second approach, after creating a set of λ offsprings, the worst fit individuals are discarded from the total population of $(\lambda+\mu)$.

A special form of this method is (1+1)-ES where one offspring is created from one parent, both the individuals are compared and the fitter one is reselected as parent for the next iteration. A pseudocode of $(\lambda+\mu)$-ES is given in Algorithm 2 [62].

Algorithm 2 $(\lambda+\mu)$-Evolutionary strategy

1: $k=0$
2: Initialize μ parents: ξ (i.e., $(\xi_1, \xi_2, \cdots, \xi_\mu)$
3: **Evaluate:** Fitness function $\Phi(\xi)$
4: **Repeat**
5: **Execute:** Obtain λ offsprings ξ' from ξ via mutation and self-adaptation.
6: **Evaluate:** Fitness function $\Phi(\xi')$
7: **Update:** Parent $\xi \leftarrow$ select best fit among$(\xi \cup \xi')$
8: **Until** termination condition

In [65] the unknown parameters P_i of the parent are learnt using a variation of Algorithm 2 called the evolutionary strategy IDA-PBC (ES-IDA-PBC). To ensure the reliability of the learnt parameters P_i, the control objective is verified for a set of initial states called the basin of attraction. The problem specific fitness function $\Phi(\xi)$ is computed using the pseudocode provided in Algorithm 3, see [65] for the detailed implementation.

Algorithm 3 $\phi(\xi)$-Fitness function

1: Given a PH system (2.1), user defined performance index function $\rho(x)$, and set of initial conditions $Init$
2: **for 1:** every initial condition indexed by j
3: $Init_j \leftarrow x(t_0)$
4: **for 2:** every time step indexed by i
5: Calculate the added energy $H_a(\xi, x)$ of (3.22)
6: Calculate $u(\xi, x)$ of (2.47)
7: Integrate $\dot{x} = (J(x) - R(x))\frac{\partial H}{\partial x} + g(x)u(\xi, x)$ to obtain the new system state $x(t_i)$
8: Calculate the performance index: $\rho_i = \rho(x(t_i))$ (Generally, a quadratic performance function is used e.g., $\rho(x) = x^T Q x$, for some positive definite weight matrix Q.)
9: **end for 2**
10: $f_j =$ **Sum**(ρ_i) (other operators such as **Max** can be used, depending on the design objective)
11: **end for 1**
12: $f =$ **Sum**(f_j) (other operators can be used to obtain f)
13: **Return** fitness value f

The authors demonstrate their results by stabilizing an underactuated Hamiltonian system (ball and beam) at a desired position.

3.6. DISCUSSION AND CONCLUSIONS

In control engineering, learning techniques are often used to address uncertainties and disturbances. Generally, learning methods are model independent. However, by augmenting learning techniques with the rich structure of the PH models numerous advantages can be obtained.

In Table 3.2 the requirement of prior system information for various learning methods that are discussed in this chapter is listed. With the exception of the evolutionary strategy (ES-IDA-PBC), all other learning methods of Table 3.2 can be used for online control tuning, hence they are capable of handling model and parameter uncertainties. For example, stabilization method like IFT, can learn an optimal feedback law even for an imprecise model. Iterative methods like ILC and RC are independent of system parameters, the only requirement being that the measured data is from a controlled-Hamiltonian system. Their robustness against model and parameter uncertainty is implied. Additionally, ILC and RC can compensate for periodic disturbances in the system.

Table 3.2: port-Hamiltonian learning control methods.

No a prior model information	Uncertain but known model and known control structure	Precise model and known control structure
Iterative learning control (ILC) [44], Repetitive control (RC) [72], Iterative feedback tuning (IFT) [64]	Adaptive control (AC) [66–69]	Evolutionary strategy (ES-IDA-PBC) [65]

In this section the prominent advantages and possible future research for each of the proposed methods will be enumerated.

3.6.1. ADAPTIVE CONTROL

Adaptive methods that are based on the PH formalism yield various advantages, for example, passivity of the closed-loop, finite L_2 gain, etc. However, several prominent tools are missing in the context of adaptive techniques for PH systems. Namely, analysis of persistency of excitation, conditions for parameter and error convergence, closed-loop stability analysis using Lyapunov-like functions, model-reference adaptive control and identification for adaptive control [50, 53], extremum seeking control [75], etc. Extensions of these methods for PH systems can provide an exciting research avenue.

3.6.2. ITERATIVE METHODS

One of the prominent iterative methods is iterative learning control (ILC). This method uses only the measured data, hence it is impervious to parameter uncertainty. This is a standard advantage for any generic ILC method. The major enhancement due to the use of a PH model is monotonic error convergence. While for LTI systems monotonic convergence can been shown by using the system's Markov parameters, for a general nonlinear system, error convergence is still an active research area. However, for a PH

system, thanks to its self-adjointness property, the monotonic error convergence of the iterative and repetitive algorithm is shown to be implicit [44, 76].

Although, the advantages are significant, the analysis and the available tools for iterative control of PH systems are limited. Several areas which can provide a valuable road-map for future research are analysis of system behavior during transient learning [87], repetitive disturbance rejection, extension of iterative methods for multi-domain PH systems, augmenting iterative methods with adaptive techniques [88, 89], etc.

3.6.3. PROOF OF CONVERGENCE

One common link that is missing in the adaptive and learning framework for PH systems is the detailed mathematical proof of convergence. For example, the convergence for ES-IDA-PBC algorithms is yet to be shown. Although the proof of convergence for iterative methods has been given, it only holds for the controlled-Hamiltonian systems and their extension to a generic PH system would be a valuable addition. Parameter convergence of adaptive control methods for PH systems is available, but it is limited to tracking control of a fully actuated mechanical system or to systems with a positive-definite dissipation matrix. A generic parameter convergence and stability proof would be extremely helpful. All these missing mathematical proofs and analysis tools can provide a valuable direction for the future research.

3.6.4. CONCLUSION

In this chapter a comprehensive review of various state-of-the art learning and adaptive control methods for PH systems is provided. An additional benefit in bringing PH models into learning and adaptive control framework is also highlighted. On the learning side, the extra PH structure results in new desirable properties. From the PH side, learning extensions introduce new values. A few notable examples are: i) the use of learning algorithms (such as ES-IDA-PBC) reduces the complexity of the PH control synthesis; ii) iterative and repetitive control for PH systems achieve monotonic error convergence; iii) global asymptotic stability can be shown for adaptive control of PH systems. Thanks to learning, control design problems can be solved, which would otherwise be intractable for PH systems. However, introducing learning does involve a certain degree of compromise particularly in terms of computational cost, and memory.

4

PASSIVITY-BASED CONTROL USING REINFORCEMENT LEARNING

As explained in Chapter 3, adaptive and learning control methods can benefit from the PH structure. This idea will be further extended here. Although the passivity-based control (PBC) is an intuitive way to achieve stabilization of a physical system, in many instances the passivity-based controller is obtained by solving a set of complex partial differential equations (PDEs), which can be extremely difficult. Additionally, there is no proper mechanism to incorporate any performance measure. In order to address these issues in this chapter a set of novel reinforcement learning based approaches will be introduced [1]. Unlike the model-based PBC, here the control law is parameterized by an unknown parameter vector. Then, these parameters are obtained online by using the actor-critic reinforcement learning, thus achieving a near-optimal control policy that satisfy the desired closed-loop objective. Compared to the standard reinforcement learning, the advantage here is that the learned solutions have a physical interpretation. Additionally as the method allows for the class of port-Hamiltonian systems to be incorporated in the actor-critic framework, this enhances the learning speed. The key advantages of combining learning with PBC are: i) the complexity of the control design procedure is reduced as the proposed methods does not require the specification of a global desired Hamiltonian, ii) performance criteria can be readily incorporated, additionally the learning algorithm is robust against model and parameter uncertainties, iii) partial knowledge about the system, given in the form of a PH model, speeds up the learning process, iv) physical meaning can be attributed to the learned control law.

4.1. INTRODUCTION

Passivity-based control has been applied to various electrical, mechanical, electromechanical applications modeled in PH form [25, 90]. Although the passivity-based

[1] Parts of this chapter is from *Olivier Sprangers, Robert Babuska, Subramanya P. Nageshrao, and Gabriel A. D. Lopes. "Reinforcement learning for port-Hamiltonian systems." IEEE Transactions on Cybernetics, vol 45, no. 5 (2015): 1003-1013.*

control of a port-Hamiltonian system provides physical interpretation of the control law, it is stymied by few challenges that hinder its widespread use, they are

1. The geometric structure of the PH system reformulates the PBC problem as a set of partial differential equations (PDEs) (2.42) or nonlinear ordinary differential equations (2.48), which can be challenging to solve [34]

2. Most solutions are founded on stability considerations and overlook performance. This is due to lack of a formal mechanism to incorporate the performance measure.

3. In model-based PBC, parameter and model uncertainty can influence the performance of the PBC law.

The classical model-based PBC strongly relies on the system model. However, for various practical applications obtaining an accurate model of the system is extremely difficult [25]. This makes the passivity-based control synthesis hard to apply. The aim of this chapter is to address these hurdles by incorporating learning. To this end, first the control law for the model-based PBC methods (2.41),(2.47), are parameterized in terms of an unknown parameter vector. Then, by using an appropriate learning approach the parameters can be learned, while simultaneously verifying the matching PDEs namely (2.42) and (2.48).

Amongst various alternative learning methods, reinforcement learning (RL) is a prominent approach that is independent of the system model. RL is a semi-supervised stochastic learning control method [12]. In RL, the controller (also called *policy* or *actor* or *agent*) optimizes its behavior by interacting with the system. For each interaction, the actor receives a numerical reward, which is often calculated as a function of the system's state transition and the control effort. Generally, in control the objective is to maximize the long term cumulative reward, an approximation of this is given by the *value function*. Hence, most of the RL algorithms learn a value function from which the control law can be derived [12, 15].

This combination of solving passivity-based control using RL can be seen as a paradigm shift from the traditional model-based control synthesis for PH systems. As there is no need to synthesize a controller in the closed-form, but instead it can be learned online with proper structural constraints. This brings a number of advantages:

1. It allows to specify the control goal in a "local" fashion through a reward function, without having to consider the entire global behavior of the system. The simplest example to illustrate this idea is to assign a reward function that gives a reward of 1 when the system is in a small neighborhood of the desired goal and 0 everywhere else [12]. The learning algorithm will eventually find a global control policy, whereas in the model-based PBC synthesis one needs to specify a desired global Hamiltonian and system matrices.

2. Learning brings performance in addition to the intrinsic stability properties of PBC. The structure of RL is such that the rewards are maximized, and these can include performance criteria, such as minimal time, energy consumption, etc.

3. Learning offers additional robustness and adaptability since it tolerates model uncertainty in the PH framework. Thus a precise model is not required, a partial knowledge of the system in the form of PH model will suffice.

From the learning point of view, the presented approaches provides a systematic way of incorporating a priori knowledge into the RL problem. The learning methods yield a controller that can be interpreted in terms of physical quantities, such as additional energy that is added to the system or extra damping etc. The same interpretability is typically not found in the traditional RL solutions. In addition one major drawback in RL is the slow and non-monotonic convergence of the learning algorithm. The absence of information about the system forces the algorithm to explore a large part of the state-space prior to finding an optimal policy. By incorporating partial knowledge of the system, the learning speed can be significantly increased [91]. The use of RL algorithms in control has a rich literature spanning over two decades [92–94].

Historically, the trends in control synthesis have oscillated between performance and stabilization. PBC of PH systems is rooted in the stability of multi-domain nonlinear systems. By including learning it is possible to address performance in the PH framework. In the experimental section of the chapter, the developed methods are shown to be robust to parameter and model uncertainties and unmodeled nonlinearities, such as control input saturation. Control input saturation in PBC for PH systems has been addressed explicitly in the literature [71, 85, 95–99]. The developed approach solves the problem of control input saturation on the learning side without the need of augmenting the model-based PBC.

The work presented in this chapter draws an interesting parallel with the application of iterative feedback tuning (IFT) [58] for the PH systems explained in Section 3.4 see also [64] for further details. Both techniques optimize the parameters of the controller online, with the difference that in IFT the objective is to minimize the error between the desired output and the measured output of the system, while the proposed approaches aim at maximizing a reward function, which can be very general. The choice of RL is warranted by its semi-supervised paradigm, as opposed to other traditional fully-supervised learning techniques, such as artificial neural networks or fuzzy approximators where the control specification (function approximation information) is input/output data instead of reward functions. Such fully-supervised techniques can be used within RL as function approximators to represent the value functions and control policies. Evolutionary algorithms, as explained in Section 3.5, can also be considered as an alternative. This is because similar to RL evolutionary algorithms rely on fitness functions that are analogous to the reward function. However, the evolutionary methods are usually computationally involved and therefore generally not suitable for online learning control.

This chapter is organized as follows. In Section 2 components of reinforcement learning and standard actor-critic along with its limitations will be discussed. Energy-balancing actor-critic, an application of reinforcement learning to solve the energy-shaping and damping-injection problem is explained in Section 3. Sections 4 explores the learning algorithms for interconnection and damping assignment (IDA) PBC for PH systems. Section 5 explores the use of RL to solve the control-by-interconnection problem. Section 6 concludes the chapter with a discussion on the design issues and possible solutions.

4.2. REINFORCEMENT LEARNING

The mechanism of performing a task, remembering its outcome and improving upon it when encountered a same or a similar situation can be widely observed in animals. The use of 'reinforcement' is an effective approach to learn in unfamiliar environment. It can be prominently observed in pet animals and as well as in humans. Hence naturally the study of this mechanism called 'learning from interactions' has its origins in psychology. However, the mathematical and computational aspects constitutes a branch of artificial intelligence called 'reinforcement learning' (RL). Since RL is one of the main topic of this thesis the essential background will be given in this section.

There are three main components for any reinforcement learning algorithm namely i) policy, also called actor or controller ii) environment or the process on which the controller acts upon and iii) reward function.

A *stochastic policy* dictates the choice of the action in the form of a probability density function, whereas a *deterministic policy* provides an action that can be applied to the system under consideration. As it will be explained later in the thesis, a state-feedback control law can be considered as synonymous to a deterministic policy. Most of the modern control algorithms are implemented on a computer thus inherently leading to sampling and discretization. Hence in this thesis discrete-time version of the reinforcement learning algorithms are detailed. Continuous-time variants are not discussed as they are not relevant in this work, for details see [100].

In control-theory and robotics, reinforcement learning methods are often used to solve an optimal control problem for a system modelled as Markov decision proces (MDP). An MDP is a tuple $\langle X, U, P, \rho \rangle$ where $X \in \mathbb{S} \subset \mathbb{R}^n$ is the state-space and $U \in \mathbb{U} \subset \mathbb{R}$ is the action-space, a single-input system (i.e., environment) is assumed for the sake of plainness. For discrete space, $P : X \times U \times X \to [0, \infty)$ is the state-transition probability of reaching a state $x_{k+1} = x'$ from the current state $x_k = x$ on applying an action $u_k = u$, where k is the time index. In discrete space setting, the state transition can be represented in compact form as $P^u_{xx'} = P^{u_k}_{x_k x_{k+1}}$. It is important to note that for a continuous-space system, represented by $x_{k+1} = f(x_k, u_k)$, the state transition P can provide only a probability of reaching a state x_{k+1} in the region $X_{k+1} \subset X$ from the current state x_k on applying an action u_k, i.e.,

$$P^u_{xx'} = P(x_{k+1} \subset X_{k+1} | x_k, u_k) = \int_{X_{k+1}} f(x_k, u_k, x') dx'. \tag{4.1}$$

After transiting to the next state x_{k+1} the controller receives a numerical reward $r_{k+1} = \rho(x_{k+1}, u_k)$. This reward a measure of user defined performance and it provides an instantaneous evaluation on the desirability of the action u_k. The reward function ρ defines the goal of the RL problem. It is generally user-defined and considered as a part of the environment. The reward function is assumed to be bounded.

The control action u_k is obtained by a state-to-action map denoted by $u_k = \pi(x_k)$. The general goal in reinforcement learning is to find an optimal map or policy π that maximizes a certain return function $R^\pi(x)$. In this thesis the discounted sum of reward

is used as the return function [2] and it is defined as

$$R^{\pi}(x_0) = \sum_{k=0}^{\infty} \gamma^k r_{k+1} = \sum_{k=0}^{\infty} \gamma^k \rho(x_{k+1}, u_k), \tag{4.2}$$

where x_0 is an initial state and $\gamma \in [0,1)$ is the discount factor. The expected value of the return is given by the state-value function, i.e.,

$$V^{\pi}(x) = E\{R^{\pi}(x|x_0 = x)\} = E\{\sum_{k=0}^{\infty} \gamma^k r_{k+1}|x_0 = x, \pi\}. \tag{4.3}$$

Since the state-value function only depends on the state x, in the literature it is also called as *value function*. It gives the expected desirability of following a policy π starting from a state x.

Similarly, the desirability of performing action u in state x and then following the policy π after the transition is given by the action-value function defined as,

$$Q^{\pi}(x,u) = E\{\sum_{k=0}^{\infty} \gamma^k r_{k+1}|x_0 = x, u_0 = u, \pi\}. \tag{4.4}$$

In the state-value function the first action u is a free variable. The relationship between the two value functions (4.3) and (4.4) is given by,

$$V^{\pi}(x) = E\{Q^{\pi}(x,u)|u = \pi(x)\}. \tag{4.5}$$

After some manipulation the value functions (4.3) and (4.4) can be written in a recursive form called the Bellman equation (for definition and properties see [12]). For an MDP having a discrete-state and finite action space, then the Bellman equation can be solved recursively resulting in value function convergence. From this value-function, a better policy can be obtained. Iteratively following these steps results in optimal value function. From this optimal value-function, an optimal control action can easily be obtained. In the case of state-value function (4.3) this is done by one-step search, whereas an action that gives the highest Q-value is chosen in the case of optimal action-value function (4.4). This is an indirect approach. Alternatively, there exist RL methods that directly search for an optimal control law.

Depending on whether the algorithm search for a value function or a control law or both, RL methods can be broadly classified into three categories [101]

- *Actor-only* algorithms directly search for an optimal control law;
- *Critic-only* methods first learn an optimal value function, from which the control law is obtained by one-step optimization;
- *Actor-Critic* algorithms [19, 102] search explicitly for an optimal control law. Additionally, they also learn a value-function which provides an evaluation on the controller's performance.

[2] Other prominent cost formulation are total reward, average reward, risk-sensitive return function etc, for details see [18].

A major goal of this thesis is to synthesize passivity-based control using the reinforcement learning approach. The useful class of RL algorithms are *Actor-only* and *Actor-Critic* methods. In both these approaches the learning algorithm uses parameterized policies.

By using actor-only methods the policy parameters that can result in an optimal controller can be directly obtained. The major advantage of the actor-only method is that it can cover the complete spectrum of the continuous action space. Since the policy parameters are updated using the gradient descent method the convergence to the optimal policy can be inferred. However, actor-only algorithms generally suffer from high-variance resulting in a slow and non-monotonous convergence [19, 103]. In addition, actor-only methods are computationally involved and therefore not always suitable for real-time learning control. Some of the widely used RL algorithms are based on temporal difference (TD) and they fall in the category of critic-only methods. They are based on learning a state-value function and the control is only implicitly defined via the value function. As this is not the current objective such algorithms are not suitable for this thesis and hence not considered.

The actor-critic methods combine the advantages of both actor-only and critic-only methods. While the actor provides a continuous action using the parameterized policy, the critic provides a low-variance gradient thus resulting in a faster convergence of the actor [19, 103]. The critic evaluates the current policy under consideration and provides an estimate of the value-function. In this thesis temporal-difference, i.e., TD(λ) [12] is used to evaluate the policy. Hence, other prominent approaches such as least-square temporal-difference (LSTD) and residual gradients are not discussed, for details see [103].

Many of the RL methods are developed for systems with finite, discrete state and action space. However, most of the physical systems operate on a continuous state-space and the control law also needs to be continuous. This issue is often addressed by using function approximation, for methods and examples see [12, 15].

The actor-critic consists of two independent parameterized components, the value-function (critic) (4.3) is approximated by using a parameter vector $v \in \mathbb{R}^{n_c}$ and a user defined basis function vector $\phi_c(x) \in \mathbb{R}^{n_c}$ as

$$\hat{V}^{\hat{\pi}}(x, v) = v^T \phi_c(x). \tag{4.6}$$

Similarly, by using the parameter vector $\theta \in \mathbb{R}^{n_a}$, the policy $\hat{\pi}(x, \theta)$ (actor) is approximated as

$$\hat{\pi}(x, \theta) = \theta^T \phi_a(x), \tag{4.7}$$

where $\phi_a(x) \in \mathbb{R}^{n_a}$ is a user-defined basis function vector. At every time step k the critic improves the value function. The actor then updates its parameters in the direction of that improvement. The critic and the actor, i.e., (4.6) and (4.7) are usually defined by a differentiable parameterization such that gradient ascent can be used to update the parameters. This is beneficial when dealing with continuous-action space [104].

For a given continuous state-space system Actor-critic (AC) methods can efficiently learn a continuous control action by using approximated value function and the policy. In terms of AC, the reinforcement learning objective can be expressed as: *find an optimal*

policy $\hat{\pi}(x,\theta)$, such that for each state x, the discounted cumulative reward $\hat{V}^{\hat{\pi}}(x,v)$ is maximized.

The unknown critic parameters are updated using the gradient-ascent rule

$$v_{k+1} = v_k + \alpha_c \delta_{k+1} \nabla_v \hat{V}(x_k, v_k), \tag{4.8}$$

where $\alpha_c > 0$ is the update rate and δ_{k+1} is the temporal difference [12]

$$\delta_{k+1} = r_{k+1} + \gamma \hat{V}(x_{k+1}, v_k) - \hat{V}(x_k, v_k). \tag{4.9}$$

In the above temporal-difference update rule, the critic is updated using the information obtained in two consecutive time steps, while the reward is received often due to the result of a series of steps. As a consequence, the plain TD update results in a slow and sample-inefficient learning. Eligibility traces $e_k \in \mathbb{R}^{n_c}$ offer a way of assigning credit also to states visited several steps earlier and as such can speed up the learning. The update for the critic parameters using eligibility trace is [12],

$$\begin{aligned} e_{k+1} &= \gamma \lambda e_k + \nabla_v \hat{V}(x_k, v_k), \\ v_{k+1} &= v_k + \alpha_c \delta_{k+1} e_{k+1}, \end{aligned} \tag{4.10}$$

where $\lambda \in [0,1]$ is the trace-decay rate.

In order to obtain an optimal policy the reinforcement learning algorithms generally use exploration. This is in order to visit new, unseen parts of the state-action space so as to possibly find better policy parameters. This is achieved by perturbing the policy (4.7) by an exploration term Δu_k. Many techniques have been developed for choosing the type of exploration term (see e.g. [105]). In this chapter gaussian noise with zero mean is used as the exploration Δu_k. The overall control input to the system is

$$u_k = \hat{\pi}(x_k, \theta_k) + \Delta u_k. \tag{4.11}$$

The policy parameter vector θ is increased in the direction of the exploration term Δu_k if the resulting temporal difference δ_{k+1} of (4.9) due to control input (4.11) is positive, otherwise it is decreased. The parameter update rule in terms of the update rate α_a is

$$\theta_{k+1} = \theta_k + \alpha_a \delta_{k+1} \Delta u_k \nabla_\theta \hat{\pi}(x_k, \theta_k). \tag{4.12}$$

Although not as common as it is used for the critic, eligibility traces can also be used for the actor parameter update [102]. However, for the sake of simplicity it is not used in the current work.

4.2.1. STANDARD ACTOR-CRITIC ALGORITHM

A schematic representation of standard actor-critic algorithm is given in Figure 4.1. The critic is used to update both the controller i.e., actor and the value function. Each run of the actor-critic algorithm is called a *trial*. Each trial starts at an initial state x_0 and ends after collecting certain prefixed number of samples. It must be noted that in this thesis discrete-time actor-critic algorithms are used. Hence the time index k is synonymous with the sample index. At the end of the trial the system is reset to the initial state x_0 and the learning iteration is repeated. The actor and critic parameters are initialized at the start of first trial. In this thesis the parameters are initialized to zero, alternatively they can be either randomly initialized or to a previously known parameter. A simple representation of the standard actor-critic scheme is given in Algorithm 4.

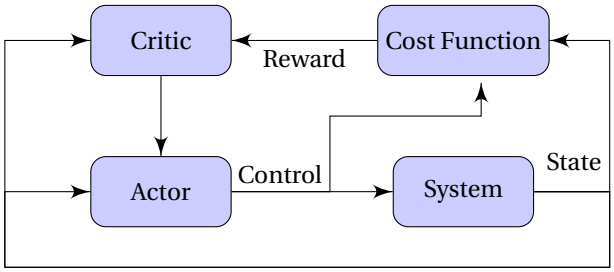


Figure 4.1: Schematic representation of the standard actor-critic algorithm.

4.2.2. FUNCTION APPROXIMATION

For continuous state and action space the value function (4.6) and the policy (4.7) are parameterized using function approximators in terms of the basis function $\phi_c(x)$, and $\phi_a(x)$, respectively. The number of basis function n_c, n_a are generally dictated by the system under consideration and/or the empirical knowledge, whereas the choice of the basis function type is generally characterized by the choice and preference of the designer. Prominent approximators are neural networks, fuzzy approximators, tile coding, local linear regression, radial basis function, etc. In this thesis polynomial, radial and Fourier basis functions [106] are used. This is because of its ease of use, the possibility to incorporate symmetry information about the system under consideration.

The polynomial basis function is a vector of polynomial degree of up to n_c or n_a. Using a multivariate Nth-order (i.e., the order of approximation) the Fourier basis for a system with n dimensions (i.e., the number of states in the system) is

$$\phi_i(\bar{x}) = \cos(\pi c_i^T \bar{x}), \quad i \in \{1,\dots,(N+1)^n\}, \tag{4.13}$$

sin basis functions[3] can be defined similar to cos basis functions, also a combination of sin and cos basis functions can also be used . In (4.13), $c_i \in \mathbb{Z}^n$ implies that all possible $N+1$ integer values, or frequencies they are combined in a vector in $\mathbb{Z}^n$ to create a matrix $c \in \mathbb{Z}^{n\times(N+1)^n}$ containing all possible frequency combinations. For example,

$$c_1 = [0\ 0]^T, \ c_2 = [1\ 0]^T, \ \dots\ , \ c_{(3+1)^2} = [4\ 4]^T, \tag{4.14}$$

for a 3$^{\text{rd}}$-order Fourier basis in 2 dimensions. The state x is scaled as

$$\bar{x}_i = \frac{x_i - x_{i,\min}}{x_{i,\max} - x_{i,\min}} \left(\bar{x}_{i,\max} - \bar{x}_{i,\min}\right) + \bar{x}_{i,\min}, \tag{4.15}$$

for $i = 1,\dots,n$ with $(\bar{x}_{i,\min}, \bar{x}_{i,\max}) = (-1,1)$. Projecting the state variables onto this symmetrical range ensures the same priority for the states with different numerical range. For the Fourier basis function the learning rate is also adjusted,

$$\alpha_{a_i} = \frac{\alpha_{a_b}}{\|c_i\|_2}, \tag{4.16}$$

[3]They are defined as $\phi_i(\bar{x}) = \cos(\pi c_i^T \bar{x}), \ \ i \in \{1,\dots,(N+1)^n\}$.

Algorithm 4 Standard actor-critic algorithm

Input: System such as (2.1), λ, γ, α_{a} for actor, α_{v} for critic. Critic and actor are approximated as (4.6) and (4.7), respectively.

$e_0(x) = 0 \qquad \forall x$
Initialize θ_0, v_0
for # Trials **do**
Initialize x_0
$k \leftarrow 1$
loop until # number of samples
  Execute: Draw action using (4.11), apply the control input to the system for e.g. (2.1), observe next state x_{k+1} and reward $r_{k+1} = \rho(x_{k+1}, u_k)$
  Temporal Difference:
    $\delta_{k+1} = r_{k+1} + \gamma \hat{V}(x_{k+1}, v_k) - \hat{V}(x_k, v_k)$
  Critic Update: total n_{c} parameters
    $e_{k+1} = \gamma \lambda e_k + \nabla_{v_k} \hat{V}(x_k, v_k)$
    Assuming (4.6) the gradient is $\nabla_{v_k} \hat{V}(x_k, v_k) = \phi_{\mathrm{c}}(x_k)$
    $v_{k+1} = v_k + \alpha_{\mathrm{v}} \delta_{k+1} e_{k+1}(x)$
  Actor update: total n_{a} parameters
    $\theta_{k+1} = \theta_k + \alpha_{\mathrm{a}} \delta_{k+1} \Delta u_k \nabla_{\theta_k} u_k(x, \theta)$
    Assuming (4.7) the gradient is $\nabla_{\theta_k} u_k(x_k, \theta) = \phi_{\mathrm{a}}(x_k)$
end loop
end for

for $i = 1, \ldots, (N+1)^n$ with $\alpha_{\mathrm{a_b}}$ is the base learning rate for the actor which needs to be tuned by the designer, where $\alpha_{\mathrm{a_1}} = \alpha_{\mathrm{a_b}}$ this is to avoid division by zero for $c_1 = [0\ 0]^T$. Equation (4.16) implies that parameters corresponding to basis functions with higher (lower) frequencies are learned slower (faster) [106].

4.2.3. S-AC EXAMPLE: PENDULUM SWING-UP AND STABILIZATION

The standard actor-critic Algorithm 4 is used to solve the problem of swinging up and stabilization of an inverted pendulum subject to control saturation. The schematic of the setup under consideration is in Figure 4.2.

The pendulum swing-up is a low-dimensional, but nonlinear control problem. The control task is to learn to swing up and stabilize the pendulum from the initial position pointing down $x_0 = [\pi, 0]^T$ to the desired equilibrium position at the top $x_* = [0, 0]^T$. When the control action is limited to a small range, the system is not able to swing up the pendulum directly, hence it must rather swing back and forth and build up the momentum so as to eventually overcome the local minimum and reach the equilibrium. This problem is commonly used as a benchmark in the RL literature [91] and it has also

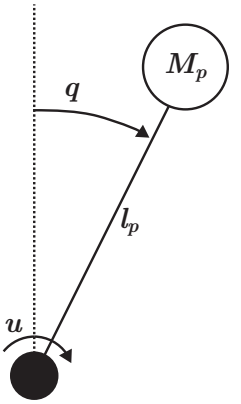


Figure 4.2: Inverted pendulum setup.

been studied in PBC [85]. The equations of motion in the PH form (2.1) are

$$\begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix} = \left(\begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & R_{22} \end{bmatrix} \right) \begin{bmatrix} \nabla_q H(q,p) \\ \nabla_p H(q,p) \end{bmatrix} \begin{bmatrix} 0 \\ \frac{K_\mathrm{p}}{R_\mathrm{p}} \end{bmatrix} u,$$
$$y = \begin{bmatrix} 0 & \frac{K_\mathrm{p}}{R_\mathrm{p}} \end{bmatrix} \begin{bmatrix} \nabla_q H(q,p) \\ \nabla_p H(q,p) \end{bmatrix}, \tag{4.17}$$

with q the angle of the pendulum and p the angular momentum, denote the full measurable state $x = [q\ p]^T$. The damping term R_{22} is

$$R_{22} = b_\mathrm{p} + \frac{K_\mathrm{p}^2}{R_\mathrm{p}}. \tag{4.18}$$

The system Hamiltonian is defined as,

$$H(q,p) = \frac{p^2}{2J_\mathrm{p}} + V(q), \tag{4.19}$$

where the potential energy $V(q)$ is

$$V(q) = M_\mathrm{p} g_\mathrm{p} l_\mathrm{p} (1 + \cos q). \tag{4.20}$$

The model parameters are given in Table 4.1.

The reward function ρ for the Algorithm 4 is defined such that it has its maximum at the desired unstable equilibrium and penalizes other states via,

$$\rho(x,u) = Q_\mathrm{r} \left(\cos(q) - 1 \right) - R_\mathrm{r} p^2, \tag{4.21}$$

where,

$$Q_\mathrm{r} = 25\ ,\ R_\mathrm{r} = \frac{0.1}{J_p^2}. \tag{4.22}$$

Table 4.1: Inverted pendulum model parameters

Model parameters	Symbol	Value	Units
Pendulum inertia	J_p	$1.90 \cdot 10^{-4}$	kgm^2
Pendulum mass	M_p	$5.2 \cdot 10^{-2}$	kg
Gravity	g_p	9.81	$\mathrm{m/s}^2$
Pendulum length	l_p	$4.20 \cdot 10^{-2}$	m
Viscous friction	b_p	$2.48 \cdot 10^{-6}$	Nms
Torque constant	K_p	$5.60 \cdot 10^{-2}$	Nm/A
Rotor resistance	R_p	9.92	Ω

The critic is defined using the Fourier basis function approximation as

$$\hat{V}(x,\upsilon) = \upsilon^T \phi_\mathrm{c}(x), \tag{4.23}$$

with $\phi_\mathrm{c}(x)$ a 3^rd-order Fourier approximators consisting of both sin and cos basis functions resulting in 55 learnable parameters υ in the domain $[q_{min}, q_{max}] \times [p_{min}, p_{max}] = [-\pi, \pi] \times [-8\pi J_p, 8\pi J_p]$. Over the similar domain the policy is approximated using a linear in parameter function approximation

$$\hat{\pi}(x,\theta) = \theta^T \phi_\mathrm{a}(x), \tag{4.24}$$

where θ is the parameter to be learned. The resulting control input is,

$$u_k = \hat{\pi}(x,\theta) + \Delta u_k, \tag{4.25}$$

where the exploration term Δu_k is zero-mean gaussian random noise. The S-AC Algorithm 4 is evaluated using the simulation parameters given in Table 4.2. The learning rate has been tuned such that no failed simulations occur.

Figure 4.3 shows the maximum, average and the minimum of the learning curve obtained from 50 consecutive simulations of the S-AC Algorithm 4. The standard actor-critic generally needs around 60 trials before a near-optimal policy is obtained.

Faster convergence for the S-AC can be achieved by increasing the learning rate $\alpha_{\mathrm{a},\theta}$. However, when using a higher learning rates a considerable number of failures were observed. As it will be compared later with the developed methods, the S-AC algorithm was found to use a much larger number of basis functions. In addition to slower convergence and higher number of basis functions, one major drawback S-AC is the lack physical interpretability of the learned control law. Additionally, prior knowledge of the system cannot be readily incorporated into the standard actor-critic method.

4.3. ACTOR-CRITIC ALGORITHM FOR STANDARD-PBC

As illustrated in the Example 7 solving the matching condition (2.42) for the standard PBC can be tedious and cumbersome. In this section an approach to parameterize the standard-PBC closed-loop Hamiltonian is developed. The unknown parameter vector is then learned using a variation of the standard actor-critic Algorithm 4 called energy balancing actor-critic (EBAC).

Table 4.2: Simulation parameters

Simulation parameters	Symbol	Value	Units
Number of trials	–	200	-
Trial duration	T_t	3	s
Sample time	T_s	0.03	s
Number of samples	N_s	100	-
Decay rate	γ	0.97	-
Eligibility trace decay	λ	0.65	-
Exploration variance	σ^2	1	-
Max control input	u_max	3	V
Min control input	u_max	-3	V
Learning rate of critic	α_c	0.05	-
Learning rate of actor	$\alpha_{\mathrm{a}_\theta}$	5×10^{-5}	-

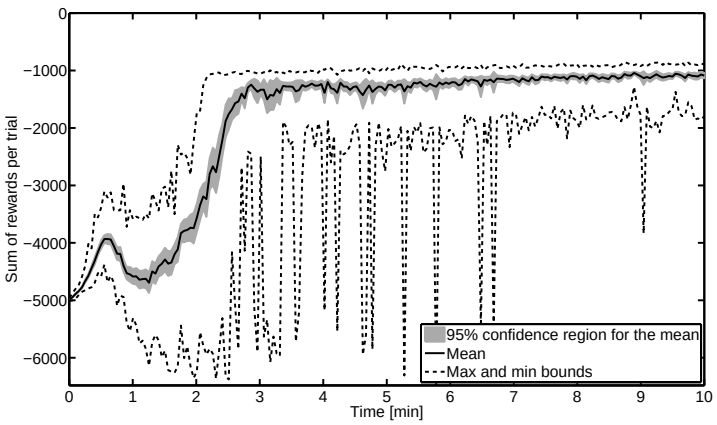


Figure 4.3: Learning curve for the Standard Actor-Critic method for 50 simulations.

4.3.1. PARAMETERIZED HAMILTONIAN FOR STANDARD PBC

As explained in Section 2.4.2 due to the dissipation obstacle (2.44) the standard PBC, (energy-balancing and damping-injection) can be used if and only if the system is controllable by using a finite amount of added energy $H_a(x)$. This implies that only the dynamics with no explicit dissipation can be regulated. Generally, in mechanical systems (2.2) the dissipation is associated with momentum p whereas position dynamics have no explicit dissipation term. Hence, by using standard PBC one can stabilize the momentum p only at the origin while the position q can be regulated to any desired state say q_d. In terms of the system's Hamiltonian, this means that only the potential energy of a mechanical system can be shaped, whereas the kinetic energy term remains unaltered. For example, the desired Hamiltonian for a mechanical system consists of the original kinetic energy term $T(x)$ in (2.3) and the shaped potential energy term $V_d(x)$, i.e.,

$$H_d(x) = T(x) + V_d(x). \tag{4.26}$$

For a general physical system, the desired Hamiltonian consists of shapable (s) and non-shapable (ns) components

$$H_d(x) = H_{ns}(x) + H_s(x). \tag{4.27}$$

A procedure to separate the non-shapable and shapable components of the state-vector can be devised by reformulating the PDE (2.42) in terms of the desired closed-loop energy $H_d(x)$, by applying (2.35)

$$\underbrace{\begin{bmatrix} g^{\perp}(x)F^T(x) \\ g^T(x) \end{bmatrix}}_{A(x)} (\nabla_x H_d(x) - \nabla_x H(x)) = 0, \tag{4.28}$$

and reformulating the kernel of $A(x)$ as

$$\ker(A(x)) = \{N(x) \in \mathbb{R}^{n\times b} : A(x)N(x) = 0\}, \tag{4.29}$$

such that (4.28) reduces to

$$\nabla_x H_d(x) - \nabla_x H(x) = N(x)a, \tag{4.30}$$

with $a \in \mathbb{R}^b$. Suppose that the state vector x can be written as $x = [w^T \; z^T]^T$, where $z \in \mathbb{R}^c$ and $w \in \mathbb{R}^d$, $c + d = n$ corresponding to the zero and non-zero elements of $N(x)$ such that

$$\begin{bmatrix} \nabla_w H_d(x) \\ \nabla_z H_d(x) \end{bmatrix} - \begin{bmatrix} \nabla_w H(x) \\ \nabla_z H(x) \end{bmatrix} = \begin{bmatrix} N_w(x) \\ 0 \end{bmatrix} a. \tag{4.31}$$

The matrix $N_w(x)$ can be assumed of rank d, which is always true for fully actuated mechanical systems. It is clear that $\nabla_z H_d(x) = \nabla_z H(x)$, which is the matching condition, and hence $\nabla_z H_d(x)$ cannot be chosen freely. Thus, only the desired closed-loop energy gradient vector $\nabla_w H_d(x)$ is free for assignment.

Normally, in the standard PBC framework, the shapable component $H_d(w)$ of the desired energy term of (4.27) is chosen to be quadratic [36]. Instead, in this chapter, the desired Hamiltonian is formulated as a linearly parameterized function approximator.

The "hat" symbol stands for the approximation, e.g. $\hat{H}_\mathrm{d}$ is the approximation of the desired Hamiltonian H_d.

$$\hat{H}_\mathrm{d}(x,\xi) = H_\mathrm{ns}(x) + \xi^T \phi_\mathrm{es}(x), \tag{4.32}$$

where $\xi \in \mathbb{R}^{n_\mathrm{es}}$ is the unknown parameter vector and $\phi_\mathrm{es}(x) \in \mathbb{R}^{n_\mathrm{es}}$ is a user defined basis function vector. In order to completely characterize (4.32) the unknown parameter vector

$$\xi = [\xi_1\ \xi_2\ \cdots\ \xi_{n_\mathrm{es}}]^T,$$

must be obtained.

It is important to constrain the local minima of $\hat{H}_\mathrm{d}(x,\xi)$ to be the desired equilibrium x_*, via the choice of the basis functions. For a mechanical system this can be done by constraining $\phi_\mathrm{es}(x)$ to be zero at the desired equilibrium q_d. This will locally ensure the minimality condition (2.34) at $x_\mathrm{d} = (q_\mathrm{d}, 0)$.

4

4.3.2. ENERGY BALANCING ACTOR-CRITIC

By using the parameterized energy function (4.32) and (2.35) in (2.41), the control policy is obtained in terms of an unknown parameter vector ξ and basis function $\phi_\mathrm{es}(x)$ as

$$\begin{aligned} u(x,\xi) &= g^\dagger(x)\big(J(x)-R(x)\big)\big(\nabla_x \hat{H}_\mathrm{d}(x,\xi) - \nabla_x H(x)\big) \\ &\quad - K(x) g^T(x) \nabla_x \hat{H}_\mathrm{d}(x), \\ &= g^\dagger(x)\big(J(x)-R(x)\big)\left(\xi^T \frac{\partial \phi_\mathrm{es}}{\partial x}(x) - \nabla_x H(x)\right) \\ &\quad - K(x) g^T(x) \nabla_x \hat{H}_\mathrm{d}(x), \\ &= \hat{\pi}(x,\xi), \end{aligned} \tag{4.33}$$

where the policy $\hat{\pi}(x,\xi)$ is a function of the user defined damping injection matrix $K(x)$.

The damping-injection matrix $K(x)$ can be considered as an additional degree of freedom. It can be parameterized by using an unknown parameter vector ψ and a user defined basis function vector $\phi_\mathrm{di}(x) \in \mathbb{R}^{n_\mathrm{di}}$ as

$$\big[\hat{K}(x,\psi)\big]_{ij} = \sum_{l=1}^{n_\mathrm{di}} [\psi]_{ijl} \big[\phi_\mathrm{di}(x)\big]_l, \tag{4.34}$$

where $[\psi]_{ij} \in \mathbb{R}^{n_\mathrm{di}}$ is constrained to verify

$$[\psi]_{ij} = [\psi]_{ji}, \tag{4.35}$$

so that the required symmetry condition of $K(x)$ is satisfied. Using the approximated $\hat{K}(x)$ matrix in the control law (4.33) results in two unknown parameter vectors, namely ξ (with n_es unknown entries) and ψ (with $m(m+1)n_\mathrm{di}/2$ unknown entries).

Most of the physical systems are subject to the control input saturation problem. The saturation constraints can be easily addressed while devising the standard-PBC law (4.33). This is done by considering a generic saturation function $\varsigma : \mathbb{R}^m \to S,\ S \subset \mathbb{R}^m$, such that

$$\varsigma(u(x)) \in S\ \forall u, \tag{4.36}$$

where S is the set of valid control inputs. The control action with exploration (4.11) becomes

$$u_k = \varsigma\left(\hat{\pi}(x_k, \xi_k, \psi_k) + \Delta u_k\right), \tag{4.37}$$

where Δu_k is a zero-mean gaussian noise. Due to the saturation constraint the exploration term to be used in the actor update (4.12) must be adjusted

$$\Delta \bar{u}_k = u_k - \hat{\pi}(x_k, \xi_k, \psi_k). \tag{4.38}$$

Note that due to this step, the exploration term $\Delta \bar{u}_k$ used in the learning algorithm is no longer drawn from the chosen distribution present in Δu_k. Furthermore, the gradients of the saturated policy will be

$$\nabla_\xi \varsigma(\hat{\pi}) = \nabla_{\hat{\pi}} \varsigma(\hat{\pi}) \nabla_\xi \hat{\pi}, \tag{4.39}$$

$$\nabla_{[\Psi]_{ij}} \varsigma(\hat{\pi}) = \nabla_{\hat{\pi}} \varsigma(\hat{\pi}) \nabla_{[\Psi]_{ij}} \hat{\pi}. \tag{4.40}$$

Although not explicitly detailed, the (lack of) differentiability of the saturation function ς can be problematic in the computation of the gradient of (4.39) and (4.40). For a traditional saturation in $u_i \in \mathbb{R}$ of the form $\max(u_{min}, \min(u_{max}, u_i))$, i.e. assuming each input u_i is bounded by u_{min} and u_{max}, then the gradient of ς is the zero matrix outside the unsaturated set S (i.e. when $u_i < u_{min}$ or $u_i > u_{max}$). In this work the gradient of ς at the boundary is assumed to be unity. For other types of saturation the function $\nabla\varsigma$ must be computed. The actor parameters $\xi_k, [\Psi_k]_{ij}$ are updated respecting the saturated policy gradients. For the parameters of the desired Hamiltonian ξ is updated as,

$$\xi_{k+1} = \xi_k + \alpha_{a,\xi} \delta_{k+1} \Delta \bar{u}_k \nabla_\xi \varsigma\left(\hat{\pi}(x_k, \xi_k, \Psi_k)\right), \tag{4.41}$$

and the parameters of the desired damping are updated as

$$[\Psi_{k+1}]_{ij} = [\Psi_k]_{ij} + \alpha_{a,[\Psi]_{ij}} \delta_{k+1} \Delta \bar{u}_k \nabla_{[\Psi_k]_{ij}} \varsigma\left(\hat{\pi}(x_k, \xi_k, \Psi_k)\right), \tag{4.42}$$

where $(i, j) = 1, \ldots, m$, while observing (4.35). Algorithm 5 gives the entire Energy-Balancing Actor-Critic (EBAC) algorithm with input saturation. For the EBAC algorithm the critic is approximated, similar to S-AC, using (4.6). A block diagram representation of the control Algorithm 5 is given in Figure 4.4.

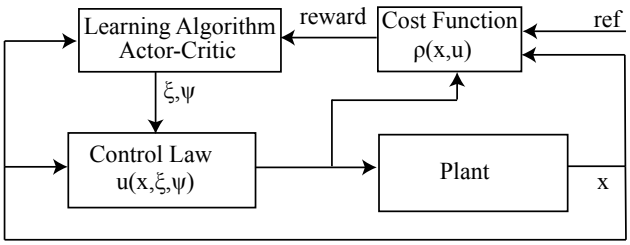


Figure 4.4: Block diagram representation of the energy-balancing actor critic (EBAC) algorithm.

Algorithm 5 Energy-Balancing Actor-Critic

Input: System (2.1), λ, γ, α_{a} for each actor (i.e., $\alpha_{\mathrm{a}\xi}$ and $\alpha_{\mathrm{a}\psi}$), α_{c} for critic. The value function is approximate using (4.6) and the policy is approximated as in (4.33) and (4.34).

$e_0(x) = 0 \qquad \forall x$
Initialize θ_0, ξ_0, ψ_0
for number of trials **do**
Initialize x_0
$k \leftarrow 1$
loop until number of samples
Execute: Draw an action using (4.33), apply the control input $u_k = \varsigma\left(\hat{\pi}(x_k, \xi_k, \psi_k) + \Delta u_k\right)$ to (2.1), observe the next state x_{k+1} and the reward $r_{k+1} = \rho(x_{k+1})$
Temporal Difference:
$\delta_{k+1} = r_{k+1} + \gamma \hat{V}(x_{k+1}, \upsilon_k) - \hat{V}(x_k, \upsilon_k)$
Critic Update:
for $i = 1, \ldots, n_{\mathrm{c}}$ **do**
$e_{i,k+1} = \gamma\lambda e_{i,k} + \nabla_{\upsilon_{i,k}} \hat{V}(x_k, \upsilon_k)$
$\upsilon_{i,k+1} = \upsilon_{i,k} + \alpha_{\mathrm{c}} \delta_{k+1} e_{i,k+1}(x)$
end for
Actor update:
for $i = 1, \ldots, n_{\mathrm{es}}$ **do**
$\xi_{i,k+1} = \xi_{i,k} + \alpha_{\mathrm{a}\xi} \delta_{k+1} \Delta\bar{u}_k \nabla_{\xi_{i,k}} \varsigma\left(\hat{\pi}(x_k, \xi_k, \psi_k)\right)$
end for
for $i = 1, \ldots, m(m+1)n_{\mathrm{di}}/2$ **do**
$\psi_{i,k+1} = \psi_{i,k} + \alpha_{\mathrm{a}\psi} \delta_{k+1} \Delta\bar{u}_k \nabla_{\psi_{i,k}} \varsigma\left(\hat{\pi}(x_k, \xi_k, \psi_k)\right)$
end for
end loop
end for

4.3.3. EXAMPLE I: PENDULUM SWING-UP

For the pendulum system (4.17), the desired Hamiltonian (4.32) is,

$$\hat{H}_\mathrm{d}(x,\xi) = \frac{p^2}{2J_\mathrm{p}} + \xi^T \phi_\mathrm{es}(q)\,. \tag{4.43}$$

Only the potential energy can be shaped, denoted by $\hat{V}_\mathrm{d}(q,\xi) = \xi^T \phi_\mathrm{es}(q)$. Furthermore, as there is only one input, $\hat{K}(x,\Psi)$ becomes a scalar

$$\hat{K}(x,\psi) = \psi^T \phi_\mathrm{di}(x)\,. \tag{4.44}$$

Thus, control law (4.33) results in

$$\begin{aligned} u(x,\xi,\psi) &= g^\dagger F \begin{bmatrix} \xi^T \nabla_q \phi_\mathrm{es} - \nabla_q V \\ 0 \end{bmatrix} - \hat{K} g^T \begin{bmatrix} \xi^T \nabla_q \phi_\mathrm{es} \\ J_\mathrm{p}^{-1} p \end{bmatrix} \\ &= -\frac{R_\mathrm{p}}{K_\mathrm{p}} \left(\xi^T \nabla_q \phi_\mathrm{es} + M_\mathrm{p} g_\mathrm{p} l_\mathrm{p} \sin(q) \right) \\ &\quad - \frac{K_\mathrm{p}}{R_\mathrm{p}} \psi^T \phi_\mathrm{di} \dot{q}, \end{aligned} \tag{4.45}$$

which is defined as the policy $\hat{\pi}(x,\xi,\psi)$, where $g^\dagger$ is the pseudo-inverse $g^\dagger = (g^T g)^{-1} g^T$. The resulting two actor updates are

$$\xi_{k+1} = \xi_k + \alpha_{\mathrm{a},\xi} \delta_{k+1} \Delta \bar{u}_k \nabla_\xi \varsigma \left(\hat{\pi}(x_k, \xi_k, \psi_k) \right), \tag{4.46}$$

$$\psi_{k+1} = \psi_k + \alpha_{\mathrm{a},\psi} \delta_{k+1} \Delta \bar{u}_k \nabla_\psi \varsigma \left(\hat{\pi}(x_k, \xi_k, \psi_k) \right), \tag{4.47}$$

for the desired potential energy $\hat{V}_\mathrm{d}(q,\xi)$ and the desired damping $\hat{K}(x,\psi)$, respectively.

The parameters were all initialized with zero vectors of appropriate dimensions, i.e. $(\theta_0,\, \xi_0,\, \psi_0) = 0$. The EBAC Algorithm 5 was first run with the system simulated in Matlab for 200 trials of three seconds each (with a near-optimal policy, the pendulum needs approximately one second to swing up). Each trial begins in the initial position $x_0 = [\pi, 0]^T$. Using the actor learning rate $\alpha_{\mathrm{a_b},\xi} = 1 \times 10^{-10}$ for the desired potential energy $\hat{V}_\mathrm{d}(q,\xi)$ and $\alpha_{\mathrm{a_b},\psi} = 0.2$ for the damping-injection term $\hat{K}(x,\psi)$ the EBAC Algorithm 5 was evaluated 50 times using the reward function (4.21) and simulation parameters from the Table 4.2. Figure 4.5 shows the average learning curve obtained after 50 simulations. The algorithm shows good convergence and on an average needs about 2 minutes (40 trials) to reach a near-optimal policy. The initial drop in performance is caused by the zero-initialization of the value function (critic), which is too optimistic compared to the true value function. Therefore, the controller explores a large part of the state space and receives a lot of negative rewards before it learns the true value of the states. Note that the learning rates were tuned such that there are no failed experiments, i.e. all 50 consecutive simulations converge to a near-optimal policy. It was observed that if the learning rates are set higher, faster learning can be achieved, however, at the cost of a higher number of failed experiments. A simulation using the policy learned in a typical experiment is given in Figure 4.6a. As can be seen, the pendulum swings back once to build up the momentum and eventually gets to the equilibrium. The desired Hamiltonian $\hat{H}_\mathrm{d}(x,\xi)$ (4.43),

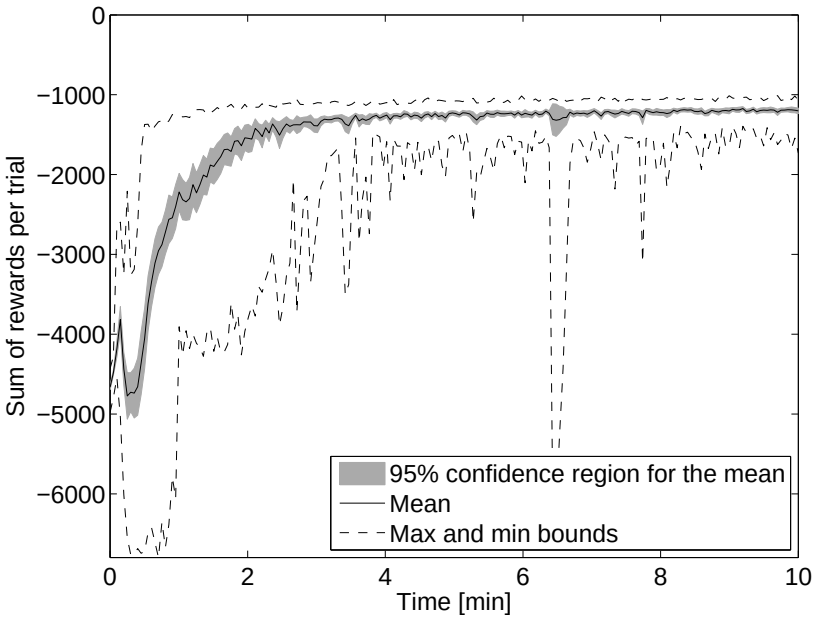
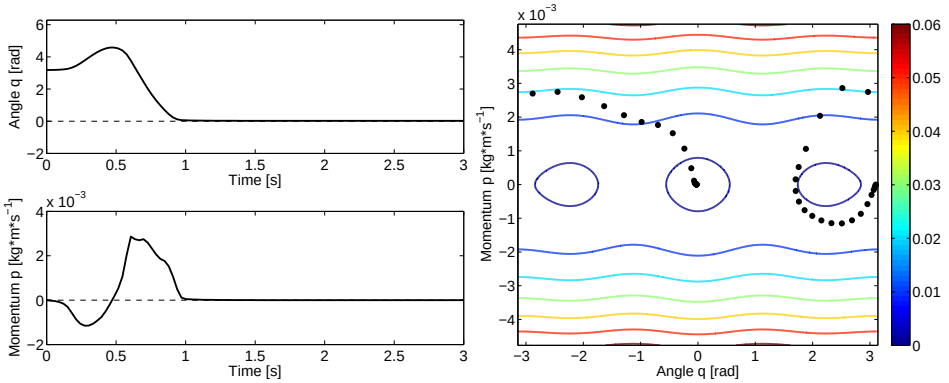


Figure 4.5: Learning curve for the proposed Energy-Balancing Actor-Critic method for 50 simulations.

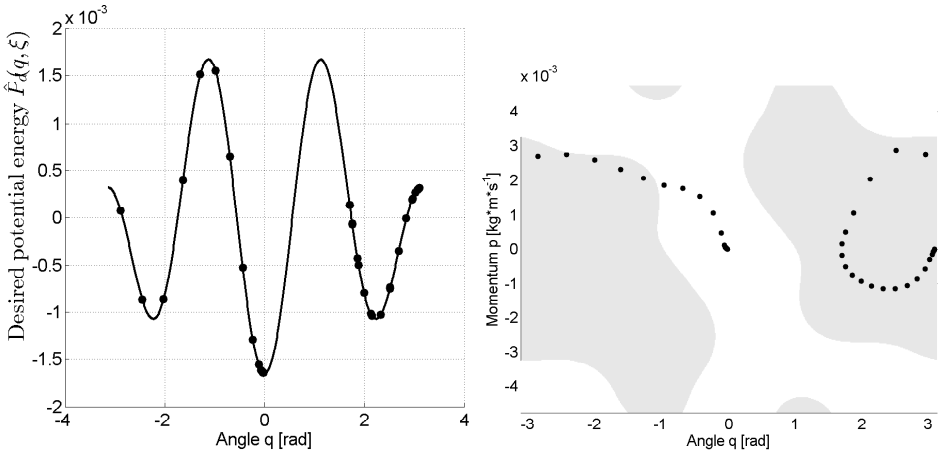


(a) Simulated response (b) Desired Hamiltonian

Figure 4.6: Simulation results for the angle q (a, top), momentum p (a, bottom) and the desired closed-loop Hamiltonian $H_{\mathrm{d}}(x,\xi,\psi)$ (b) including the simulated trajectory (black dots) using the policy learned.

acquired through learning, is given in Figure 4.6b. There are three minima, of which one corresponds to the desired equilibrium. The other two equilibria are undesirable wells that come from the shaped potential energy $\hat{V}_{\mathrm{d}}(q,\xi)$ (Figure 4.7a). These minima are the result of the algorithm trying to swing up the pendulum in a single swing, which is not possible due to the saturation. Hence, a swing-up strategy is necessary to avoid staying

in these wells. The number of these undesirable wells is a function of the control saturation and of the number of basis functions chosen to approximate $\hat{V}_\mathrm{d}(q,\xi)$. The learned damping $\hat{K}(x,\psi)$ (Figure 4.7b) is positive (white) towards the equilibrium thus extracting energy from the system, while it is negative (gray) in the region of the initial state. The latter corresponds to pumping energy into the system, which is necessary to build up the momentum for the swing-up and to escape the undesirable wells of $\hat{V}_\mathrm{d}(q,\xi)$ (see discussion of expression (4.34)). A disadvantage is that control law (4.33), with the suggested basis functions, is always zero for the set $\Omega = \{x \mid x = (0+j\pi, 0),\ j = 1,2,\ldots\}$ which implies that it is zero not only at the desired equilibrium, but also at the initial state x_0. During learning this is not a problem since there is constant exploration, but after learning the system should not be initialized at exactly x_0 otherwise it will stay in this set. This can be solved by initializing with a small perturbation ϵ around x_0. In real-life systems it will also be less of a problem as noise is generally present in the sensor readings.

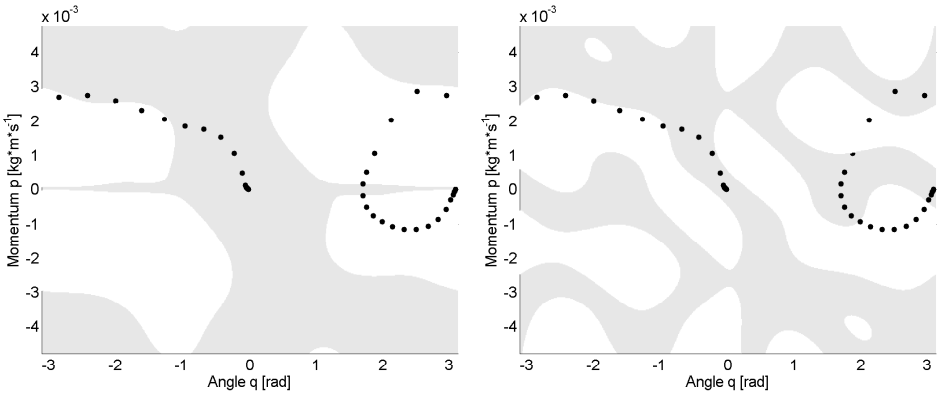


(a) $\hat{V}_\mathrm{d}(q,\xi)$ (b) $\mathrm{sgn}\left(\hat{K}(x,\psi)\right)$

Figure 4.7: Desired potential energy (a) and desired damping (b) (gray: negative; white: positive) for a typical learning experiment. The black dots indicate the individual samples of the simulation in Figure 4.6a.

STABILITY OF THE LEARNED CONTROLLER

Due to the control saturation the target dynamics might not satisfy (2.39). Hence, to conclude the local stability of x_* based on passivity first calculate $\dot{\hat{H}}_\mathrm{d}(x,\xi)$ for the unsaturated case (Figure 4.8a) and the saturated case ($\dot{\hat{H}}_{\mathrm{d,sat}}(x,\xi)$) and compute the sign of the difference (Figure 4.8b). By looking at Figure 4.8b, it appears that $\exists\delta \subset \mathbb{R}^n : |x - x_*| < \delta$ such that $\dot{\hat{H}}_{\mathrm{d,sat}}(x,\xi) = \dot{\hat{H}}_\mathrm{d}(x,\xi)$. It can be seen from Figure 4.8b that such a δ exists, i.e., a small gray region around the equilibrium x_* exists. Hence, by using $\dot{\hat{H}}_\mathrm{d}(x,\xi)$ around x_* the stability using passivity can be assessed. Extensive simulations show that similar behavior is always achieved.



(a) $\operatorname{sgn}\left(\dot{\hat{H}}_{\mathrm{d}}(x,\psi)\right)$ (b) $\dot{\hat{H}}_{\mathrm{d,diff}}(x,\psi)$

Figure 4.8: Signum of $\dot{\hat{H}}_{\mathrm{d}}(x,\psi)$ (a) indicating positive (white) and negative (gray) regions and (b) $\dot{\hat{H}}_{\mathrm{d,diff}}(x,\psi) = \operatorname{sgn}\left(\dot{\hat{H}}_{\mathrm{d}}(x,\psi) - \dot{\hat{H}}_{\mathrm{d,sat}}(x,\psi)\right)$ indicating regions where $\dot{\hat{H}}_{\mathrm{d}}(x,\psi) = \dot{\hat{H}}_{\mathrm{d,sat}}(x,\psi)$ (gray) and $\dot{\hat{H}}_{\mathrm{d}}(x,\psi) \neq \dot{\hat{H}}_{\mathrm{d,sat}}(x,\psi)$ (white). Black dots indicate the simulated trajectory.

REAL-TIME EXPERIMENTS

Using the physical setup shown in Figure 4.2, 20 learning experiments were run using identical settings as in the simulations. There were no failures in this set of experiments (i.e., the learning process always converged to a near-optimal policy capable of swinging up and stabilizing the pendulum). The results are illustrated in Figure 4.9. The algorithm shows slightly slower convergence - about 3 minutes of learning (60 trials) to reach a near-optimal policy instead of 40 - and a less consistent average when compared to Figure 4.5. This can be attributed to a combination of model mismatch and the symmetrical basis functions (note that it is not possible to incorporate non-symmetrical friction that is present in the real system). Overall, the performance can be considered good when compared to the simulation results. Also, the same performance dip is present which can again be attributed to the optimistic value function initialization.

For the EBAC controller (4.33), physical meaning can be attributed to the learned control law. This is not possible for the standard actor-critic controller in (4.24). Additionally, when compared to the standard actor-critic (see Section 4.2.3) that needed more than 100 policy parameters, energy balancing actor-critic (EBAC) can achieve the same control objective with only 40 control parameters. Also, it can be observed from Fig.4.5, that the proposed methods can learn the control policy much faster when compared to the standard actor-critic algorithm (see Fig.4.3). An improvement of 30-50% in learning speed was noticed. This faster convergence can be attributed to the available prior knowledge in the form of PH model.

4.3.4. EXAMPLE II: REGULATION OF A 2-DOF MANIPULATOR ARM

The EBAC Algorithm 5 is used for set-point regulation of a fully actuated 2-DOF manipulator arm. The schematic and the physical setup used in this work is shown in the

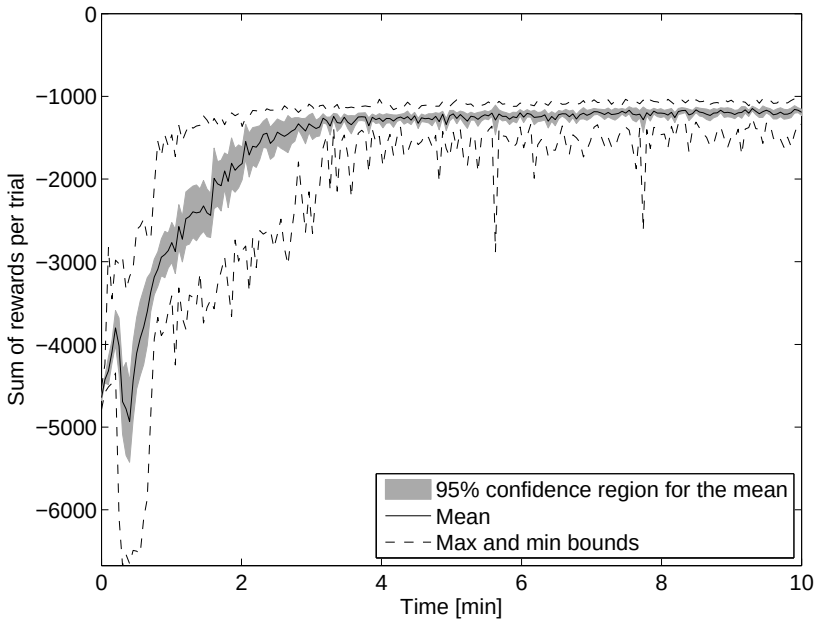


Figure 4.9: Learning curve for the proposed Energy-Balancing Actor-Critic method for 20 experiments with the real physical system.

Figure 4.10. The manipulator has 2 links, each characterized by a link length l_i, mass m_i, center of gravity r_i, and moment of inertia I_i where $i \in \{1,2\}$.

The arm can operate either in the vertical or in the horizontal plane. Here only the equations of motion for the vertical plane are given. For the horizontal plane, the potential energy terms in (2.3) are neglected. The system Hamiltonian is given by (2.3) with $q = [q_1\ q_2]^T$, and $p = [p_1\ p_2]^T = M(q)\dot{q}$, and the system's state-vector $x = [q^T\ p^T]^T$. The mass-inertia matrix $M(q)$ is

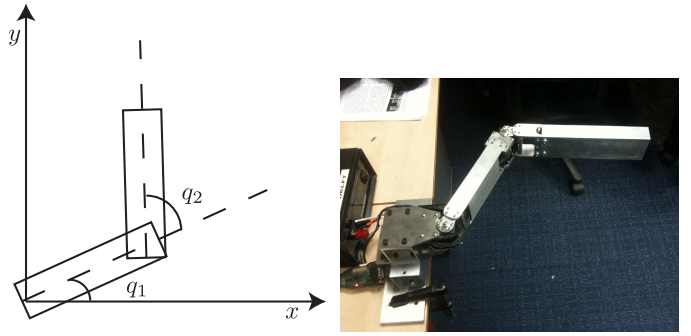
$$M(q) = \begin{bmatrix} C_1 + C_2 + 2C_3\cos(q_2) & C_2 + C_3\cos(q_2) \\ C_2 + C_3\cos(q_2) & C_2 \end{bmatrix}, \tag{4.48}$$

and the potential energy $V(q)$ is

$$V(q) = C_4\sin(q_1) + C_5\sin(q_1 + q_2). \tag{4.49}$$

The constants $C_1, \ldots, C_5$ are defined as

$$\begin{aligned} C_1 &= m_1 r_1^2 + m_2 l_1^2 + I_1, \\ C_2 &= m_2 r_2^2 + I_2, \\ C_3 &= m_2 l_1 r_2, \\ C_4 &= m_1 g(r_1 + l_1), \\ C_5 &= m_2 g r_2. \end{aligned}$$



(a) (b)

Figure 4.10: A two degree of freedom manipulator arm.

The equations of motion for the manipulator arm in PH form (2.1) are

$$\begin{aligned} \begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix} &= \left(\begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ 0 & R_{22} \end{bmatrix} \right) \begin{bmatrix} \nabla_q H(x) \\ \nabla_p H(x) \end{bmatrix} + \begin{bmatrix} 0 \\ g_{21} \end{bmatrix} u, \\ y &= \begin{bmatrix} 0 & g_{21}^T \end{bmatrix} \begin{bmatrix} \nabla_q H(x) \\ \nabla_p H(x) \end{bmatrix}, \end{aligned} \tag{4.50}$$

where R_{22} is the dissipation matrix in terms of the friction component μ

$$R_{22} = \begin{bmatrix} \mu & 0 \\ 0 & \mu \end{bmatrix},$$

and g_{21} is the input matrix in terms of gear ratio g_r and scaling factor b

$$g_{21} = \begin{bmatrix} g_r b & 0 \\ 0 & g_r b \end{bmatrix}.$$

The identified system parameters of (4.50) are given in Table 4.3.

The shaped energy term (4.32) is parameterized using 2^{nd} order Fourier basis functions (4.13) resulting in 9 learnable parameters $\xi_1, \ldots, \xi_9$. Along similar lines, the damping injection term in (4.34) and the value function (4.6) are approximated using 2^{nd} order Fourier basis function resulting in 243 and 81 learnable parameters, for the actor and critic, respectively. The choice for the Fourier basis function is influenced by the symmetry in the state and action space, since it can result in a smaller number of basis functions as compared to the radial basis function (RBF) or polynomial basis function. This reduces the number of parameters to be learnt hence resulting in a relatively faster convergence of the learning algorithm. In order to guarantee equal weight to all the system states, they are scaled to an uniform range of $\bar{x} \in (-1, 1)$. In addition, $\bar{x}$ is mapped to zero at $x = x_d$. This is to explicitly satisfy the equilibrium requirement of equation (2.34).

For the 2-DOF manipulator system (4.50), using energy-balance (4.32) and damping

Table 4.3: Manipulator arm parameters

Model parameters	Link	Symbol	Value	Units
Length of link	1	l_1	18.5×10^{-2}	m
Center of mass of link	1	r_1	11.05×10^{-2}	m
	2	r_2	12.3×10^{-2}	m
Mass of link	1	m_1	0.5	kg
	2	m_2	0.5	kg
Moment of inertia of link	1	I_1	5×10^{-3}	kg m^2
	2	I_2	5×10^{-3}	kg m^2
Gear ratio		g_r	193	
Scaling factor		b	3.74×10^{-2}	
Friction coefficient		μ	0.8738	

injection term (4.34) the control action (4.33) is

$$\begin{bmatrix} u_1(x,\xi,\psi) \\ u_2(x,\xi,\psi) \end{bmatrix} = \begin{bmatrix} \frac{-1}{g_r b}\left(\xi^T \nabla_{q_1}\phi_{es}(q) - \nabla_{q_1} H_s(q)\right) \\ \frac{-1}{g_r b}\left(\xi^T \nabla_{q_2}\phi_{es}(q) - \nabla_{q_2} H_s(q)\right) \end{bmatrix} - \begin{bmatrix} \psi_{11}\phi_{di}(x) & \psi_{12}\phi_{di}(x) \\ \psi_{21}\phi_{di}(x) & \psi_{22}\phi_{di}(x) \end{bmatrix} \begin{bmatrix} \nabla_{p_1} H_{ns}(x) \\ \nabla_{p_2} H_{ns}(x) \end{bmatrix}, \tag{4.51}$$

with $\psi_{12} = \psi_{21}$, where $H_{ns} = \frac{1}{2} p^T M^{-1}(q) p$ and $H_s = V(q)$.

A state feedback control law that stabilizes the manipulator arm at any desired state $x_d = \begin{bmatrix} q_1^* & q_2^* & 0 & 0 \end{bmatrix}^T$ is learned by using the EBAC Algorithm 5. The reward function ρ for the algorithm is formulated such that, at the desired state x_d the reward is maximum, and everywhere else a penalty is incurred

$$\rho(q,p) = \begin{bmatrix} Q_r & Q_r \end{bmatrix} \begin{bmatrix} \cos(q_1 - q_1^*) - 1 \\ \cos(q_2 - q_2^*) - 1 \end{bmatrix} - \begin{bmatrix} p_1 & p_2 \end{bmatrix} \begin{bmatrix} P_r & 0 \\ 0 & P_r \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix}, \tag{4.52}$$

where $Q_r = 25$ and $P_r = 10^4$ are the weighting constants. The cosine function is used in (4.52) to take into account the fact that the angle measurements wrap around 2π, i.e., the use of cosine function is consistent with the mapping $S^1 \to \mathbb{R}$ for the angle. Additionally, this proved to improve performance over a purely quadratic reward, such as the one used in e.g. [91]. When the system is at the desired equilibrium $x_d = (q_1^*, q_2^*, 0, 0)$ there is no penalty. Elsewhere there is a negative reward proportional to the error between the system state and the desired state x_d.

The simulation parameters and system bounds for EBAC Algorithm 5 are given in Table 4.4 and Table 4.5, respectively.

Using simulation the parameters (Table 4.4), the reward function (4.52) the EBAC Algorithm 5 is used to learn the unknown parameters of the control law (4.51). The simulation was repeated for 100 trials each of 3 seconds. This procedure is repeated 50 times

Table 4.4: Learning parameters

Parameters	**Symbol**	**Value**	**Units**
Trials	–	100	-
Trial duration	T_t	3	s
Sample time	T_s	0.01	s
Decay rate	γ	0.97	-
Eligibility trace	λ	0.65	-
Exploration variance	σ^2	1/3	-
Learning rate for critic	α_c	0.01	-
Learning rate for $\hat{V}_\mathrm{d}(q,\xi)$	$\alpha_{\mathrm{a},\xi}$	1×10^{-2}	-
Learning rate for $\hat{K}_\mathrm{d}(x,\psi)$	$\alpha_{\mathrm{a},\psi}$	1×10^{-8}	-

Table 4.5: Bounds on system states and input

Bounds	**Symbol**	**Value**	**Units**
Max control input	u_max	1	-
Min control input	u_min	-1	-
Maximum angle	q_max	$5\pi/12$	rad
Minimum angle	q_min	$-5\pi/12$	rad
Maximum momentum	p_max	$2\pi\times10^{-2}$	kg rad/sec
Minimum momentum	p_min	$-2\pi\times10^{-2}$	kg rad/sec

and the resulting mean, minimum, maximum, and confidence region of the learning curve are plotted in Figure 4.11.

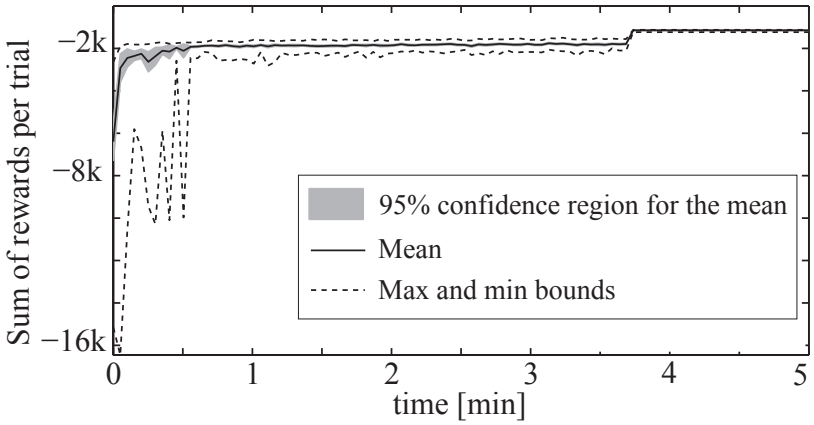


Figure 4.11: Results for the EBAC method for 50 learning simulations (k denotes 10^3).

As evident from the average learning curve in Figure 4.11, the algorithm shows good convergence as it takes around 10 trials (i.e. 30 sec) to obtain a near optimal policy. During the initial stage a dip in the learning curve is visible due to zero-initialization of the unknown parameters υ, ξ, and ψ, which is too optimistic compared to the learned solution. Once the algorithm has converged with sufficient accuracy the exploration is reduced by setting the variance of Δu_k to 0.05, resulting in a considerable jump in the learning curve at 3 minutes and 45 seconds into simulation.

By following the similar procedure the parameters of the EBAC control law (4.51) was learned experimentally on a physical setup. The evaluation of the learned controllers is depicted in Figures 4.12 and 4.13. The two learned control laws, one in simulation and the other on the physical setup, provide a comparable performance.

A major drawback of the standard PBC is the model and parameter dependency, i.e., for model and parameter uncertainties the resulting ES-DI controller might not be able to achieve zero steady-state errors [67]. This issue can be straightforwardly handled by the EBAC Algorithm 5 due to its learning capabilities. This is evaluated by intentionally considering an incorrect system Hamiltonian, neglecting the potential energy term $V(q)$ in (2.3). The physical equivalence is that the arm operates in the vertical plane while to design the control law, the horizontal plane of operation was assumed. Figure 4.14 illustrates the comparison between standard PBC and EBAC for an imprecise system Hamiltonian, whereas the standard PBC results in a small steady state error. The EBAC Algorithm 5 successfully compensates for modeling errors. The EBAC algorithm was evaluated for parameter uncertainties. This is done by using incorrect mass and length values in the control law (4.51). The EBAC algorithm was able to compensate and learn an optimal control law with zero steady-state-errors. However, the learning algorithm was found to be sensitive for the variations in the friction coefficient.

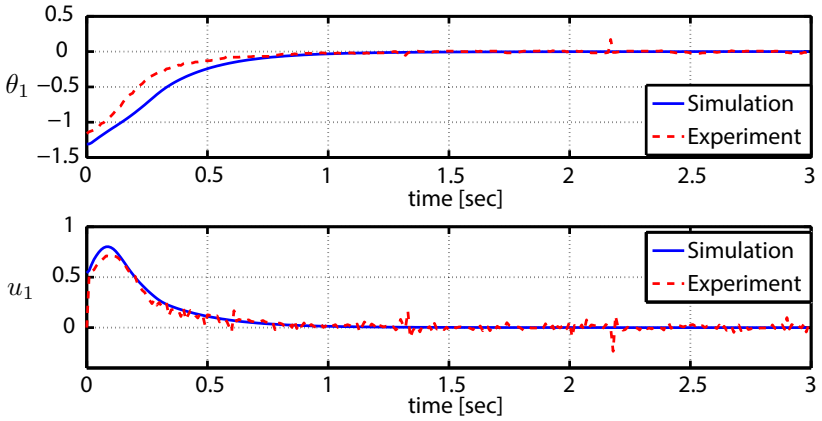


Figure 4.12: Comparison: Simulation and experimental results θ_1 and u_1.

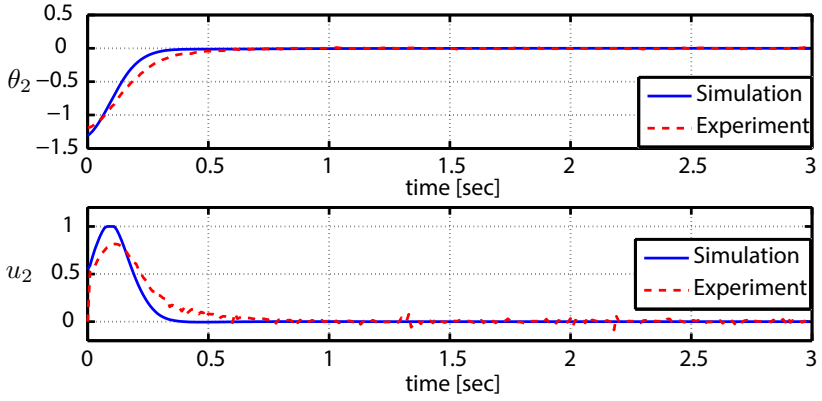


Figure 4.13: Comparison: Simulation and experimental results θ_2 and u_2.

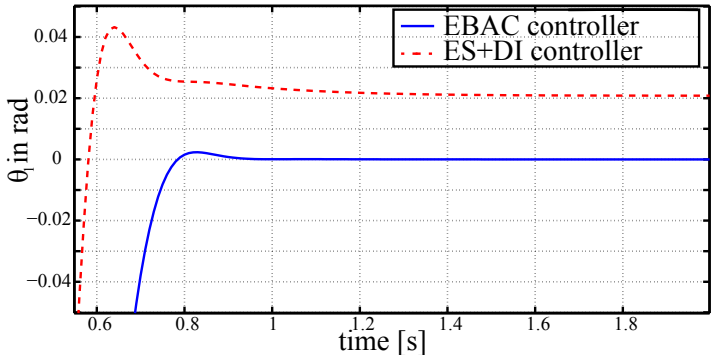


Figure 4.14: Comparison: Standard PBC (ES-DI) and Learning control via EBAC.

4.4. ACTOR-CRITIC ALGORITHMS FOR IDA-PBC

Due to the dissipation obstacle (2.42) the standard passivity-based method and hence energy-balance actor-critic (EBAC) can only be applied to a limited set of physical systems. This problem can be addressed by using the interconnection and damping assignment PBC (IDA-PBC). Also unlike standard-PBC, in IDA-PBC the original system does not need to be in the port-Hamiltonian form (2.1), any generic input-affine nonlinear form (2.13) would suffice. In this section a modified version of the online standard actor-critic learning Algorithm 4 is developed to solve the algebraic IDA-PBC and non-parametric IDA-PBC problems. The combination of these two method can be used to obtain the parameterized IDA-PBC controller, which is not detailed in this thesis.

4.4.1. ALGEBRAIC IDA-PBC

For a generic input-affine nonlinear system (2.13), the algebraic IDA-PBC objective is to find a state-feedback law $u = \beta(x)$ such that the resulting closed-loop is of the form [38],

$$\dot{x} = F_{\mathrm{d}}(x)\nabla_x H_{\mathrm{d}}(x), \tag{4.53}$$

where ∇ denotes the gradient operator $\frac{\partial}{\partial x}$. In algebraic IDA-PBC the desired Hamiltonian needs to satisfy the minimality condition (2.34) and the system matrix $F_{\mathrm{d}}(x)$ is obtained by the desired interconnection and dissipation matrix as

$$F_{\mathrm{d}}(x) = J_{\mathrm{d}}(x) - R_{\mathrm{d}}(x). \tag{4.54}$$

Using the Moore-Penrose inverse of the input matrix $g(x)$, the control law $\beta(x)$ that achieves the desired closed-loop (4.53) is

$$\beta(x) = \big(g^T(x)g(x)\big)^{-1} g^T(x)\Big(F_{\mathrm{d}}(x)\nabla_x H_{\mathrm{d}}(x) - f(x)\Big). \tag{4.55}$$

The control law in (2.47) and (4.55) are equivalent, where the PH system dynamics $\big(J(x) - R(x)\big)\frac{\partial H}{\partial x}$ in (2.47) is replaced by a more generic nonlinear system $f(x)$ in (4.55). Some of the unknown elements of $F_{\mathrm{d}}(x)$ and $H_{\mathrm{d}}(x)$ can be obtained by using the matching condition

$$g^{\perp}(x)\big(F_{\mathrm{d}}(x)\nabla_x H_{\mathrm{d}}(x) - f(x)\big) = 0, \tag{4.56}$$

where $g^{\perp}(x) \in \mathbb{R}^{(n-m)\times n}$ is the full-rank left annihilator matrix of $g(x)$, i.e., $g^{\perp}(x)g(x) = 0$. The control law (4.55) is a generalized version of (2.47) for the nonlinear system (2.13). The same implies for the matching condition (4.56) and (2.48).

In this section the algebraic IDA-PBC method is explained by using a fully actuated mechanical system as an example. Some of the difficulties encountered in using algebraic IDA-PBC will also be highlighted. Consider a fully actuated mechanical system

$$\begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial H}{\partial q}(x) \\ \frac{\partial H}{\partial p}(x) \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} u, \tag{4.57}$$

where the state vector $x = [q^T\ p^T]^T$ consists of the generalized position $q \in \mathbb{R}^{\bar{n}}$ and generalized momentum $p \in \mathbb{R}^{\bar{n}}$, with $2\bar{n} = n$. The total energy or the system Hamiltonian $H(x)$ is given by the sum of the kinetic and potential energy (2.3).

In algebraic IDA-PBC, one can choose the desired closed-loop Hamiltonian to be quadratic in increments. The minimality condition (2.34) at $x_* = [q_*^T \; 0]^T$ can be ensured by choosing $H_\mathrm{d}(x)$ as

$$H_\mathrm{d}(x) = \frac{1}{2} p^T M^{-1}(q) p + \frac{1}{2}(q - q_*)^T \Lambda (q - q_*) \tag{4.58}$$

where $\Lambda \in \mathbb{R}^{\bar{n} \times \bar{n}}$ is a positive-definite scaling matrix.

For a generic system matrix $F_\mathrm{d}(x)$

$$F_\mathrm{d}(x) = \begin{bmatrix} F_{11}(x) & F_{12}(x) \\ F_{21}(x) & F_{22}(x) \end{bmatrix} \tag{4.59}$$

by using (4.57)–(4.59) in (4.56) results in the algebraic equation

$$F_{11}(x)\Lambda(q - q_*) + F_{12}(x) M^{-1}(q) p - M^{-1}(q) p = 0, \tag{4.60}$$

which can be trivially solved by choosing $F_{11}(x) = 0$ and $F_{12}(x) = I$. Similarly by substituting (4.57)–(4.59) in (4.55) the control law will be

$$u = \beta(x) = F_{21}(x)\Lambda(q - q_*) + F_{22}(x) M^{-1}(q) p + \frac{\partial H}{\partial q}, \tag{4.61}$$

where the unknown entries F_{21} and F_{22} need to chosen appropriately. For simple control problems, like stabilization of the mass-spring-damper, the choice of F_{21} and F_{22} is straightforward [36]. However, for more challenging control tasks such as the pendulum swing-up and stabilization, finding these parameters can be difficult [85].

In this work, rather than choosing the unknown elements F_{21} and F_{22}, they are parameterized by using linear-in-parameters function approximators

$$\beta(x) = \xi_1^T \phi(x) \Lambda(q - q_*) + \xi_2^T \phi(x) M^{-1}(q) p + \frac{\partial H}{\partial q}, \tag{4.62}$$

where $\xi = [\xi_1^T \; \xi_2^T]^T$ is the unknown parameter vector and $\phi(x)$ is a user-defined matrix of basis functions.

AIDA-AC ALGORITHM

The algebraic interconnection and damping assignment actor-critic algorithm (AIDA-AC) is constructed as follows. Consider the generic algebraic IDA control law in (4.55), parameterize the matrix F_d as $F_\mathrm{d}(x, \xi)$ to obtain

$$\hat{\pi}(x, \xi) = \left(g^T(x) g(x)\right)^{-1} g^T(x) \Big(\underbrace{\xi^T \phi(x)}_{F_\mathrm{d}(x,\xi)} \nabla_x H_\mathrm{d}(x) - f(x) \Big), \tag{4.63}$$

where ξ is the unknown parameter matrix. These parameters are updated by using the standard actor-critic Algorithm 4. A block diagram representation of the algebraic IDA learning algorithm is given in Figure 4.15. The following two examples demonstrates the feasibility of the algorithm, first it is evaluated for swing-up and stabilization of the pendulum at the upright position. In the second example the learning algorithm is used to solve the regulation control of a magnetic levitation system.

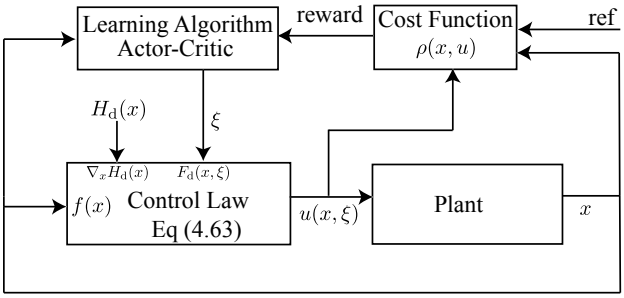


Figure 4.15: Block diagram representation of AC algorithm for Algebraic IDA-PBC.

4

EXAMPLE I: PENDULUM SWING-UP AND STABILIZATION

The AIDA-AC algorithm is evaluated for the pendulum swing-up and stabilization task. As explained in [85], devising a single smooth energy-based control-law that can achieve both swing-up and stabilization of a pendulum is an arduous task. Here, by using the learning method the swing-up and stabilization of a pendulum can be achieved with relatively low controller complexity.

For the pendulum dynamics (4.17), the system parameters of the laboratory setup given in Figure 4.2 are in Table 4.1. The desired Hamiltonian is chosen to be quadratic in increments as

$$H_{\mathrm{d}}(x) = \frac{1}{2}\gamma_{\mathrm{q}}(q-q_*)^2 + \frac{p^2}{2J_{\mathrm{p}}}, \tag{4.64}$$

where $x = [q\ p]^T$, γ_{q} is a unit conversion factor. It is evident that the desired Hamiltonian $H_{\mathrm{d}}(x)$ satisfies the minimality condition (2.34) at $x_* = (q_*, p) = (0,0)$. By using the desired closed-loop matrix (4.59), the matching condition (4.56) is

$$\frac{p}{J_{\mathrm{p}}} = F_{11}(x)\gamma_{\mathrm{q}}(q-q_*) + F_{22}(x)\frac{p}{J_{\mathrm{p}}}, \tag{4.65}$$

which can be trivially solved by choosing $F_{11}(x) = 0$ and $F_{12}(x) = 1$. For the sake of plainness we chose $F_{22}(x) = R_{22}$, where R_{22} is given in (4.18). The algebraic IDA-PBC feedback control law (4.63) for the pendulum (4.17) using (4.59) and (4.64) is

$$\begin{aligned}\beta(x) &= -F_{21}(x)\gamma_{\mathrm{q}}(q-q_*) - M_{\mathrm{p}}g l_{\mathrm{p}}\sin(q),\\ &= -\xi^T\phi(x)\gamma_{\mathrm{q}}(q-q_*) - M_{\mathrm{p}}g l_{\mathrm{p}}\sin(q).\end{aligned} \tag{4.66}$$

The unknown vector ξ is then learned using the standard actor-critic Algorithm 4. The actor and critic learning rate is given in Table 4.6. For other simulation parameters, see Table 4.2.

Table 4.6: Learning rates for the pendulum swing-up task(for rest of the parameters see Table 4.2)

Learning rate	**Symbol**	**Value [Units]**
Learning rate critic	α_{c}	0.01 [-]
Learning rate $F_{21}(x)$	$\alpha_{\mathrm{a}\xi}$	1×10^{-8} [-]

Using the quadratic reward function $\rho(x,u) = Q_r q^2 - R_r p^2$ where $Q_r = 30$, $R_r = \frac{0.1}{J_p^2}$, two controllers are learnt, one in simulation and the other on the physical system. The evaluation of the learned control laws in simulation and experiment is given in Figure 4.16. Due to the limited actuation, the pendulum first builds up the required mo-

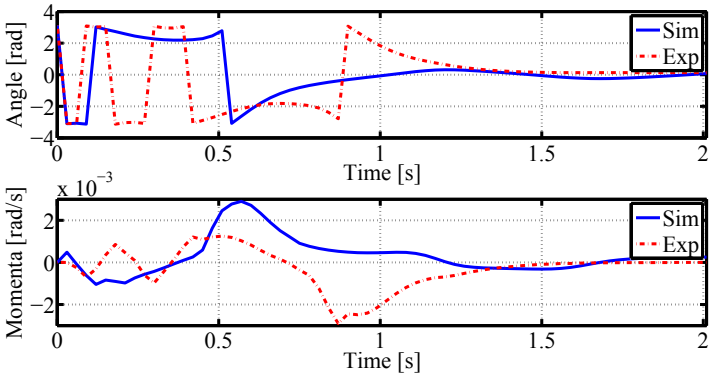


Figure 4.16: Simulation and experimental result for pendulum swing-up using AIDA-PBC.

mentum by swinging back and forth. After sufficient energy is achieved, the controller is able to swing-up and stabilize the pendulum at the desired up-right position.

EXAMPLE II: STABILIZATION OF MAGNETIC LEVITATION SYSTEM

The dynamics of the magnetic levitation system [107], illustrated in Figure 4.17, are

$$\begin{aligned} M\ddot{q} &= Mg - \frac{e^2 C_1}{2\big(C_1 + L_0(C_2+q)\big)^2}, \\ \dot{e} &= -R\frac{e(C_2+q)}{C_1 + L_0(C_2+q)} + u, \end{aligned} \tag{4.67}$$

where q is the position of the steel ball, and $e = L(q)i$ is the magnetic flux, a function of the current i through the coil and the varying-inductance $L(q)$ given by

$$L(q) = \frac{C_1}{C_2+q} + L_0. \tag{4.68}$$

The actuating signal is the voltage u across the coil. The system parameters from [107] are given in Table 4.7.

Table 4.7: System parameters for the magnetic levitation system

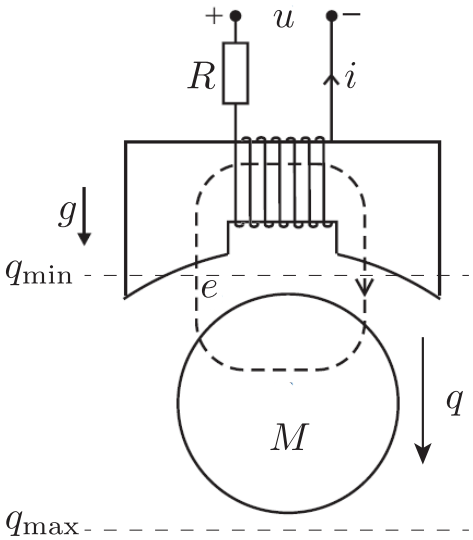


Figure 4.17: Schematic of the magnetic levitation system. Adopted from [32].

Model parameters	Symbol	Value	Units
Mass of steel ball	M	0.8	kg
Electrical resistance	R	11.68	Ω
Coil parameter 1	C_1	1.6×10^{-3}	Hm
Coil parameter 2	C_2	7×10^{-3}	m
Nominal inductance	L_0	0.8052	H
Gravity	g	9.81	m/s^2

The desired Hamiltonian is

$$H_{\mathrm{d}}(x) = \frac{1}{2}\gamma_q(q-q_*)^2 + \frac{p^2}{2M} + \frac{1}{2L_0}(e-e_*)^2, \tag{4.69}$$

where the system state is $x = [q \quad p \quad e]^T$ in terms $p = M\dot{q}$ which is the momentum and $e_* = \sqrt{2Mg/C_1}\big(C_1 + L_0(C_2 + q_*)\big)$ is the required flux at the desired position q_*. Observe that the desired Hamiltonian $H_{\mathrm{d}}(x)$ satisfies the minimality condition (2.34) at $x_* = (q_*, 0, e_*)^T$. A generic system matrix $F_{\mathrm{d}}(x)$ for the magnetic levitation system is

$$F_{\mathrm{d}}(x) = \begin{bmatrix} F_{11}(x) & F_{12}(x) & F_{13}(x) \\ F_{21}(x) & F_{22}(x) & F_{23}(x) \\ F_{31}(x) & F_{32}(x) & F_{33}(x) \end{bmatrix}. \tag{4.70}$$

By using the desired closed-loop matrix (4.70), the first matching condition (4.56) is

$$\frac{p}{M} = F_{11}(x)\gamma_{\mathrm{q}}(q-q_*) + F_{12}(x)\frac{p}{M} + F_{13}(x)\frac{e-e_*}{L_0}, \tag{4.71}$$

which can be trivially solved by choosing $F_{11}(x) = 0$, $F_{12}(x) = 1$ and $F_{13}(x) = 0$. For the sake of plainness by choosing $F_{21}(x) = 1$, $F_{31}(x) = 0$, and $F_{33}(x) = R$ where the resistance R is given in Table 4.7. The achievable algebraic IDA-PBC closed loop is

$$\begin{bmatrix} \dot{q} \\ \dot{p} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -F_{22}(x) & F_{23}(x) \\ 0 & -F_{23}(x) & -R \end{bmatrix} \begin{bmatrix} \frac{\partial H_\mathrm{d}}{\partial q}(x) \\ \frac{\partial H_\mathrm{d}}{\partial p}(x) \\ \frac{\partial H_\mathrm{d}}{\partial e}(x) \end{bmatrix}. \tag{4.72}$$

The feedback control law (4.63) for the magnetic levitation system (4.67) using (4.72) and (4.69) is

$$\begin{aligned} \beta(x) &= -F_{23}(x)\frac{p}{M} - R\frac{(e-e_*)}{L_0} + R\frac{e(C_2+q)}{\big(C_1+L_0(C_2+q)\big)} \\ &= -\xi^T\phi(x)\frac{p}{M} - R\frac{(e-e_*)}{L_0} + R\frac{e(C_2+q)}{\big(C_1+L_0(C_2+q)\big)} \end{aligned} \tag{4.73}$$

The unknown parameter vector ξ of (4.73) is learnt using the standard actor-critic Algorithm 4. It must be noted that for the choice of the desired system matrix (4.70) and desired Hamiltonian (4.69), the second matching condition

$$Mg - \frac{e^2 C_1}{2\big(C_1+L_0(C_2+q)\big)^2} = F_{22}(x)\frac{p}{M} + F_{23}(x)\frac{e-e_*}{L_0}, \tag{4.74}$$

was not explicitly solved. This is to ensure a higher freedom in learning at the cost of guaranteed closed-loop passivity. For the learning algorithm the required simulation parameters are given in Table 4.8.

Table 4.8: Learning parameters for magnetic levitation system

Parameter	**Symbol**	**Value**	**Units**
Trials	–	100	-
Time per trial	T_t	2	s
Sample time	T_s	0.004	s
Decay rate	γ	0.95	-
Eligibility trace	λ	0.65	-
Learning rate critic	α_c	0.01	-
Learning rate $F_{23}(x)$	$\alpha_{\mathrm{a}\xi}$	1×10^{-7}	-

Due to physical constraints, the control input and the system states are bounded, their respective ranges are given in Table 4.9. A stabilizing control law without pre-magnetization was learned in simulation. The resulting learning curve and a sample simulation of the learnt control law are illustrated in Figure 4.18 and Figure 4.19, respectively. For the magnetic levitation system (4.67), the algebraic-IDA control law (4.73) was learned 20 times the resulting the average learning curve is given in Figure 4.20.

Table 4.9: Bounds on system states and input for the magnetic levitation system

System state	Symbol	Value	Units
Input voltage	$u_{\max}$	60	V
	$u_{\min}$	-60	V
Position	$q_{\max}$	13	mm
	$q_{\min}$	0	mm
Momentum	$p_{\max}$	3×10^{-1}	kg m/s
	$p_{\min}$	-3×10^{-1}	kg m/s
Magnetic flux	$e_{\max}$	3	Wb
	$e_{\min}$	-3	Wb

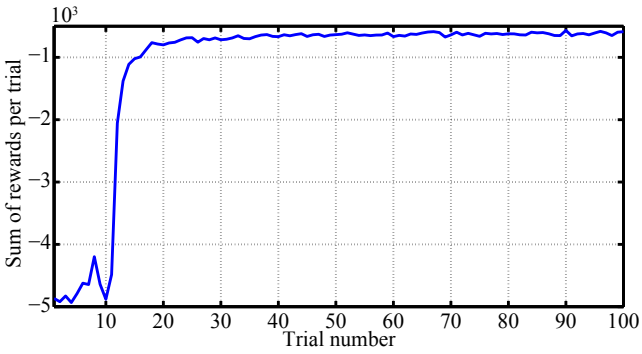


Figure 4.18: Magnetic levitation learning curve for algebraic IDA-PBC.

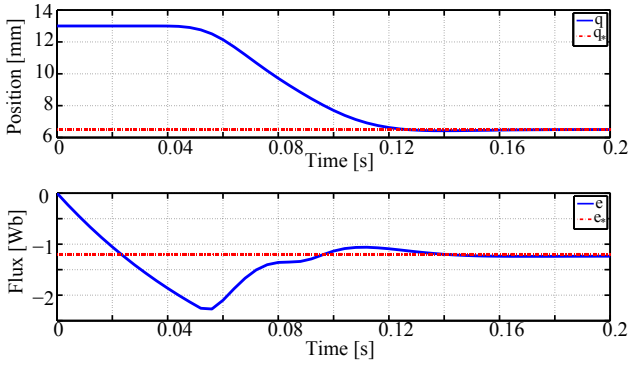


Figure 4.19: Evaluation of learned control law for magnetic levitation using AIDA-AC.

Although there is an input, the steel ball stays in the rest position (i.e. $13mm$) till 0.05 seconds, this is due to the time required to magnetize the coil.

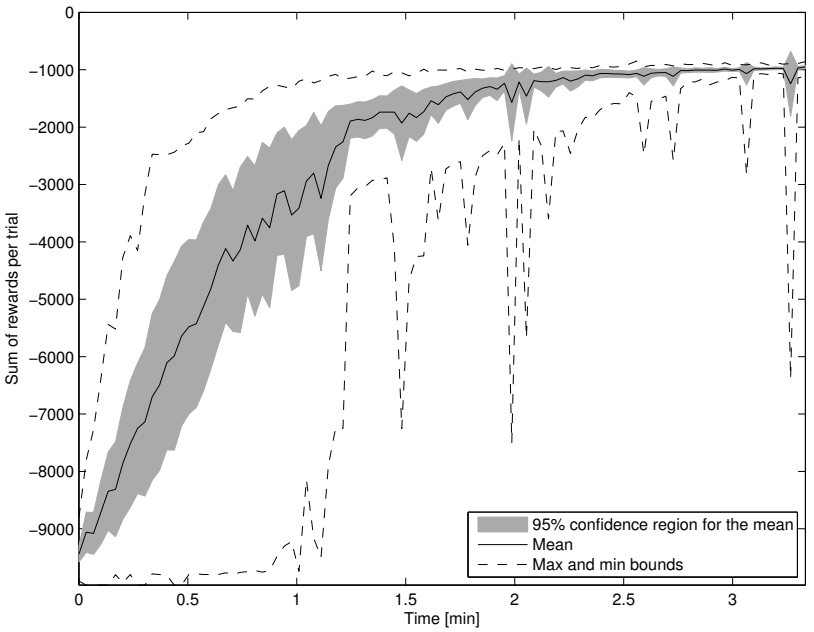


Figure 4.20: Learning curve for magnetic levitation system using algebraic-AC for 20 simulations.

4.4.2. NON-PARAMETERIZED IDA-PBC

In this section the design issues associated with non-parameterized IDA-PBC control design are explained. The control synthesis procedure is simplified by using function approximators in terms of unknown parameter vectors. These parameters are then learned using the actor-critic approach. This is evaluated for the stabilization of a generic input-affine nonlinear system. The combination of this method with the AIDA-AC approach introduced in the previous section can be used to address the parameterized IDA-PBC synthesis problem.

CONTROL PARAMETRIZATION

As explained in Section 4 of Chapter 2, for the non-parameterized IDA-PBC the desired system matrix $F_{\mathrm{d}}(x)$ defined in (4.54) is fixed. The design objective is to find an appropriate gradient of the desired Hamiltonian, i.e., $\frac{\partial H_{\mathrm{d}}}{\partial x}(x)$ such that the matching condition (4.56) is satisfied and the system (2.13) is stabilized at the desired equilibrium x_*. However as explained in [28] finding the correct gradient $\frac{\partial H_{\mathrm{d}}}{\partial x}(x)$ can be cumbersome. Instead the gradient can be replaced as $\frac{\partial H_{\mathrm{d}}}{\partial x}(x) = [h_1(x) \quad h_2(x) \quad \cdots \quad h_n(x)]^T$, where $h_1(x), h_2(x), \cdots, h_n(x)$ etc. are unknown functions [28]. By using a prefixed system matrix $F_{\mathrm{d}}(x)$, the IDA-PBC control law (4.55) can be obtained in terms of $(n-m)$ gradient components, $(h_1(x), h_2(x), \cdots, h_{n-m}(x))$. This can be explained with a simple example.

Consider a fully actuated mechanical system

$$\begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial H}{\partial q}(x) \\ \frac{\partial H}{\partial p}(x) \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} u, \tag{4.75}$$

where the generalized position $q \in \mathbb{R}^{\bar{n}}$ and the momentum $p \in \mathbb{R}^{\bar{n}}$ constitutes the state vector $x = [q^T \; p^T]^T$ with $2\bar{n} = n$. The total energy or the system Hamiltonian $H(x)$ is given by the sum of the kinetic and potential energy

$$H(x) = \frac{1}{2} p^T M^{-1}(q) p + V(q), \tag{4.76}$$

where the mass-inertia matrix $M(q) \in \mathbb{R}^{\bar{n} \times \bar{n}}$ is positive-definite. The potential energy term $V(q) \in \mathbb{R}$ is assumed to be bounded from below. By fixing the desired system matrix $F_{\mathrm{d}}(x)$ as

$$F_{\mathrm{d}}(x) = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}, \tag{4.77}$$

the matching condition (4.56) for (4.75) using (4.77) and the gradient $\nabla_x H_{\mathrm{d}}(x) = [h_1(x) \; h_2(x)]^T$ is

$$-h_1(x) + h_2(x) = M^{-1}(q) p, \tag{4.78}$$

the component $h_1(x)$ can be parameterized by using the function approximation in terms of an unknown parameter vector $\xi \in \mathbb{R}^{n_{\mathrm{c}}}$ and a known basis function vector $\phi_{\mathrm{c}}(x) \in \mathbb{R}^{n_{\mathrm{c}}}$ i.e., $h_1(x) = \xi^T \phi_{\mathrm{c}}(x)$. Then the non-parameterized IDA-PBC control law for the system (4.75) is

$$u = \beta(x) = -\xi^T \phi_{\mathrm{c}}(x) - M^{-1}(q) p + \frac{\partial H}{\partial q}, \tag{4.79}$$

The minimality condition (2.34) at $x_* = [q_*^T \; 0]^T$ can be ensured by having an appropriate value for the basis function $\phi_{\mathrm{c}}(x_{\mathrm{d}})$. The unknown parameter vector ξ can be learned online by using reinforcement learning as explained in the following section.

NON-PARAMETERIZED IDA ACTOR-CRITIC

The non-parameterized IDA algorithm is constructed similar to algebraic IDA. First consider the generic IDA control law in (4.55) for an affine nonlinear system (2.13), parameterize the Hamiltonian derivative as $\nabla_x H_{\mathrm{d}}(x, \xi) = \xi^T \phi_{\mathrm{c}}(x) = [h_1(x) \quad h_2(x) \quad \cdots \quad h_{n-m}(x)]^T$, where $(h_1(x), h_2(x), \cdots, h_{n-m}(x))$ are free components and they satisfy the matching condition (4.56). The control law is

$$\hat{\pi}(x, \xi) = \left(g^T(x) g(x)\right)^{-1} g^T(x) \Big(F_{\mathrm{d}} \underbrace{\xi^T \phi_{\mathrm{c}}(x)}_{\nabla_x H_{\mathrm{d}}(x, \xi)} - f(x) \Big), \tag{4.80}$$

where ξ is the unknown parameter matrix. These parameters are updated by using the standard actor-critic Algorithm 4. A block diagram representation of the non-parametric IDA-PBC learning algorithm is given in Figure 4.21. The following example demonstrates the feasibility of the algorithm, it is evaluated to show the stabilization of an affine nonlinear system at the origin.

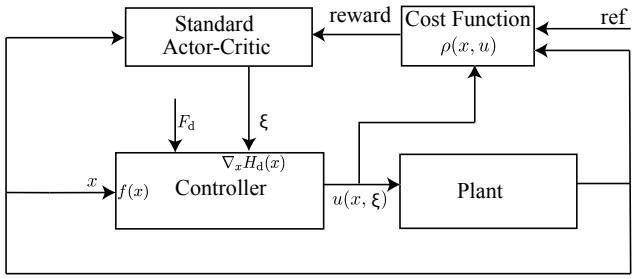


Figure 4.21: Block diagram representation of AC algorithm for non-parameterized IDA-PBC.

4

EXAMPLE: NONLINEAR SYSTEM

For the nonlinear system

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_1^2 + x_2 \\ x_1 + 2x_2 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \tag{4.81}$$

non parametric actor-critic algorithm is used to stabilize the system at the origin from a given initial state. The system matrix $F_\mathrm{d}(x)$ is fixed as

$$F_\mathrm{d}(x) = \begin{bmatrix} -1 & 1 \\ -1 & -1 \end{bmatrix}. \tag{4.82}$$

Using (4.81), (4.82) with $\frac{\partial H_\mathrm{d}}{\partial x} = [h_1(x) \quad h_2(x)]^T$ and $g^\perp = [1 \quad 0]$ in the matching condition (4.56) results in

$$-h_1(x) + h_2(x) = x_1^2 + x_2. \tag{4.83}$$

The resulting non-parameterized IDA-PBC control law (4.55) for $h_2(x) = h_1(x) + x_1^2 + x_2$ is

$$\beta(x) = -2h_1(x) - x_1^2 - x_1 - 3x_2, \tag{4.84}$$

where $h_1(x)$ is yet to be designed. In this work this function is parameterized using a function approximator as $h_1(x) = \xi^T \phi_\mathrm{c}(x)$. The unknown vector ξ is then learned using the standard actor-critic Algorithm 4. The actor and the critic learning rate is given in Table 4.10. For other simulation parameters, see Table 4.2. The control input and the system states are bounded and their respective ranges are given in Table 4.11.

Table 4.10: Learning rates for the pendulum swing-up task (for rest of the parameters see Table 4.2)

Learning rate	**Symbol**	**Value [Units]**
Learning rate critic	α_c	0.01 [-]
Learning rate $F_{21}(x)$	$\alpha_{\mathrm{a}\xi}$	1×10^{-4} [-]

Using the square root reward function $\rho(x,u) = -Q_1\sqrt{\bar{x}_1} - Q_2\sqrt{\bar{x}_2}$ where $Q_1 = Q_2 = 5$, and $\bar{x} = sign(x)\frac{x}{x_\mathrm{max}}$. A stabilizing control law was learned in simulation. The resulting learning curve and a sample simulation of the learnt control law are illustrated in Figure 4.22 and Figure 4.23, respectively.

Table 4.11: Bounds on system states and input for the nonlinear system

System state	Symbol	Value
Input	$u_{\max}$	60
	$u_{\min}$	-60
State	$x_{\max}$	5
	$x_{\min}$	-5

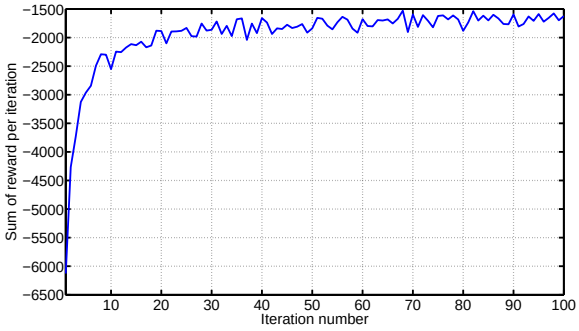


Figure 4.22: Non-parameterized IDA-PBC learning curve for nonlinear system (4.81).

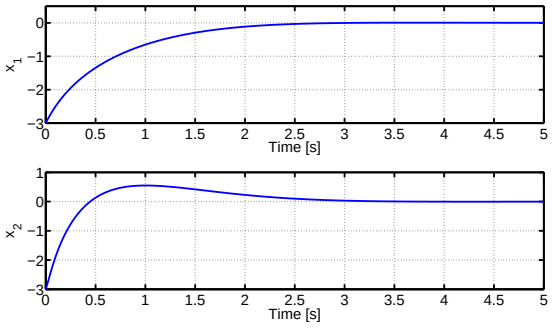


Figure 4.23: Evaluation of non-parameterized IDA-PBC learned control law for the nonlinear system (4.81).

4.5. CONTROL-BY-INTERCONNECTION USING RL

For a given port-Hamiltonian system (2.1) depending on whether the state is independent of dissipation or not, its state-space can be separated into shapable and non-shapable components (i.e., $x = [x_s^T \; x_{ns}^T]^T$) [36]. This classification is dictated by the dissipation obstacle (2.57), which requires the state-dependent function S in (2.52) to be a function of only the shapable component. Thus the static relationship between controller and the system states is $\xi = S(x_s)$. For a mechanical system (2.2), the dissipation component

of the system is embedded in the submatrix D. Since the dissipation influences only the momentum p, thus making it the non-shapable component, based on the earlier remark, the static relationship between the controller and system states will be $\xi = S(q)$.

A straightforward choice for the function S would be $S(q) = q + \kappa$ where κ is some constant that ensures equilibrium condition for $x_* = (q_*, 0)$, a common choice is $\kappa = -q_*$. The resulting controller Hamiltonian will be $H_c(\xi) = H_c(q + \kappa)$.

It must be noted that the controller Hamiltonian $H_c(q+\kappa)$ is still an unknown function and it may have infinitely many solutions. Also in CbI, it is not possible to incorporate performance measures to determine a valid choice for $H_c(q+\kappa)$. The aforementioned issues can be addressed by first parameterizing the controller Hamiltonian as a polynomial of degree n_a,

$$H_c(q+\kappa) = \frac{\theta_1}{2}(q+\kappa)^2 + \frac{\theta_2}{3}(q+\kappa)^3 + \cdots + \frac{\theta_{n_a}}{n_a+1}(q+\kappa)^{n_a+1} \tag{4.85}$$

where $\theta = (\theta_1, \cdots, \theta_{n_a})$ is an unknown parameter vector. The parameters can then be learned by using the standard actor-critic algorithm 4.

4.5.1. CBI-AC ALGORITHM

For the controller Hamiltonian (4.85), the control law for CbI is

$$\begin{aligned} u &= -y_c \\ &= -g_c^T(\xi)\frac{\partial H_c}{\partial \xi}(\xi) \\ &= -g_c^T(\xi)\left(\theta_1(q+\kappa) + \theta_2(q+\kappa)^2 + \cdots + \theta_n(q+\kappa)^n\right), \\ &= -g_c^T(\xi)\left(\theta^T\phi_a(x_s)\right), \\ &= \hat{\pi}(x_s, \theta) \end{aligned} \tag{4.86}$$

where θ is the unknown parameter matrix. These parameters can be learned using the standard actor-critic scheme Algorithm 4. A block diagram representation of the control by interconnection learning algorithm is given in Figure 4.24.

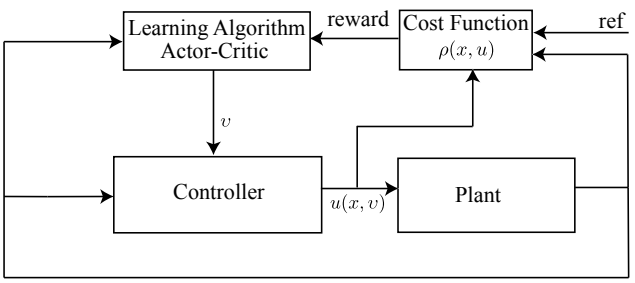
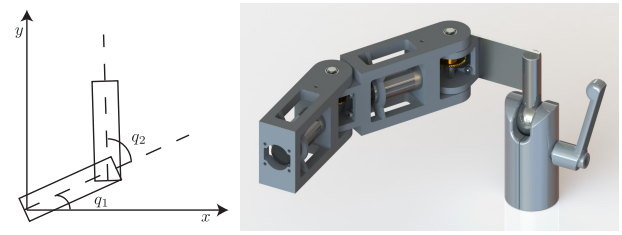


Figure 4.24: Block diagram representation of AC algorithm for Control by interconnection.



(a) (b)

Figure 4.25: A two degree of freedom manipulator arm.

The Algorithm 4 is modified to consider the saturation function which limits the control input between an upper and a lower bound. For a traditional saturation in $u \in \mathbb{R}$ of the form $\max(u_{min}, \min(u_{max}, u))$, i.e. assuming input u is bounded by u_{min} and u_{max}, then the gradient of the policy is the zero outside the unsaturated region (i.e. when $u \leq u_{min}$ or $u \geq u_{max}$). For other types of saturation the function the gradient of the policy must be computed. It must be noted that during learning the asymptotic stability of the equilibrium cannot be guaranteed, however, the closed loop remains passive. This is because the controller Hamiltonian is learned while ensuring the chain of equalities (2.54)–(2.57). The boundedness of the actor parameter (controller parameter θ in (4.85)) can be ensured by using an appropriate projection operator [30].

4.5.2. EXAMPLE: MANIPULATOR ARM

The setpoint regulation of a two degree-of-freedom manipulator arm by using control-by-interconnection is evaluated using the Algorithm 4. The schematic of the physical setup is given in Figure 4.25. The equations of motion for the manipulator arm is given in (4.50) and its parameters is in Table 4.12.

Table 4.12: Parameter values of the manipulator arm

Model parameters	**Link**	**Symbol**	**Value**	**Units**
Length of link	1	l_1	10×10^{-2}	m
	2	l_2	10×10^{-2}	m
Center of mass of link	1	r_1	5×10^{-2}	m
	2	r_2	5×10^{-2}	m
Mass of link	1	m_1	0.2	kg
	2	m_2	0.2	kg
Moment of inertia of link	1	I_1	5×10^{-4}	kgm^2
	2	I_2	5×10^{-4}	kgm^2
Gear ratio		g_r	20	
Scaling factor		b	3×10^{-1}	
Friction coefficient		μ	0.15	

The control input to the manipulator arm (4.50) using (4.85) to ensure the equilibrium condition for $x_* = (q_*, 0)$ is

$$u = \theta_1(q - q_*) + \theta_2(q - q_*)^2 + \cdots + \theta_{n_a}(q - q_*)^{n_a}. \tag{4.87}$$

The parameters of the control law (4.87) can then be learned using the Algorithm 4. The simulation and system bounds used for the evaluation are given in Table 4.13 and Table 4.14, respectively. The algorithm was implemented in Matlab, the simulation was repeated for 100 trials each of 2 seconds (i.e., 2000 samples).

Table 4.13: Learning parameters for manipulator arm

Parameter	Symbol	Value	Units
Trials	–	100	-
Trial duration	T_t	2	s
Sample time	T_s	0.001	s
Discount factor	γ	0.97	-
Eligibility trace	λ	0.67	-
Critic learning rate	α_v	0.01	-
Actor learning rate	α_a	5×10^{-7}	-

Table 4.14: Bounds on system states and input

System variable	Symbol	Value	Units
Control input	u_{max}	0.3	-
	u_{min}	-0.3	-
Position	q_{max}	1	rad
	q_{min}	-1	rad
Momentum	p_{max}	π	kg rad/s
	p_{min}	$-\pi$	kg rad/s

The algorithm was evaluated for two different variants of cost functions. First using a quadratic type of cost function the usefulness of the CbI-AC is demonstrated. For this purpose the reward function ρ is formulated as

$$\rho(x, u) = -(x - x_*)^T Q_x (x - x_*) - u^T Q_u u, \tag{4.88}$$

where Q_x penalizes the controller if the system state x is away from the desired state x_*. Similarly, Q_u will penalize the control effort if it uses too large a control action. The chosen values are

$$Q_x = \begin{bmatrix} 1 \times 10^2 & 0 & 0 & 0 \\ 0 & 1 \times 10^2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$Q_u = \begin{bmatrix} 8 \times 10^3 & 0 \\ 0 & 8 \times 10^3 \end{bmatrix}.$$

The choice for Q_u is motivated by the saturation bounds. The CbI-AC is evaluated using the parameters of Table 4.13 for the cost function (4.88). The control objective is to stabilize the arm at the desired position $x_* = (0,\ 0,\ 0,\ 0)^T$ from an initial position $x_0 = (q_{\min},\ q_{\min},\ 0,\ 0)^T$. The simulation is repeated 20 times, the resulting average, minimum, maximum and the confidence region of the learning curve are plotted in Figure 4.26.

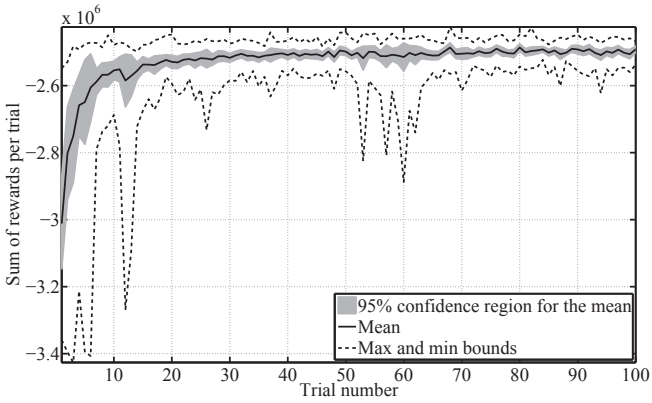


Figure 4.26: Result of CbI-AC for cost function (4.88).

The evaluation of the learned control law that stabilizes the arm at $x_* = (0,\ 0,\ 0,\ 0)^T$ from an initial position $x_0 = (q_{\min},\ q_{\min},\ 0,\ 0)^T$ is given in Figure 4.27.

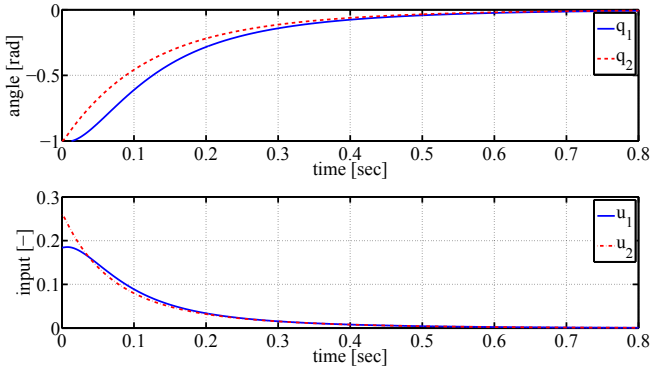


Figure 4.27: Evaluation of CbI-AC control law for linear quadratic cost function.

As it is evident from the response shown in Figure 4.27 there is no control saturation and the system input is well within the bounds. However, the response is rather slow. In order to achieve faster step response a lower Q_u can be used thus resulting in a higher control action. Also a faster response can be achieved by incorporating a time-weighted

penalty for the controller. For this purpose the reward function is made time varying

$$\rho(x,u,k) = -\eta^k (x - x_*)' Q_x (x - x_*), \tag{4.89}$$

where $\eta > 1$ is some constant and k is the time index. The longer the controller takes to stabilize the system at the desired position x_* the more negative reward it will receive. It must be noted that the reward function (4.89) depends on the time index k hence it does not strictly satisfy the Markov property. However, the system dynamics can still be assumed to be a Markov decision process as this reduces the complexity of the problem [12]. The CbI-AC Algorithm 4 is evaluated using the parameters of Table 4.13 for the cost function (4.89). The simulation is repeated 20 times, the resulting average, minimum, maximum and the confidence region of the learning curve are plotted in Figure 4.28.

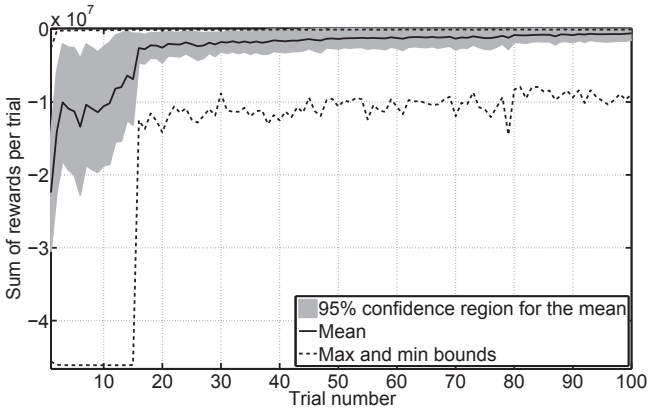


Figure 4.28: Result of CbI-AC for cost function (4.89).

The evaluation of the learned control law that stabilizes the system at the origin is given in Figure 4.29.

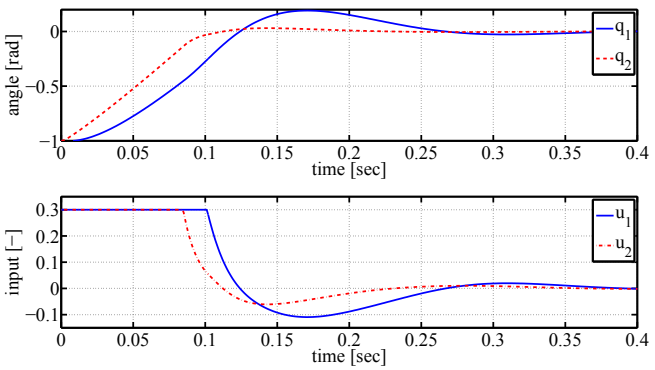


Figure 4.29: Evaluation of CbI-AC control law for time-based cost function.

4.6. DISCUSSION AND CONCLUSIONS

Based on the results and observations done in this chapter it can be summarized that the general advantages of combining learning with PBC are

- Learning based algorithms (EBAC, AIDA-AC, CbI-AC etc.) can avoid solving complex nonlinear equations. For example the PDE's resulting from the matching condition was either solved implicitly or used to formulate the control law.
- RL allows for the local specification of the stabilization or regulation objective, i.e., the minimality condition (2.34) can be easily satisfied for the desired Hamiltonian, whereas in model-based PBC, one must specify the global desired Hamiltonian.
- The use of prior information in the form PH model increases the learning speed. This has been experimentally observed for a simple pendulum stabilization, regulation of a manipulator arm etc.
- Robustness against model and parameter uncertainty can be achieved thanks to learning.
- Nonlinearities such as control saturation can be easily handled by RL.
- Physical meaning can be attributed to the learned control law.

In this chapter various learning based approaches were evaluated for different physical systems, such as pendulum, manipulator arm, magnetic levitation etc. The PBC control law was systematically parameterized using function approximators and the unknown parameters were learned either using the standard actor-critic algorithm 4 or a variation of the same. Because of the use of gradient-based parameter update rule, the actor-critic based algorithms introduced in this chapter can find a locally (near-) optimal control law. Additionally, due to its learning capabilities the introduced actor-critic based algorithms are robust against model and parameter uncertainty. However, learning for PH systems introduces a few notable challenges. For example, exploration, an integral part of actor-critic techniques (hence also the introduced methods), may not be feasible when it is too dangerous to explore the system's state-space, particularly in safety-critical applications. Similar to many RL algorithms, the developed methods are also affected by the *curse of dimensionality*.

Additionally learning itself introduces new complexities. For example, Algorithm 4 has many learning parameters that need to be tuned by the designer (e.g. $\gamma, \lambda, \alpha_a, \alpha_c$, etc.). It must be noted that, among these parameters only the learning rate for the actor and the critic needs to be carefully chosen, since the rest are nominal values and can be obtained from the literature. An improper learning rate may result in poor or no learning. To obtain an appropriate learning rates (i.e., α's) a typical approach followed in RL community is gridding. The learning rate α is gridded over a suitable range and for each combination the learning process is repeated until a satisfactory result is obtained [18]. This is a rather long and computationally expensive approach. Nevertheless, there is a great potential in integrating learning algorithms within the PH framework, as it considerably simplifies the control design process.

5

CONVERGENCE OF PASSIVITY-BASED ACTOR-CRITIC ALGORITHMS

The learning algorithms introduced in Chapter 4 follow the standard actor-critic framework of Algorithm 4. In this chapter a generic proof-of-convergence is given, which can be applied to the methods discussed in Chapter 4, this is provided Gaussian exploration is used. For the convergence proof, in this chapter the system is assumed to operate in a discrete time and discrete space setting, this is a major deviation from the methods discussed in Chapter 4. The discretization in space and time can be partly justified due to the computer control of physical systems. Since when using a digital computer discretization in both time and space is inevitable.

This chapter is formulated in a tutorial framework, to this end we compile all the relevant material from the literature, such as policy gradient (Section 2), stochastic approximation theory (Section 4) etc. We have developed a framework to convert an existing parameterized control law in to policy and we obtain the parameter update rule for this policy (Section 3). Finally the available actor-critic convergence proof [30, 108] is extended for discounted reward setting in Section 5.

5.1. INTRODUCTION

In this chapter the proof of convergence for the passivity-based learning algorithms proposed in Chapter 4 is given. As stated in Chapter 4, the objective in RL is to find a parameter vector θ of the policy $\pi(x,\theta)$ so as to maximize a user defined return function. When evident, the parameter θ will not be explicitly indicated. Algorithms in Chapter 4 use discounted reward as the return function. For the purpose of proof, in this chapter we deviate from the nomenclature that is used in the previous chapter. As will be explained later, the actor update (4.12) can be represented as a stochastic ODE. Instead of maximizing a return function as done in Chapter 4, the stability and the convergence proof

can be considerably simplified by opting for a minimization problem. The RL objective can be redefined as: find an optimal policy $\pi(x,\theta)$ so that a user defined *cost function* $J^{\pi}(x)$ is minimized. From the basics of the optimization theory the parameter update for θ to minimize the cost function is [109, 110]

$$\theta_{k+1} = \theta_k - \alpha_{\text{ak}} \nabla_{\theta} J^{\pi}(x), \tag{5.1}$$

where k is the time index and ∇ is the gradient derivative.

For the stability and convergence proof, the system is assumed to operate in a discrete time and discrete space setting. This assumption can be justified due to the computer control of physical systems. Since when using a digital computer discretization in both time and space is inevitable. This is due to the sampling and resolution, respectively. Use of discrete time framework is consistent with the algorithms presented in Chapter 4. Assuming the operating-space of a system is limited. The state space and action space can be discretized into a finite but large number of sections. Because of the discrete setting, a probability mass function can be used to depict the state transition probability. This simplifies the proof discussed in this chapter. Additionally, the discrete time and discrete space representation simplifies the notation and thus enhances readability.

The chapter is organized as follows, in Section 2, policy gradient for discounted reward is provided. The proof is a repetition of the available results in [18, 19, 30, 104]. It is detailed here for the sake of completeness. We have developed a framework to convert the parameterized control law to a probability density function (i.e., RL policy) and it is given in Section 3. For alternate representations of the control policy see [111]. In Section 4 a brief overview on the stochastic approximation algorithm is given, it is based on the available literature, for e.g., [29, 112–114]. Following this, in Section 5, an outline for the actor and critic convergence in discounted reward setting is given. This is reformulation of the convergence proof available in [30, 108], it is detailed here for the sake of completeness. In Section 6 concluding remarks are provided.

5.2. POLICY GRADIENT FOR DISCOUNTED REWARD SETTING

Actor-critic is a reinforcement learning method that can be used to solve an optimal control problem for a Markov decision process (MDP). As introduced in Chapter 4, an MDP is a tuple $\langle X, U, P, \rho \rangle$ where $X \in \mathbb{S} \subset \mathbb{R}^n$ is the state-space and $U \in \mathbb{U} \subset \mathbb{R}$ is the action-space. Here a single-input single-output system is assumed for the sake of notion. The proof of convergence, developed in this chapter can be easily extended to multi-input multi-output system. In a discrete space setting, $P : X \times U \times X \to [0,\infty)$ is the state-transition probability density function of reaching state $x_{k+1} = x'$ from the current state $x_k = x$ on applying action $u_k = u$, where k is the time index. The state transition can be represented in the compact form as $P^{u}_{xx'} = P^{u_k}_{x_k x_{k+1}}$. The reward function ρ is a measure of user defined performance criterion and it provides an instantaneous reward $r_{k+1} = \rho(x_k, u_k, x_{k+1})$.

The purpose of an actor-critic algorithm is to obtain a policy $\pi(x,u)$, which dictates the probability of choosing action u in state x. For a continuous state space and action space the policy is formulated in terms of an unknown parameter vector $\theta \in \mathbb{R}^{n_{\text{a}}}$ as

$\pi_\theta(x,u)$. Now the RL objective can be restated as, find an optimal vector θ such that a cost function $J^\pi(x)$ is minimized. However, in order to simplify the proof, in this chapter, a minimization problem is considered. In this chapter the discounted function $J^\pi(x)$ is minimized. The discounted return function (4.2) results in the cost-to-go function to be minimized as

$$J^\pi(x) = E\{\sum_{k=0}^{\infty} \gamma^k r_{k+1}\}. \tag{5.2}$$

The difference when compared to Chapter 4 is that here the cost function is used to solve a minimization problem whereas the value function was used for solving maximization problem. It must be explicitly noted that the change from reward to cost setting and changing maximization to minimization problem will result in the same policy.

In (5.2) $\gamma \in (0,1)$ is the discount factor used to ensure the boundedness of the cost-function. As the objective is to find a parameter vector θ so as to minimize the cost function (4.2), this implies the policy vector needs to be updated along the gradient of the cost function

$$\theta_{k+1} = \Gamma\Big(\theta_k - \alpha_{\text{ak}} \nabla_\theta J\Big), \tag{5.3}$$

where the operator ∇_θ is the partial derivative $\frac{\partial}{\partial \theta}$, the argument θ is neglected when it is evident and α_{ak} is a scaling factor. The function Γ projects the vector θ into an compact convex set, this is to ensure the boundedness of the iterates [30]. The gradient of the cost function is given by the policy gradient theorem (5.8) [103, 104].

5.2.1. POLICY GRADIENT FOR DISCOUNTED REWARD

Consider the Bellman equation that relates the state and the action-value functions

$$\begin{aligned} Q^\pi(x,u) &= E\{\rho(x,u) + \sum_{k=1}^{\infty} \gamma^k r_{k+1}\}, \\ &= E\{r_{k+1} + \gamma \sum_{x'} P^u_{xx'} V^\pi(x')\}, \end{aligned} \tag{5.4}$$

where $Q^\pi(x,u)$ is the action-value function or Q-value function and it gives the discounted cost for starting from an initial state x and applying an initial action u.

In reinforcement learning, the well known relationship between the state-value function and the action-value function is [12]

$$V^\pi(x) = \sum_u \pi_\theta(x,u) Q^\pi(x,u). \tag{5.5}$$

The derivative of the value function w.r.t. the policy parameter vector θ is [1],

$$\begin{aligned} \frac{\partial V^\pi}{\partial \theta}(x_0) &= \frac{\partial}{\partial \theta} \sum_{u_0} \Big(\pi_\theta(x_0,u_0) Q^\pi(x_0,u_0)\Big), \\ &= \sum_{u_0} \Big(\frac{\partial \pi_\theta}{\partial \theta}(x_0,u_0) Q^\pi(x_0,u_0) + \pi_\theta(x_0,u_0) \frac{\partial Q^\pi}{\partial \theta}(x_0,u_0)\Big), \end{aligned} \tag{5.6}$$

[1] When evident the Expectation $E\{\}$ is omitted, this is reduce clattering and to enhance readability.

by using the equality (5.4) the gradient of the value function is,

$$\frac{\partial V^{\pi}}{\partial \theta}(x_0) = \sum_{u_0}\left(\frac{\partial \pi_\theta}{\partial \theta}(x_0,u_0)Q^{\pi}(x_0,u_0) + \pi_\theta(x_0,u_0)\frac{\partial}{\partial \theta}\left[r_1 + \gamma\sum_{x_1} P^{u_0}_{x_0x_1}V^{\pi}(x_1)\right]\right),$$

Since the instantaneous reward r_k is independent of the parameter vector θ. This results in $\frac{\partial r_k}{\partial \theta} = 0$, where k is the time index.

$$\frac{\partial V^{\pi}}{\partial \theta}(x_0) = \sum_{u_0}\left(\frac{\partial \pi_\theta}{\partial \theta}(x_0,u_0)Q^{\pi}(x_0,u_0) + \pi_\theta(x_0,u_0)\sum_{x_1}\gamma P^{u_0}_{x_0x_1}\frac{\partial V^{\pi}}{\partial \theta}(x_1)\right), \tag{5.7}$$

by substituting equation (5.6) for the next state-action pair (x_1,u_1) in (5.7),

$$\begin{aligned}\frac{\partial V^{\pi}}{\partial \theta}(x_0) = \sum_{u_0}\Big(&\frac{\partial \pi_\theta}{\partial \theta}(x_0,u_0)Q^{\pi}(x_0,u_0)\\ &+ \pi_\theta(x_0,u_0)\sum_{x_1}\gamma P^{u_0}_{x_0x_1}\sum_{u_1}\Big(\frac{\partial \pi_\theta}{\partial \theta}(x_1,u_1)Q^{\pi}(x_1,u_1)\\ &+ \pi_\theta(x_1,u_1)\frac{\partial Q^{\pi}}{\partial \theta}(x_1,u_1)\Big)\Big),\end{aligned}$$

by following the similar process for $\frac{\partial Q^{\pi}}{\partial \theta}(x_1,u_1)$, the gradient of the cost function is,

$$\begin{aligned}\frac{\partial V^{\pi}}{\partial \theta}(x_0) = \sum_{u_0}\Big(&\frac{\partial \pi_\theta}{\partial \theta}(x_0,u_0)Q^{\pi}(x_0,u_0)\\ &+ \pi_\theta(x_0,u_0)\sum_{x_1}\gamma P^{u_0}_{x_0x_1}\sum_{u_1}\frac{\partial \pi_\theta}{\partial \theta}(x_1,u_1)Q^{\pi}(x_1,u_1)\\ &+ \pi_\theta(x_0,u_0)\underbrace{\sum_{x_1}\gamma P^{u_0}_{x_0x_1}\pi_\theta(x_1,u_1)\sum_{x_2}\gamma P^{u_1}_{x_1x_2}}_{\gamma^2 P^{\pi}_{x_0x_2}}\frac{\partial V^{\pi}}{\partial \theta}(x_2)\Big),\end{aligned}$$

where $P^{\pi}_{x_0x_2}$ is the probability of reaching state x_2 from an initial state x_0 by following the policy π. By repeating the same procedure for more iterations and using the condition $\sum_{u_0}\pi_\theta(x_0,u_0) = 1$ the policy gradient is simplified as

$$\frac{\partial V^{\pi}}{\partial \theta}(x_0) = \sum_{k=0}^{\infty}\gamma^k P^{\pi}_{x_0x_k}\sum_{u_k}\frac{\partial \pi_\theta}{\partial \theta}(x_k,u_k)Q^{\pi}(x_k,u_k).$$

Using the notation $d^{\pi}(x_0) = \sum_{k=0}^{\infty}\gamma^k P^{\pi}_{x_0x_k}$ the policy gradient will be

$$\frac{\partial V^{\pi}}{\partial \theta}(x_0) = \sum_{x} d^{\pi}(x_0)\sum_{u}\frac{\partial \pi_\theta}{\partial \theta}(x,u)Q^{\pi}(x,u).$$

The gradient of the cost function is

$$\begin{aligned}\frac{\partial J^{\pi}}{\partial \theta}(x_0) &= \sum_{x} d^{\pi}(x_0)\sum_{u}\frac{\partial \pi_\theta}{\partial \theta}(x,u)Q^{\pi}(x,u),\\ &= \sum_{x} d^{\pi}(x_0)\sum_{u}\pi_\theta(x,u)\nabla_\theta \ln \pi_\theta(x,u)Q^{\pi}(x,u).\end{aligned} \tag{5.8}$$

The second equation is obtained by using the equality $\frac{\partial \pi_\theta}{\partial \theta}(x,u) = \pi_\theta(x,u)\nabla_\theta \ln \pi_\theta(x,u)$ in terms of the compatibility function $\nabla_\theta \ln \pi_\theta(x,u)$. For the policy gradient (5.8) a state dependent baseline function can be added which reduces its variance.

5.2.2. APPROXIMATE POLICY GRADIENT: INCLUSION OF BASELINE

By adding a baseline function $b(x)$ to the policy gradient (5.8)

$$\begin{aligned}
\frac{\partial J^\pi}{\partial \theta}(x) &= \sum_x d^\pi(x) \sum_u \frac{\partial \pi_\theta}{\partial \theta}(x,u)\left[Q^\pi(x,u) - b(x)\right], \\
&= \sum_x d^\pi(x) \sum_u \frac{\partial \pi_\theta}{\partial \theta}(x,u) Q^\pi(x,u) - \sum_x d^\pi(x) b(x) \frac{\partial}{\partial \theta} \sum_u \pi_\theta(x,u), \\
&= \sum_x d^\pi(x) \sum_u \frac{\partial \pi_\theta}{\partial \theta}(x,u) Q^\pi(x,u) - \sum_x d^\pi(x) b(x) \frac{\partial}{\partial \theta}(1), \\
&= \sum_x d^\pi(x) \sum_u \frac{\partial \pi_\theta}{\partial \theta}(x,u) Q^\pi(x,u), \qquad (5.9)
\end{aligned}$$

The baseline function $b(x)$ does not change the gradient of the cost function. It is solely used to reduce the variance of the gradient. The modified policy gradient is

$$\frac{\partial J^\pi}{\partial \theta}(x) = \sum_x d^\pi(x) \sum_u \frac{\partial \pi_\theta}{\partial \theta}(x,u)[Q^\pi(x,u) - b(x)], \qquad (5.10)$$

the state-value function is generally chosen as the baseline function, i.e., $b(x) = V^\pi(x)$. This is because the resulting advantage function defined as $A^\pi(x,u) = Q^\pi(x,u) - V^\pi(x)$ can be estimated by using the temporal difference, i.e., $A^\pi(x,u) = \mathbf{E}[\delta_k | x_k, u_k, \pi]$. This can be easily demonstrated using the definition of the temporal difference

$$\begin{aligned}
\mathbf{E}[\delta_k | x_k = x,\, u_k = u,\, \pi] &= \mathbf{E}[r_{k+1} + \gamma V^\pi(x_{k+1}) - V^\pi(x_k)], \\
&= \mathbf{E}[r_{k+1} + \sum_{x'} \gamma P^u_{xx'} V^\pi(x')] - \mathbf{E}[V^\pi(x)], \\
&= Q^\pi(x,u) - V^\pi(x), \\
&= A^\pi(x,u). \qquad (5.11)
\end{aligned}$$

Using this the policy gradient (5.8) becomes,

$$\begin{aligned}
\frac{\partial J^\pi}{\partial \theta}(x) &= \sum_x d^\pi(x) \sum_u \frac{\partial \pi_\theta}{\partial \theta}(x,u)\delta_k, \\
&= \sum_x d^\pi(x) \sum_u \pi_\theta(x,u)\nabla \ln \pi_\theta(x,u)\delta_k. \qquad (5.12)
\end{aligned}$$

The temporal difference for the policy gradient can be obtained by using the approximate state-value function called the critic

$$\delta_k = r_{k+1} + \gamma \hat{V}^\pi(x_{k+1}, \upsilon_{k+1}) - \hat{V}^\pi(x_k, \upsilon_k). \qquad (5.13)$$

The critic is defined as $\hat{V}^\pi(x,\upsilon) = \upsilon^T \phi_{\mathrm{v}}(x)$, where $\upsilon \in \mathbb{R}^{n_{\mathrm{v}}}$ is an unknown parameter vector and $\phi_{\mathrm{v}}(x) \in \mathbb{R}^{n_{\mathrm{v}}}$ is an user-defined vector of basis functions. The unknown critic parameters are updated using the following gradient update rule [103]

$$\upsilon_{k+1} = \upsilon_k + \alpha_{\mathrm{vk}} \delta_{k+1} \nabla_\upsilon \hat{V}(x_k, \upsilon_k), \qquad (5.14)$$

where α_{vk} is the critic update rate. The rate of parameter convergence can be increased by using the eligibility trace $e_k \in \mathbb{R}^{n_v}$, this results in the following parameter update rule

$$\begin{aligned} e_{k+1} &= \gamma\lambda e_k + \nabla_v \hat{V}(x_k, v_k), \\ v_{k+1} &= v_k + \alpha_{\text{vk}} \delta_{k+1} e_{k+1}, \end{aligned} \tag{5.15}$$

where $\lambda \in [0,1]$ is the trace decay rate.

5.3. CONTROL LAW TO POLICY

The feedback control such as (4.33), (4.55) etc. generally has a state-to-action map $\beta_\theta(x)$. The control law is parameterized in terms of some unknown parameter vector θ. A possible option to obtain an optimal θ is by using actor-critic approach. However, in order to use (5.3) the state-action map $\beta_\theta(x)$ must be converted into a policy $\pi_\theta(x,u)$. This is done by exploiting a requirement for the online RL algorithm called the exploration [12]. Here a Gaussian noise with zero mean is added to the output of the state-action map. The final control input u that is applied to the system is,

$$\begin{aligned} u &= \beta_\theta(x_k) + \mathcal{N}(0, \sigma^2), \\ &= \beta_\theta(x_k) + \Delta u, \end{aligned} \tag{5.16}$$

this can also be written as a probability density function

$$\pi_\theta(x,u) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-(u-\beta_\theta(x))^2}{2\sigma^2}\right). \tag{5.17}$$

This can be considered as the policy for the passivity based actor-critic algorithms introduced in Chapter 4. The gradient of the policy (5.17) is

$$\begin{aligned} \frac{\partial \pi_\theta}{\partial \theta} &= \frac{1}{\sqrt{2\pi}\sigma} \exp\Big(\frac{-(u-\beta_\theta(x))^2}{2\sigma^2}\Big) 2\frac{(u-\beta_\theta(x))}{2\sigma^2} \frac{\partial \beta_\theta}{\partial \theta}(x), \\ &= \frac{1}{\sigma^2} \pi_\theta(x,u)(u-\beta_\theta(x)) \frac{\partial \beta_\theta}{\partial \theta}(x), \\ &= \frac{1}{\sigma^2} \pi_\theta(x,u) \Delta u \frac{\partial \beta_\theta}{\partial \theta}(x). \end{aligned} \tag{5.18}$$

In the gradient (5.18), the scalar factor $\frac{1}{\sigma^2}$ can be neglected for the sake of plainness. By using (5.18), the policy gradient (5.12) is

$$\frac{\partial J^\pi}{\partial \theta}(x) = \sum_x d^\pi(x) \sum_u \pi_\theta(x,u) \frac{\partial \beta_\theta}{\partial \theta}(x) \delta_k \Delta u. \tag{5.19}$$

Note that even for the compatibility function the policy gradient will be same as (5.19). By using (5.19) in (5.12) the parameter update for the state to action map $\beta_\theta(x)$ is

$$\theta_{k+1} = \Gamma\left(\theta_k - \alpha_{\text{ak}} \frac{\partial \beta_\theta}{\partial \theta}(x) \delta_k \Delta u\right). \tag{5.20}$$

5.4. STOCHASTIC APPROXIMATION ALGORITHM

Prior to showing the proof-of-convergence for the actor-critic algorithms, a generic stochastic approximation framework is introduced. Consider the following assumptions are true:

Assumption 1. The policy $\pi_\theta(x,u)$ is continuously differentiable in the parameter θ.

Assumption 2. The Markov chain induced by the policy $\pi_\theta(x,u)$ is both aperiodic and irreducible.

Consider a generic stochastic approximation algorithm

$$\vartheta_{k+1} = \vartheta_k + \alpha_k \left[p(\vartheta_k) + M_{k+1} \right] \tag{5.21}$$

where $\vartheta \in \mathbb{R}^l$ is a parameter vector, α_k is a sequence of positive numbers and M_k is a sequence of uncorrelated noise with zero mean. Observe that the critic update (5.14) and the policy update (5.20) can be represented in the generic form, albeit after some minor modifications.

In [29] it is shown that the algorithm (5.21) can be approximated by an ODE,

$$\dot{\vartheta}(\tau) = p\left(\vartheta(\tau)\right), \tag{5.22}$$

if the ODE (5.22) has a globally asymptotically stable equilibrium ϑ^* then the convergence of the iterates can be shown, i.e., $\vartheta_k \to \vartheta^*$ as $k \to \infty$ provided the following assumptions are satisfied.

Assumption 3. The function $p(\cdot)$ is Lipschitz. Additionally, there exists a steady state ODE defined as

$$\dot{\vartheta} = p_\infty(\vartheta) := \lim_{\eta \to \infty} \frac{p(\eta\vartheta)}{\eta}. \tag{5.23}$$

Furthermore, the origin of the steady state ODE (5.23) is a globally asymptotically stable equilibrium.

Assumption 4. The sequence M_k is a martingale difference sequence with $\mathscr{F}_k = \sigma\left(\vartheta(i), M(i), i \le k\right)$, additionally the discretization error is upper-bounded

$$\mathbf{E}\left[\|M_{k+1}\|^2 | \mathscr{F}_k \right] \le C_0 \left(1 + \|\vartheta_k\|^2\right). \tag{5.24}$$

Assumption 5. The sequence α_k is tapering and satisfies

$$\sum_k \alpha_k = \infty, \sum_k \alpha_k^2 < \infty. \tag{5.25}$$

In [112] two major theorems are introduced, one for stability and the other for the convergence. They are repeated here for the sake of completeness, for proof see [29, 112].

Theorem 1. Stability and convergence theorem

- Stability: If the Assumptions 3-5 are satisfied, then for any initial condition $\vartheta(0) \in \mathbb{R}^l$, $\sup_k \|\vartheta_k\| < \infty$ with probability 1 (w.p.1).
- Convergence: Suppose all the listed assumptions are satisfied, additionally the ODE (5.22) has an asymptotically stable equilibrium ϑ^*. Then $\vartheta_k \to \vartheta^*$ w.p.1 as $k \to \infty$ for any initial condition $\vartheta(0) \in \mathbb{R}^l$.

5.5. PROOF OF CONVERGENCE

In this section the critic and actor convergence is shown, prior to that the following assumptions on the critic basis function vector and the learning rate are made [108].

Assumption 6. The critic basis vector $\{\phi_{\mathrm{v}}^{(i)}\}_{i=1}^{i=n_{\mathrm{v}}}$ is linearly independent. Alternatively, let Φ be the $n_{\mathrm{s}} \times n_{\mathrm{v}}$ dimensional matrix (n_{s} is the cardinality of the space $\mathbb{S}$) whose ith column is given by $\phi_{\mathrm{v}}^{(i)} = \left(\phi_{\mathrm{v}}^{(i)}(x), x \in \mathbb{S}\right)$ is full rank.

Assumption 7. The actor and critic learning rate α_{ak} and α_{ck} satisfy

$$\sum_k \alpha_{\mathrm{ak}} = \sum_k \alpha_{\mathrm{vk}} = \infty,$$
$$\sum_k \alpha_{\mathrm{ak}}^2, \sum_k \alpha_{\mathrm{vk}}^2 < \infty,$$
$$\alpha_{\mathrm{ak}} < \alpha_{\mathrm{vk}}.$$

5

5.5.1. CRITIC CONVERGENCE

The critic convergence can be shown using the stability and convergence theorem.

Theorem 2. Critic convergence: Under Assumptions 1, 2, 6, and 7, for a given policy π_θ the critic iterates (5.14) converges, i.e., $v \to v^\pi$ with probability one, where v^π is the unique solution to

$$\Phi^T D \Phi v^\pi = \Phi^T D \left(R^\pi + \gamma P \Phi v^\pi\right), \tag{5.26}$$

where D is the diagonal matrix with entries $d^\pi(x), \forall x \in \mathbb{S}$. And R^π is the reward vector and P is the transition probability matrix.

Proof: Prior to the proof of convergence of the critic some preliminary manipulation of the value function is required. Generally, the critic approximates the value function as $\hat{V}(x_k) = v^T \phi_{\mathrm{v}}(x_k)$ where k is the time index. The state-value function is approximated in terms of the unknown parameter vector $v = (v_1, v_2, \cdots, v_{n_{\mathrm{v}}})$ and the basis function vector $\phi_{\mathrm{v}}(x_k) = \left(\phi_1(x_k), \phi_2(x_k), \cdots, \phi_{n_{\mathrm{v}}}(x_k)\right)$. The bounded state-space can be discretized into a large but finite number of sections $x \in \{x^1, x^2, \cdots, x^{n_{\mathrm{s}}}\}$, where n_{s} is a large number. The expected value of the state-value function is

$$\begin{aligned}
\mathbf{E}[\hat{V}(x)] &= \mathbf{E}[v^T \phi_{\mathrm{v}}(x)] \\
&= \sum_x d(x) v^T \phi_{\mathrm{v}}(x) \\
&= d(x^1)\left(\phi_1(x^1)v_1 + \cdots + \phi_{n_{\mathrm{v}}}(x^1)v_{n_{\mathrm{v}}}\right) \\
&\quad + d(x^2)\left(\phi_1(x^2)v_1 + \cdots + \phi_{n_{\mathrm{v}}}(x^2)v_{n_{\mathrm{v}}}\right) \\
&\quad \cdots \\
&\quad + d(x^{n_{\mathrm{s}}})\left(\phi_1(x^{n_{\mathrm{s}}})v_1 + \cdots + \phi_{n_{\mathrm{v}}}(x^{n_{\mathrm{s}}})v_{n_{\mathrm{v}}}\right),
\end{aligned} \tag{5.27}$$

this can be written in compact matrix form as

$$\mathbf{E}[V(x)] = D\Phi\vartheta, \tag{5.28}$$

where D is the diagonal matrix with entries $d(x)$ for all $x \in \mathbb{S}$. Φ is the $n_s \times n_v$ dimensional matrix (n_s is the cardinality of the space $\mathbb{S}$) whose ith column is given by $\phi_v^{(i)} = \left(\phi_v^{(i)}(x), x \in \mathbb{S}\right)$, and $\vartheta = (\vartheta_1, \vartheta_2, \cdots, \vartheta_{n_v})^T$ is the parameter vector. Consider the critic update (5.14). Assuming a linearly parameterized state-value function the critic iterate is

$$\begin{aligned} v_{k+1} &= v_k + \alpha_{vk}\delta_k \phi_v(x_k) \\ &= v_k + \alpha_{vk}\left(\underbrace{\mathbf{E}\left[\delta_k \phi_v(x_k)|\mathscr{F}_k\right]}_{p(v)} + \underbrace{\delta_k \phi_v(x_k) - \mathbf{E}\left[\delta_k \phi_v(x_k)|\mathscr{F}_k\right]}_{M_{k+1}}\right) \\ &= v_k + \alpha_{vk}\left(p(v) + M_{k+1}\right) \end{aligned} \tag{5.29}$$

is of the form (5.22), with M_{k+1} the Martingale difference sequence. The stochastic critic ODE $\dot{v} = p(v)$ can be written in a compact form as

$$\begin{aligned} \dot{v} &= \mathbf{E}\left[\delta_k \phi_v(x_k)\right] \\ &= \sum_x d(x) \sum_u \pi_\theta(x,u)\Big(r_{k+1} + \gamma v^T \sum_{x'} P(x,u,x')\phi_v(x') - v^T \phi_v(x)\Big)\phi_v(x) \\ &= \Phi^T D\left(R^\pi + \gamma P \Phi v - \Phi v\right) \end{aligned} \tag{5.30}$$

where $R^\pi = \left(\sum_u \pi_\theta(x^1,u)\rho(x^1,u), \sum_u \pi_\theta(x^2,u)\rho(x^2,u) \cdots \sum_u \pi_\theta(x^{n_s},u)\rho(x^{n_s},u)\right)^T$ and P is the state-to-state transition probability matrix defined as

$$P = \begin{pmatrix} p(x^1,x^1) & \cdots & p(x^1,x^{n_s}) \\ \vdots & \ddots & \vdots \\ p(x^{n_s},x^1) & \cdots & p(x^{n_s},x^{n_s}) \end{pmatrix}, \tag{5.31}$$

where $p(x^i,x^j) = \sum_u \pi_\theta(x^i,u)P(x^i,u,x^j)$, i and $j \in (1,n_s)$.

We show using Theorem 1 that the iterates (5.14) are bounded. Consider the steady state ODE defined as in (5.22)

$$\begin{aligned} \dot{v} &= \lim_{\eta\to\infty} \frac{\Phi^T D\left(R^\pi + \gamma\eta P\Phi v - \eta\Phi v\right)}{\eta} \\ &= \Phi^T D\left(\gamma P - I\right)\Phi v, \end{aligned} \tag{5.32}$$

because the diagonal matrix $D > 0$ is positive definite and for $0 < \gamma < 1$ the matrix $(\gamma P - I)$ is negative definite, the ODE (5.32) has a unique globally asymptotical stable equilibrium at the origin, provided Φ is full rank.

Assumptions 3-5 of Section 5.4 along with boundedness condition are satisfied hence the iterates (5.14) will converge to an equilibrium of (5.30), say v^π which is the unique solution of (5.26). This can be shown by following the same procedure as in [30].

5.5.2. ACTOR CONVERGENCE

Since (5.3) is of the form (5.21) the iterated θ can be approximated by an ODE

$$\dot{\theta} = \hat{\Gamma}(-\nabla J), \tag{5.33}$$

where, for any continuous function v, $\hat{\Gamma}$ is defined as

$$\hat{\Gamma}(y) = \lim_{\eta \to 0} \Big[\frac{\Gamma(\theta_k + \eta y) - \theta_k}{\eta} \Big]. \tag{5.34}$$

here $y = v(\theta_k)$. Let $\mathbb{Z}$ be the set of equilibrium points of the ODE (5.33) and $\mathbb{Z}^\epsilon$ neighborhood of $\mathbb{Z}$, i.e., $\mathbb{Z}^\epsilon = (x | \|x - z\| < \epsilon, z \in \mathbb{Z})$, the actor convergence can be stated as:

Theorem 3. Actor convergence: Under the assumption 1,2, 6, and 7, and for the error in the value function approximation $\varepsilon = {v^{\pi}}^T \phi_{\mathrm{v}}(x) - V^{\pi}(x)$ the policy iterates obtained using (5.20) converges to $\mathbb{Z}^\epsilon$, i.e., $\theta_k \to \theta^* \in \mathbb{Z}^\epsilon$ as $k \to \infty$ w.p.1 provided $\varepsilon \to 0$.

Proof: Consider the parameter update law (5.20) for the feedback controller $\beta_\theta(x)$

$$\theta_{k+1} = \Gamma(\theta_k - \alpha_{\mathrm{a}k} \nabla_\theta \beta_\theta(x) \Delta u \delta_k),$$

this can be reformulated as

$$\theta_{k+1} = \Gamma\Big(\theta_k - \alpha_{\mathrm{a}k} \mathbf{E}\left[\nabla_\theta \beta_\theta(x) \Delta u \delta_k^\pi | \mathscr{F}_k\right] - \alpha_{\mathrm{a}k} M^1 - \alpha_{\mathrm{a}k} M^2\Big), \tag{5.35}$$

where $M^1 = \left(\nabla_\theta \beta_\theta(x) \Delta u \delta_k - \mathbf{E}\left[\nabla_\theta \beta_\theta(x) \Delta u \delta_k | \mathscr{F}_k\right]\right)$ and $M^2 = \mathbf{E}\left[\nabla_\theta \beta_\theta(x) \Delta u \left(\delta_k - \delta_k^\pi\right) | \mathscr{F}_k\right]$ are martingale sequences that converge, for proof see [30]. Hence the update (5.35) can be approximated by an ODE

$$\dot{\theta} = \hat{\Gamma}\left(-\mathbf{E}\left[\nabla_\theta \beta_\theta(x) \Delta u \delta_k^\pi\right]\right). \tag{5.36}$$

The expected gradient $\mathbf{E}\left[\nabla_\theta \beta_\theta(x) \Delta u \delta_k^\pi\right]$

$$\begin{aligned}
&= \Sigma_x d^\pi(x) \Sigma_u \nabla \pi(x,u) \left(R(x,u) + \Sigma_{x'} \gamma P_{xx'}^u {v^{\pi}}^T \phi_{\mathrm{v}}(x)\right) \\
&= \Sigma_x d^\pi(x) \Sigma_u \nabla \pi(x,u) \Big(R(x,u) \\
&+ \Sigma_{x'} \gamma P_{xx'}^u V^\pi(x') + \Sigma_{x'} \gamma P_{xx'}^u \left({v^{\pi}}^T \phi_{\mathrm{v}}(x) - V^\pi(x')\right)\Big) \\
&= \nabla J + \varepsilon.
\end{aligned} \tag{5.37}$$

Using this in (5.36) and Hirsch lemma [115] it is easy to show $\theta_k \to \theta^*$ as $k \to \infty$ provided the error in value function approximation ε converges, i.e., $\varepsilon \to 0$. This can be shown by following the same procedure as in [30].

5.6. DISCUSSION

The policy gradient theorem provides a low variance parameter update rule for the stochastic policy. As explained in this chapter, in order to use the policy gradient theorem, the parameterized control law must be converted into a stochastic policy. As shown in Section 5.3, this can be done by adding a Gaussian noise to the control input. The added noise acts as an exploration term. For the algorithms presented in Chapter 4, the policy gradient is obtained online by directly interacting with the system and it is used to update the parameters of the control law. This is a general framework and can be used to

obtain the policy gradient for any parameterized policy, provided Gaussian exploration is used.

The actor and critic proof given in Section 5.5 is a generic proof. It can be used as a basis to demonstrate the convergence of any parameterized control law in the standard actor-critic setting. The convergence proof is valid provided the 7 assumptions that are introduced in this chapter are explicitly satisfied. Assumption 1 and 2 are standard requirements in the stochastic algorithm framework. For the developed actor-critic algorithms, introduced in Chapter 4, Assumption 1 and 2 are implied and hence they are not explicitly verified. Assumption 3 is used to demonstrate the stability of the steady state ODE. For the developed methods this can be easily shown and it is explicitly discussed in Section 5.5. Although assumption 4 was not explicitly verified, for the presented methods this can be readily checked by following the framework available in [30]. Assumption 7 is a generalization of Assumption 5, and it is used to ensure separate time scale for critic and actor convergence. Assumption 6 is a key element and it requires the state-space to be discrete. This can be justified by the computer control of physical systems, since in digital computers discretization in time and space is inevitable. This can be attributed to the effects of sampling and resolution, respectively. Assuming the operating-space of a system to be limited it can be discretized into a finite but large number of sections. For linearly independent basis function vector Assumption 6 is satisfied, as detailed in Section 5.5. Since many physical systems have a bounded operating space, discretization of the state-space is justified. The critic and actor convergence is based on the general framework available in the stochastic approximation theory [29, 109, 113].

6

OUTPUT SYNCHRONIZATION OF HETEROGENEOUS SYSTEMS USING REINFORCEMENT LEARNING

This chapter shifts from the standard actor-critic approach discussed in Chapter 4 to the application of reinforcement learning to solve the output tracking for linear multi-agent systems (MAS). In this heterogeneous network all the agents are assumed to be different not only in their dynamics but also in the state dimension. The available standard methods for the output synchronization of a network of heterogeneous MAS require either the solution of the output regulator equations or the incorporation of a p-copy or internal model of the leader's dynamics. By contrast, in the developed method neither one is needed. Moreover, here both the leader and the follower's dynamics is assumed to be unknown. First, a distributed adaptive observer is designed to estimate the leader's state for each agent, this does not require any knowledge of the leader's dynamics matrix. A novel model-free off-policy reinforcement learning algorithm is then developed to solve the optimal output synchronization problem online in real time. It is shown that this distributed reinforcement learning approach implicitly satisfies the output regulation equation without actually solving it. In addition the developed method does not require any knowledge of the leader's dynamics matrix or that of the agent's dynamics. The effectiveness of the proposed approach is verified by a simulation study.

6.1. INTRODUCTION

Cooperative control of multi-agent systems has undergone a paradigm shift from centralized to distributed control. This is due to the reliability, flexibility, scalability and the computational efficiency of the distributed control methods. In distributed control, unlike centralized control, there is no central authority with the ability to control the network of agents as a whole. Instead, each agent designs a controller based on the limited information about itself and its neighbors. The control objective is that all

the agents reach agreement on certain quantities of interests. If the common value that agents agree on is not specified, then the problem is called leaderless consensus. If all agents follow the trajectories of a leader node, then the problem is known as cooperative tracking (leader-follower) control. A rich body of literature has been developed for the distributed control of multi-agent systems. See for example [116–120] to name a few.

Most of the available work on distributed control focuses on the state synchronization of a homogeneous multiagent network, where individual agents have identical dynamics. In many real-world applications of multi-agent systems, the individual systems generally do not have identical dynamics. This has led to the emergence of new challenges in the design of distributed controllers for heterogeneous systems, in which the dynamics and state dimension of each of the agents can be different. Since state synchronization is not practical for general heterogeneous systems (as individual systems may have different state dimensions), distributed output synchronization of heterogeneous systems has attracted wide attention in the literature [23, 24, 121–125]. However, existing state-of-the-art methods require complete knowledge of the agents and the leader's dynamics matrix which is not available in many real-world applications. In practical applications, it is often desirable to have a model-free distributed controllers conducive to real time implementation and able to handle modeling and parameter uncertainties in the dynamics of the agents. Moreover, solutions found by these methods are generally far from optimal.

Adaptive and robust distributed controllers have been developed in the literature to adapt online to modeling uncertainties in the dynamics of the agents [126–138]. However, classical adaptive and robust distributed controllers do not converge to an optimal distributed solution. Optimal distributed control refers to a class of methods that can be used to synthesize a distributed control policy which results in best possible team behavior with respect to prescribed criteria (i.e., local control policies which leads to minimization of local performances for each agent). A suboptimal distributed controller is designed in [139] for linear homogenous systems using linear quadratic regulator. The distributed games on graphs are presented in [140] in which each agent only minimizes its own performance index. In [141], the optimal linear-consensus algorithm for multi-agent systems with single-integrator dynamics is proposed. The distributed inverse optimal control is also considered in [142]. All mentioned optimal distributed controllers are limited to state synchronization of homogeneous systems and they require complete knowledge of the agents and the leader. To our knowledge distributed adaptive optimal output synchronization is not considered in the literature.

Over the last decades there has been increasing interest in the development of multi-agent learning systems. The objective is to create agents that can learn from experience about how to interact with other agents in a best possible way [143–147]. Reinforcement learning (RL) techniques have been used prominently to design adaptive optimal controllers for both single-agent and multi-agent systems. RL algorithms enables us to solve the optimal control problem in an online manner without requiring the complete knowledge of the system dynamics. The control is learned by using only the measured data along the system trajectory. In [140], RL has been used to learn the optimal control law for each agent in a network of homogenous systems. This method requires solving a set of coupled algebraic Riccati equations (AREs) which can be extremely hard to solve.

Besides, it requires complete knowledge of the system dynamics. Finally, this method is limited to state synchronization of homogenous systems and cannot be extended for solving optimal distributed output synchronization problem.

In this chapter, a novel RL algorithm is developed to solve the output synchronization problem of a heterogeneous multi-agent system. It is shown that the explicit solution to the output regulator equation is not needed, hence the agents do not need to know the leader's dynamics. The key components of the given method are

- A distributed adaptive observer is designed to estimate the leader's state. This observer does not require the knowledge of the leader's dynamics matrix.

- A novel off-policy RL algorithm is developed to solve the output synchronization problem without requiring any knowledge of the agent's dynamics or the leader's dynamics matrix.

- It is shown that this distributed RL approach implicitly satisfies the output regulation equations without actually solving them.

The proposed approach is as follows. The estimated leader's state obtained from the presented distributed observer is used along with the local state of each agent to design a model-free optimal output synchronization controller. For each agent the objective is to track the output of an exo-system i.e., the leader in an optimal manner. To this end, the optimal output synchronization problem is cast into a set of optimal output tracking problems for each agent. A local discounted performance function is defined for each agent. Minimization of this cost function gives both feedback and feedforward gains. An online solution to the tracking problem is then found by using an off-policy RL algorithm. This algorithm does not require any knowledge of the dynamics of the agents and uses only the measured data along the system and the reference trajectories to find the optimal distributed solution to the output synchronization problem. A simulation is conducted to verify the effectiveness of the proposed method.

This chapter is organized as follows, In Section 2 the essential theoretical background is provided. The details along with the required proof of convergence of the distributed adaptive observe is given in Section 3. The off-policy RL algorithm for the output regulation is explained in Section 4. It is shown in Section 5 that the well-known separation principle is satisfied and thus the observer and the controller design problem can be treated separately. In Section 6, the simulation results for output tracking of multi-agent heterogeneous systems is given. Section 7 concludes the chapter and gives directions for future research.

6.2. THEORETICAL BACKGROUND

In this section, the essential theoretical background on graph theory is provided. The problem of output synchronization for heterogeneous multi-agent systems is also defined. The standard solution to this problem is presented and its shortcoming is emphasized.

6.2.1. GRAPH THEORY

Consider a weighted directed graph or digraph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{A}\}$ consisting of a nonempty finite set of N nodes $\mathcal{V} = (v_1, v_2, \cdots, v_N)$, a set of edges or arcs $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ and the associated adjacency matrix $\mathcal{A} = [a_{ij}] \in \mathbb{R}^{N \times N}$. Here the diagraph is assumed to be time-invariant or alternatively $\mathcal{A}$ is assumed to be constant. An edge from a node v_j to v_i is indicated by an arrow with head at node i and tail at node j, this implies that the information flow is from node j to node i. The neighbor set of node i is depicted by $N_i = \{j | (v_j, v_i) \in \mathcal{E}\}$. For each node the entry a_{ij} of the adjacency matrix $\mathcal{A}$ is nonzero (i.e., $a_{ij} > 0$) if and only if there is an edge $(v_j, v_i) \in \mathcal{A}$ else $a_{ij} = 0$, also a_{ij} indicates the weight associated with the graph edge. Only simple graphs without a self-loop is considered, this means $a_{ii} = 0$. The in-degree of a node i is defined as

$$d_i = \sum_{j=1}^{N} a_{ij}, \tag{6.1}$$

and in-degree matrix as

$$D = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_N \end{bmatrix}, \tag{6.2}$$

where $D \in \mathbb{R}^{N \times N}$. The graph Laplacian matrix is defined as

$$L = D - \mathcal{A}. \tag{6.3}$$

The product $L\mathbf{1}_N = 0$ for $\mathbf{1}_N = [1 \quad 1 \quad \cdots \quad 1]^T \in \mathbb{R}^N$. The out-degree of a node i is defined as

$$d_i^o = \sum_{j=1}^{N} a_{ji}. \tag{6.4}$$

A graph is said to be balanced if its in-degree is same as the out-degree, this implies $L^T\mathbf{1}_N = 0$. In an undirected for every edge $a_{ji} = a_{ij}$ that is the adjacency matrix $\mathcal{A}$ is symmetric. For a given digraph $\mathcal{G}$ a sequence of successive edges in the form $\Big((v_i, v_k), (v_k, v_l), \cdots \cdots, (v_m, v_j)\Big)$ gives a directed path from node i to node j. A diagraph is said to have a spanning tree if there exist a root node i_r, such that there is a directed path from i_r to every other node in the graph.

Assumption 1. The digraph $\mathcal{G}$ has a spanning tree and the leader is pinned to the root node i_r , with a pinning [1] gain $g_i > 0$.

Observe that the leader can be pinned to multiple nodes in the graph or the leader can itself be a root node. This results in a diagonal pinning matrix $G = \text{diag}[g_i] \in \mathbb{R}^{N \times N}$ with the pinning gain $g_i > 0$ if the node has access to the leader else otherwise zero. Under the above assumption, the eigenvalues of $L + G$ have positive real parts.

[1] It means the root node is connected, i.e., 'pinned' to the leader.

6.2.2. OUTPUT SYNCHRONIZATION OF HETEROGENEOUS MULTI-AGENT SYSTEMS

Consider that the dynamics of the leader or trajectory generator to be followed are

$$\dot{\zeta}_0 = S\zeta_0, \tag{6.5}$$

where $\zeta_0 \in \mathbb{R}^p$ is the reference trajectory, and $S \in \mathbb{R}^{p\times p}$ is the leader's dynamic matrix. The leader output can be defined as

$$y_0 = R\zeta_0, \tag{6.6}$$

where $y_0 \in \mathbb{R}^q$.

Assumption 2. The leader's dynamic matrix is marginally stable i.e. S in (6.5) has non-repeated eigenvalues on the imaginary axis.

The dynamics of N linear heterogeneous followers is given by

$$\begin{aligned} \dot{x}_i &= A_i x_i + B_i u_i, \\ y_i &= C_i x_i, \end{aligned} \tag{6.7}$$

where $x_i \in \mathbb{R}^{n_i}$ is the system state, $u_i \in \mathbb{R}^{m_i}$ is the input and $y_i \in \mathbb{R}^q$ is the output for $i = 1, \cdots, N$ agents. The multi-agent system is called heterogeneous because agents dynamics (A_i, B_i, C_i) and their state dimension are generally not the same.

Assumption 3. (A_i, B_i) is stabilizable and (A_i, C_i) is observable.

Problem 1. (Output Synchronization):, Design local control protocols u_i such that the outputs of all heterogeneous agents synchronize to the output of the leader node. That is, $y_i(t) - y_0(t) \to 0 \; \forall i$.

To solve this problem, standard methods in the literature require solving the output regulation equations given by

$$\begin{aligned} A_i\Pi_i + B_i\Gamma_i &= \Pi_i S, \\ C_i\Pi_i &= R, \end{aligned} \tag{6.8}$$

where $\Pi_i \in \mathbb{R}^{n_i\times p}$ and $\Gamma_i \in \mathbb{R}^{m_i\times p}$ for $i = 1, \cdots, N$ are the solution of the output regulator equation (6.8). Based on these solutions, the following standard controller guarantees the output tracking among heterogeneous agents [24, 123],

$$u_i = K_{1i}(x_i - \Pi_i\zeta_0) + \Gamma_i\zeta_0, \tag{6.9}$$

where $K_{1i} \in \mathbb{R}^{m_i\times m_i}$ is the state-feedback gain which stabilizes $A_i + B_iK_{1i}$. The tracking control law (6.9) depends on the agent's state and the leader's state. However, the leader state ζ_0 is generally not available to all agents in a distributed multi-agent network. This issue is circumvented in the literature [117] by designing the following local observer

called synchronizer. This local observer is used in order to obtain an estimate of the leader's trajectory in all the agents

$$\dot{\zeta}_i = S\zeta_i + c\left[\sum_{j=1}^{N} a_{ij}(\zeta_j - \zeta_i) + g_i(\zeta_0 - \zeta_i)\right], \tag{6.10}$$

resulting in the modified tracking law

$$u_i = K_{1i}(x_i - \Pi_i\zeta_i) + \Gamma_i\zeta_i, \tag{6.11}$$

where ζ_i is the estimation of ζ_0 for the agent i and the constant c is the coupling gain. The output synchronization is guaranteed when the control protocol (6.11) is applied to the multi-agent system.

The observer estimation error for agent i is defined as

$$\delta_i(t) = \zeta_i(t) - \zeta_0(t), \tag{6.12}$$

and the local neighborhood observation error for node i is defined as

$$e_i = \sum_{j=1}^{N} a_{ij}(\zeta_j - \zeta_0) + g_i(\zeta_0 - \zeta_i). \tag{6.13}$$

Remark 1. Note that the solution to the output regulator equation (6.8) for each agent requires the complete knowledge of the leader's dynamic matrix, i.e., S , which is overly conservative. Moreover, all the agents need to be aware of their own dynamics, i.e. (A_i, B_i, C_i), to solve the output regulator equation (6.8) and to obtain feedforward component K_{1i} . This knowledge, however, is not available in many applications.

6.3. DISTRIBUTED ADAPTIVE OBSERVER DESIGN

In the previous section, a standard solution to output regulation for heterogeneous multi-agent systems was given. The standard approach requires the solution of the output regulator equations (6.8). This needs the full knowledge of the leader's dynamics (S, R), and the agent's dynamics (A_i, B_i, C_i).

In this section, a novel distributed adaptive observer is designed to estimate the leader's state for all the agents. In contrast to the standard observer (6.10), the proposed method does not require the knowledge of the leader's dynamic matrix S. In the next section, it is shown how to use this adaptive observer along with reinforcement learning to solve Problem 1 without solving the regulator equations (6.8) and without knowing the agent's dynamics.

To estimate the leader's state, the following distributed observer is used.

$$\dot{\zeta}_i = \hat{S}_i\zeta_i + c\left[\sum_{j=1}^{N} a_{ij}(\zeta_j - \zeta_i) + g_i(\zeta_0 - \zeta_i)\right], \tag{6.14}$$

where $\hat{S}_i \in \mathbb{R}^{p\times p}$ is the estimation of the leader's dynamic matrix S for node i. Using the

tuning law for the leader's dynamics estimator $\hat{S}_i$ as

$$\begin{aligned}\dot{\hat{S}}_{\text{vec}i} &= \Pi\left(-\Gamma_{Si}(I_q \otimes \zeta_i)\left[\sum_{j=1}^{N} a_{ij}(\zeta_j - \zeta_i) + g_i(\zeta_0 - \zeta_i)\right]\right), \\ &= \Pi\left(-\Gamma_{Si}(I_q \otimes \zeta_i)e_i\right) \end{aligned} \tag{6.15}$$

where $\hat{S}_{\text{vec}i}$ is the vector representation of $\hat{S}_i$, Γ_{Si} is the diagonal positive update rate matrix, and $\otimes$ is the Kronecker product [148]. The vector form is obtained by stacking each row of $\hat{S}_i$, and Π is the projection operator to ensure boundedness of the update $\hat{S}_{\text{vec}i}$. Note that the projection operator is not used in the simulation analysis, it is only used to assume boundedness of the estimate $\hat{S}_{\text{vec}i}$ in the proof of convergence for the local adaptive synchronizer.

The following two theorems shows the convergence of the local estimator ζ_i. The first theorem proves that the estimation error $\delta_i(t)$ in (6.12) converges to zero, i.e., $\delta_i(t) \to 0$ for an undirected graph, whereas the second theorem demonstrates that the estimation error can be made arbitrarily small $\delta_i(t) < \epsilon$ for a generic directed graph.

Theorem 1. For a symmetric graph Laplacian matrix, i.e., $L = L^T$ [2]. Based on Assumption 1, 2, consider the distributed observer and the update law given in (6.14) and (6.15), respectively. Then, the observer estimation error (6.12) converges to zero, i.e., $\delta_i(t) \to 0$, provided the constant c in (6.14) is chosen large enough.

Proof Differentiating (6.12) and by using (6.5) and (6.14) gives the local estimation error dynamics for each observer

$$\dot{\delta}_i = \hat{S}_i\zeta_i + c\left[\sum_{j=1}^{N} a_{ij}(\zeta_j - \zeta_i) + g_i(\zeta_0 - \zeta_i)\right] - S\zeta_0, \tag{6.16}$$

which can be rewritten as

$$\dot{\delta}_i = S_i\zeta_i + c\left[\sum_{j=1}^{N} a_{ij}(\zeta_j - \zeta_i) + g_i(\zeta_0 - \zeta_i)\right] - S\zeta_0 + (\hat{S}_i - S)\zeta_i. \tag{6.17}$$

The global error dynamics becomes

$$\dot{\delta} = \left[I_N \otimes S - c(L+G) \otimes I_p\right]\delta + \tilde{S}\zeta \tag{6.18}$$

where $\delta = [\zeta - \underline{\zeta}]$ in terms of the vectors $\zeta = [\zeta_1^T(t)\,\zeta_2^T(t)\cdots\zeta_N^T(t)]^T$ and $\underline{\zeta}_0 = [\zeta_0^T(t)\,\zeta_0^T(t)\cdots\zeta_0^T(t)]^T$. The error in the leader's dynamics estimation is in $\tilde{S} = \text{diag}[\hat{S}_1 - S, \hat{S}_2 - S, \cdots, \hat{S}_N - S]$. Equation (6.18) in compact form is

$$\dot{\delta} = A_s\delta + \zeta_M\tilde{S}_{\text{vec}} \tag{6.19}$$

where

$$A_s = I_N \otimes S - c(L+G) \otimes I_p, \tag{6.20}$$

$$\zeta_M = \text{diag}\left[I_p \otimes \zeta_1^T, I_p \otimes \zeta_2^T, \cdots, I_p \otimes \zeta_N^T\right], \tag{6.21}$$

[2] This is possible when the subgraph consisting of N followers forms an undirected graph.

where $\tilde{S}_{\text{vec}}$ is the vector representation of the error in leader dynamics estimation.

The error dynamics matrix A_s defined in (6.20) can be made Hurwitz for an appropriate choice of the constant c, because $L+G$ is nonsingular and has eigenvalues with positive real part. Now consider the Lyapunov function

$$V = \delta^T \left[(L+G)\otimes I_p\right]\delta + \tilde{S}_{\text{vec}}^T \Gamma_S^{-1} \tilde{S}_{\text{vec}} \tag{6.22}$$

since the subgraph with N followers is undirected this makes $\left[(L+G)\otimes I_p\right]$ symmetric positive definite [124], and Γ_S is a diagonal positive definite scaling matrix. The derivative of the Lyapunov function is

$$\begin{aligned}\dot{V} &= \dot{\delta}^T \left[(L+G)\otimes I_p\right]\delta + \delta^T P \left[(L+G)\otimes I_p\right]\dot{\delta} + \dot{\tilde{S}}_{\text{vec}}^T \Gamma_S^{-1} \tilde{S}_{\text{vec}} + \tilde{S}_{\text{vec}}^T \Gamma_S^{-1} \dot{\tilde{S}}_{\text{vec}} \\ &= \delta^T \left(\left[(L+G)\otimes I_p\right] A_S + A_S^T \left[(L+G)\otimes I_p\right]\right)\delta + \tilde{S}_{\text{vec}}^T \zeta_M^T \left[(L+G)\otimes I_p\right]\delta \\ &+ \delta^T \left[(L+G)\otimes I_p\right]^T \zeta_M \tilde{S}_{\text{vec}} + \dot{\tilde{S}}_{\text{vec}}^T \Gamma_S^{-1} \tilde{S}_{\text{vec}} + \tilde{S}_{\text{vec}}^T \Gamma_S^{-1} \dot{\tilde{S}}_{\text{vec}}\end{aligned} \tag{6.23}$$

By choosing

$$\tilde{S}_{\text{vec}} = \hat{S}_{\text{vec}} = -\Gamma_S \zeta_M^T P \delta \tag{6.24}$$

the Lyapunov derivative becomes

$$\dot{V} = \delta^T \left(\left[(L+G)\otimes I_p\right] A_S + A_S^T \left[(L+G)\otimes I_p\right]\right)\delta. \tag{6.25}$$

In order to demonstrate the negative semi-definiteness of $\dot{V}$, we need to show $\left[(L+G)\otimes I_p\right] A_S + A_S^T \left[(L+G)\otimes I_p\right] < 0$. For the sake of plainness let us redefine $I_N \otimes S = M$ and $(L+G)\otimes I_q = N$, where M has no eigenvalues with positive real parts, and N is non-singular. The Lyapunov equation $\left[(L+G)\otimes I_p\right] A_S + A_S^T \left[(L+G)\otimes I_p\right]$ in compact form is

$$\begin{aligned}&= N(M - cN) + (M^T - cN^T)N^T \\ &= (NM + M^T N^T) - c(NN + N^T N^T).\end{aligned} \tag{6.26}$$

Where the property for the undirect graph $\left[(L+G)\otimes I_p\right] = \left[(L+G)\otimes I_p\right]^T$ is used in the first equation. It is well known that for any given Hermitian matrices E, F, and some constant c, the eigenvalue of the matrix sum is

$$\lambda_{i+j-1}(E - cF) = \lambda_i(E) - c\lambda_j(F) \tag{6.27}$$

for $i + j \leq N+1, i \leq N, j \leq N$. Thus, if c is greater than a certain bound, one can ensure that the eigenvalues of matrix sum $E - cF$ to have the negative real part. Note that the terms in (6.26) are of form (6.27), hence the eigenvalues of the terms

$$(NM + M^T N^T) - c(NN + N^T N^T)$$

has negative real part provided that c is large enough. Additionally, the overall matrix $(NM + M^T N^T) - c(NN + N^T N^T)$ is symmetric, as it is obtained by addition and subtraction of the symmetric matrices. This confirms that the eigenvalues of $\left[(L+G)\otimes I_p\right] A_S +$

$A_S^T\left[(L+G)\otimes I_p\right]$ are negative real and thus proves the negative semi-definiteness of the Lyapunov derivative (6.25). That is,

$$\dot{V} = -\delta^T Q\delta \leq 0 \tag{6.28}$$

for some $Q \geq 0$. This shows the convergence of the local synchronizer, i.e., $\delta_i \to 0$. Since all the entries in the update rule (6.24) are block diagonal the parameter update rule for each agent gives (6.15) this completes the proof.

Theorem 2. For a given generic directed graph. Based on Assumption 1, 2, consider the distributed observer and the update law given in (6.14) and (6.15), respectively. Then, the observer estimation error (6.12) can be made arbitrarily small, i.e., $\delta_i(t) < \epsilon$ provided the constant c in (6.14) is chosen large enough.

Proof Consider the global observation error

$$e = \left[(L+G)\otimes I_p\right]\left(\zeta(t) - \underline{\zeta}_0(t)\right). \tag{6.29}$$

Differentiating (6.29), and by using (6.5) and (6.14) gives the estimation error dynamics as

$$\dot{e} = \left[(L+G)\otimes I_p\right]\begin{pmatrix} \hat{S}_1\zeta_1 + c\left[\sum_{j=1}^N a_{1j}(\zeta_j-\zeta_1) + g_1(\zeta_0-\zeta_1)\right] - S\zeta_0 \\ \hat{S}_2\zeta_2 + c\left[\sum_{j=1}^N a_{2j}(\zeta_j-\zeta_2) + g_2(\zeta_0-\zeta_2)\right] - S\zeta_0 \\ \vdots \\ \hat{S}_N\zeta_N + c\left[\sum_{j=1}^N a_{Nj}(\zeta_j-\zeta_N) + g_N(\zeta_0-\zeta_N)\right] - S\zeta_0 \end{pmatrix} \tag{6.30}$$

By adding and subtracting $S\zeta_i$ the error dynamics (6.30) becomes

$$\dot{e} = \left((L+G)\otimes I_p\right)\left[(I_N\otimes S)\left(\zeta(t) - \underline{\zeta}_0(t)\right) - c\left((L+G)\otimes I_p\right)\left(\zeta(t) - \underline{\zeta}_0(t)\right) + \tilde{S}\zeta\right]. \tag{6.31}$$

Using the matrix property $(A\otimes I)(I\otimes B) = (I\otimes B)(A\otimes I)$, (6.31) can be rewritten as

$$\dot{e} = \left[(I_N\otimes S) - c\left((L+G)\otimes I_p\right)\right]e + \left((L+G)\otimes I_p\right)\tilde{S}\zeta. \tag{6.32}$$

Now consider the Lyapunov function

$$V = \delta^T P\delta + \tilde{S}_{\text{vec}}^T\Gamma_S^{-1}\tilde{S}_{\text{vec}} \tag{6.33}$$

where P is positive definite and it is obtained as

$$\begin{aligned} p &= \left((L+G)\otimes I_p\right)^{-T}\mathbf{1}_{Nq}, \\ P &= \text{diag}[p] \end{aligned} \tag{6.34}$$

where $\mathbf{1}_{Nq}$ is vector with all entries 1. Taking the derivative of (6.33) using (6.32) is

$$\begin{aligned} \dot{V} = {} & e^T\left[(I_N\otimes S)^T P + P(I_N\otimes S)\right]e - ce^T\left[\left((L+G)\otimes I_p\right)^T P + P\left((L+G)\otimes I_p\right)\right]e \\ & + e^T P\left((L+G)\otimes I_p\right)\tilde{S}\zeta + \zeta^T\tilde{S}^T\left((L+G)\otimes I_p\right)^T Pe \\ & + \dot{\tilde{S}}_{\text{vec}}^T\Gamma_S^{-1}\tilde{S}_{\text{vec}} + \tilde{S}_{\text{vec}}^T\Gamma_S^{-1}\dot{\tilde{S}}_{\text{vec}}. \end{aligned} \tag{6.35}$$

The first term $\left[(I_N \otimes S)^T P + P(I_N \otimes S)\right]$ is skew-symmetric hence the norm is 0, whereas the second term $\left[\left((L+G)\otimes I_p\right)^T P + P\left((L+G)\otimes I_p\right)\right] = Q$, for a positive definite matrix Q. Hence the Lyapunov derivative (6.35) reduced to

$$\begin{aligned}\dot{V} = &-ce^T Qe \\ &+ e^T P\left((D+G-A)\otimes I_p\right)\tilde{S}\zeta + \zeta^T \tilde{S}^T\left((D+G-A)\otimes I_p\right)^T Pe \\ &+ \dot{\tilde{S}}_{\text{vec}}^T \Gamma_S^{-1}\tilde{S}_{\text{vec}} + \tilde{S}_{\text{vec}}^T\Gamma_S^{-1}\dot{\tilde{S}}_{\text{vec}}.\end{aligned} \tag{6.36}$$

Using the matrix property $(A+B)\otimes C = (A\otimes C)+(B\otimes C)$, equation (6.36) will be

$$\begin{aligned}\dot{V} = &-ce^T Qe \\ &- e^T P\left(A\otimes I_p\right)\tilde{S}\zeta - \zeta^T\tilde{S}^T\left(A\otimes I_p\right)^T Pe \\ &+ \tilde{S}_{\text{vec}}^T\zeta_M^T\left[(D+G)\otimes I_p\right]^T Pe + e^T P\left[(D+G)\otimes I_p\right]\zeta_M\tilde{S}_{\text{vec}} \\ &+ \dot{\tilde{S}}_{\text{vec}}^T\Gamma_S^{-1}\tilde{S}_{\text{vec}} + \tilde{S}_{\text{vec}}^T\Gamma_S^{-1}\dot{\tilde{S}}_{\text{vec}}.\end{aligned} \tag{6.37}$$

By choosing

$$\dot{\tilde{S}}_{\text{vec}} = \dot{\hat{S}}_{\text{vec}} = -\Gamma_S\zeta_M^T\left[(D+G)\otimes I_p\right]^T Pe, \tag{6.38}$$

the block diagonal structure of (6.38) results in (6.15). The Lyapunov derivative (6.37) using (6.38) is

$$\dot{V} = -ce^T Qe - e^T P\left(A\otimes I_p\right)\tilde{S}\underline{\zeta}_0, \tag{6.39}$$

Observe that $\tilde{S}$ is bounded due to the boundedness of $\hat{S}$ and S, also $\underline{\zeta}_0$ is bounded due to marginally stable leader. This results in

$$\dot{V} \leq -c\underline{\sigma}(Q)\|e\|^2 - \nu S_M\underline{\sigma}(A)\underline{\sigma}(P)\|e\|\zeta_{0M}, \tag{6.40}$$

where, S_M is the maximum absolute bound on the parameter estimation error and ζ_{0M} is the maximum absolute bound on the leader's state. Now by choosing a large c and a relatively small ν the observer error can be made arbitrarily small. This shows that provided c is large enough, the convergence of the local synchronizer, is bounded by an arbitrarily small error ε i.e., $\delta_i(t) < \epsilon$. This completes the proof.

Remark 2. Theorem 1 provides the proof of convergence for the local synchronizer i.e., $\delta_i \to 0$. Note that the convergence of the parameter $\hat{S}_i$ to the true leader dynamics S cannot be guaranteed. In fact, the convergence of the $\hat{S}_i$ to the true dynamics is not required. As the reinforcement learning based optimal tracking control law presented in Section 6.6 doesn't need the knowledge of S .

6.4. OPTIMAL MODEL-FREE OUTPUT REGULATION

In this section, a reinforcement learning (RL) algorithm is proposed to make the agents track the leader's output using an optimal tracking control of each agent. Based on the adaptive observer of Section 6.3, it is assumed that every agent has a local estimate of the leader's trajectory. In Section 6.5, this RL based optimal control will be combined with

the distributed adaptive synchronizer of Section 6.3. Due to this combination, the design of the output synchronizing controller does not require either the leader's dynamics S or the agent's dynamics (A_i, B_i, C_i). This is because solution to (6.8) is not explicitly needed.

Consider a linear continuous-time system with the following dynamics

$$\begin{aligned} \dot{x}_i &= A_i x_i + B_i u_i \\ y_i &= C_i x_i \end{aligned} \tag{6.41}$$

where $x_i \in \mathbb{R}^{n_i}$ is the system state, $y_i \in \mathbb{R}^q$ is the system output, $u_i \in \mathbb{R}^{m_i}$ is the control input, $A_i \in \mathbb{R}^{n_i \times n_i}$ gives the drift dynamics of the system, and $B_i \in \mathbb{R}^{n_i \times m_i}$ is the input matrix. It is assumed that the pair (A_i, B_i) is stabilizable and the pair (A_i, C_i) is observable.

Assume that the reference trajectory $\zeta_0 \in \mathbb{R}^q$ is bounded and it is generated by the command generator system given by (6.5) and (6.6). Assume that S in (6.5) is a marginally stable matrix with appropriate dimension.

In optimal output regulation problem, the goal is to find a control policy to make the system output y_i in (6.41) follow the reference trajectory output y_0 generated by (6.5) and (6.6), while minimizing a predefined performance function. Define the discounted performance function for the system (6.41) as [149]

$$V(x_i(t), u_i(t)) = \int_t^\infty e^{-\gamma_{c_i}(\tau - t)} \Big((y_i - y_0)^T Q_i (y_i - y_0) + u_i^T W_i u_i \Big) d\tau \tag{6.42}$$

where the state weight matrix Q_i and the control input weight matrix W_i are symmetric positive definite, and $\gamma_{c_i} > 0$ is the discount factor.

Remark 3. The discount factor $\gamma_{c_i} > 0$ in (6.42) is used to ensure that the performance function is bounded for a given control policy which assures the output regulation. This is because the steady state part of the control input does not go to zero unless the command generator dynamics is stable.

Consider a fixed state-feedback control policy linear in the system state and the command generator state as

$$u_i = K_{1i} x_i + K_{2i} \zeta_0 \tag{6.43}$$

and define an augmented state as

$$X_i(t) = \left[x_i(t)^T \ \ \zeta_0^T \right]^T \in \mathbb{R}^{n_i + p} \tag{6.44}$$

where ζ_0 is given by (6.5). The control input (6.43) in terms of the augmented state (6.44) becomes

$$u_i = K_{1i} x_i + K_{2i} = K_i X_i \tag{6.45}$$

where $K_i = [K_{1i} \ K_{2i}]$. Moreover, the augmented dynamics, with an abuse of notation become [149]

$$\dot{X}_i = T_i X_i + B_{1i} u_i \tag{6.46}$$

with

$$T_i = \begin{bmatrix} A_i & 0 \\ 0 & S \end{bmatrix}, B_{1i} = \begin{bmatrix} B_i \\ 0 \end{bmatrix} \tag{6.47}$$

Finally, the value function for a control policy in form of (6.45) can be written as the quadratic form [149]

$$\begin{aligned} V(X_i(t)) &= \int_t^\infty e^{-\gamma_{c_i}(\tau-t)} X_i^T \left(C_{1i}^T Q_i C_{1i} + K_i^T W_i K_i\right) X_i \, d\tau \\ &= X_i^T(t) P_i X_i(t) \end{aligned} \tag{6.48}$$

where

$$C_{1i} = [C_i \;\; -R] \tag{6.49}$$

with R as in (6.6). The optimal control input is then given by $u_i = K_i X_i$ [149] with

$$K_i = [K_{1i} \;\; K_{2i}] = -W_i^{-1} B_{1i}^T P_i \tag{6.50}$$

where P_i is the solution to the discounted algebraic Riccati equation (ARE)

$$T_i^T P_i + P_i T_i - \gamma_{c_i} P_i + C_{1i}^T Q_i C_{1i} - P_i B_{1i} W_i^{-1} B_{1i}^T P_i = 0 \tag{6.51}$$

The ARE (6.51) is first solved for P_i. Then the optimal gain is obtained by substituting the ARE solution to (6.50).

6

6.4.1. AN UPPER BOUND FOR DISCOUNT FACTOR TO ASSURE ASYMPTOTIC OUTPUT REGULATION

In this subsection, an upper bound is found for the discount factor in the performance function (6.42) to assure that the tracking error $e_{ri} = y_i - y_0$ goes to zero asymptotically, when the optimal control gain (6.50) found by solving the ARE (6.51) is applied to the system. In [149], the authors showed that the control gain given in (6.50) makes $e^{-\gamma_{c_i} t} e_{ri}$ converge to zero asymptotically. However, the tracking error may diverge if the discount factor is not chosen appropriately. The following theorem shows that perfect output regulation is achieved if (6.50) is applied to the system and the discount factor is chosen small enough.

Theorem 3. From Assumption 2 and 3, the system (6.41) is stabilizable and the command generator (6.5) is marginally stable. Let the control input (6.45) with gain given by (6.50), (6.51) be applied to the system. Then, $A_i + B_i K_{1i}$ is Hurwitz and the tracking error $e_{ri} = y_i - y_0$ goes to zero asymptotically fast, if the discount factor satisfies the following condition

$$\gamma_{c_i} \leq \gamma_{c_i}^* = 2\|(B_i W_i^{-1} B_i^T Q_i)^{1/2}\| \tag{6.52}$$

Proof: We first show that $A_i + B_i K_{1i}$ is Hurwitz. To this end, define

$$P_i = \begin{bmatrix} P_{11}^i & P_{12}^i \\ P_{21}^i & P_{22}^i \end{bmatrix} \tag{6.53}$$

Then, using (6.47), for the upper left-hand side of the discounted ARE (6.51) one has

$$A_i^T P_{11}^i A_i - \gamma_{c_i} P_{11}^i + C_i^T Q_i C_i - P_{11}^i B_i W_i^{-1} B_i^T P_{11}^i = 0 \tag{6.54}$$

and the control gain K_{1i} becomes

$$K_{1i} = -W_i^{-1} B^T P_{11}^i \tag{6.55}$$

Since $Q_i > 0$ and (A_i, C_i) is observable, then $(A_i, Q_i^{1/2} C_i)$ is observable and thus there exists an unique positive definite solution P_{11}^i to (6.54). It is shown in [150] that if condition (6.52) is satisfied, then the eigenvalues of the closed-loop system $A_i - B_i W_i^{-1} B_i^T P_{11}^i = A_i + B_i K_{1i}$ have negative definite parts and thus $A_i + B_i K_{1i}$ is Hurwitz. On the other hand, it is shown in [149] that there exists a positive semi-definite solution to ARE (6.51) if (A_i, B_i) is stabilizable and $S - 0.5\gamma_{c_i} I$ is stable. Since (A_i, B_i) is assumed stabilizable and S is assumed marginally stable, existence of a positive semi-definite solution to the ARE (6.51) is guaranteed. Multiplying the left and right-hand sides of the ARE (6.51) by X_i^T and X_i , respectively, one has

$$2X_i^T T_i^T P_i X_i - \gamma_{c_i} X_i^T P_i X_i + X_i^T C_{1i}^T Q_i C_{1i} X_i - (P_i X_i)^T B_{1i} W_i^{-1} B_{1i}^T (P_i X_i) = 0 \tag{6.56}$$

From this equation one can see that if $P_i X_i = 0$ then $X_i^T C_{1i}^T Q_i C_{1i} X_i = 0$. That is, the null space of P_i is a subspace of the null space of $C_{1i}^T Q_i C_{1i}$. This indicates that if $X_i^T P_i X_i = 0$ then $X_i^T C_{1i}^T Q_i C_{1i} X_i$ and thus $(y_i - y_0)^T Q_i (y_i - y_0) = 0$ which yields $e_{ri} = y_i - y_0 = 0$. Therefore, the null space of P_i is in fact a subspace of the space in which the tracking error is zero. Now, consider the following Lyapunov function

$$V_i(X_i) = X_i^T P_i X_i \geq 0 \tag{6.57}$$

To complete the proof, it remains to show that $\dot{V}_i(X_i) < 0$ if $X_i^T P_i X_i \neq 0$ and (6.52) is satisfied. This is because since $P_i \geq 0$, if $\dot{V}_i(X_i) < 0$, then $\dot{V}_i(X_i) = \dot{X}_i^T P_i X_i = 0$ and consequently $P_i X_i = 0$ which conclude the tracking error is zero. On the other hand, if $\dot{V}_i(X_i) < 0$, then, starting from any initial trajectory, it converges to the null space of P_i which is a subspace of the space of the solutions in which the tracking error is zero. To show that $\dot{V}_i(X_i) < 0$ if (6.52) is satisfied for all X_i such that $P_i X_i \neq 0$, taking the derivative of $V_i(X_i)$ gives

$$\dot{V}_i(X_i) = X_i^T (P_i A_{ci} + A_{ci}^T P_i) X_i \tag{6.58}$$

where

$$A_{ci} = \begin{bmatrix} A_i + B_i K_{1i} & B_i K_{2i} \\ 0 & S \end{bmatrix} \tag{6.59}$$

is the closed-loop dynamics. Assume now that λ_k is an eigenvalue of A_{ci} and X_k is its corresponding eigenvector. That is,

$$A_{ci} X_k = \lambda_k X_k, \; k = 1, \cdots, n_i + p. \tag{6.60}$$

Assuming for simplicity that A_{ci} is diagonalizable, then for any arbitrary vector X_i one has

$$X_i = \sum_{k=1}^{n_i+p} \alpha_k X_k \tag{6.61}$$

for some α_k. Using (6.59) and (6.60) in (6.58) yields

$$\dot{V}_i(X_i) = 2 \sum_{k=1}^{n_i+p} \alpha_k^2 Re(\lambda_k) X_k^T P_i X_k \tag{6.62}$$

If condition (6.52) is satisfied, then $A_i+B_iK_{1i}$ is Hurwitz and since S is assumed marginally stable, one has $Re(\lambda_k)<0\ \forall k=1:n_i$ and $Re(\lambda_k)=0\ \forall k=n_i+1:n_i+p$ for the eigenvalues of A_{ci} in (6.59) . Therefore, if $P_iX_i\neq 0$ and (6.52) is satisfied, then $\dot{V}_i(X_i)<0$ and this completes the proof.

6.4.2. MODEL-FREE OFF-POLICY REINFORCEMENT LEARNING FOR SOLVING OPTIMAL OUTPUT REGULATION

In this subsection, a state-feedback off-policy integral reinforcement learning (IRL) algorithm is given to learn the solution to the discounted optimal output regulation problem. This algorithm does not require any knowledge of the system dynamics or the leader's dynamics matrix S. In order to obviate the requirement of the knowledge of the system dynamics, the off-policy IRL algorithm was proposed in [150] for solving the optimal regulation problem with an undiscounted performance function. This method is extended in [149] for discounted performance functions such that it can be used for solving optimal output tracking problems. To this end, the system dynamics (6.46) is first written as

$$\dot{X}_i = T_iX_i + B_{1i}(-K_i^*X_i+u_i) \tag{6.63}$$

With the abuse of notation $T_i=T_i+B_{1i}K_i^*$. Then, the Bellman equation becomes [149]

$$\begin{aligned}
&e^{-\gamma_{c_i}\delta t}X_i(t+\delta T)^TP_i^\kappa X_i(t+\delta t)-X_i(t)^TP_i^\kappa X_i(t)\\
&=\int_t^{t+\delta t}\frac{d}{d\tau}(e^{-\gamma_{c_i}(\tau-t)}X_i^TP_i^\kappa X_i)d\tau\\
&=\int_t^{t+\delta t}e^{-\gamma_{c_i}(\tau-t)}\big[X_i^T(T_i^TP_i^\kappa+P_i^\kappa T_i-\gamma_{c_i}P_i^\kappa)X_i+2(u_i-K_i^\kappa X_i)^TB_{1i}^TP_i^\kappa X_i\big]d\tau\\
&=\int_t^{t+\delta t}e^{-\gamma_{c_i}(\tau-t)}X_i^TQ_iX_id\tau+2\int_t^{(t+\delta t)}e^{-\gamma_{c_i}(\tau-t)}(u_i-K_i^\kappa X_i)^TW_iK_i^{\kappa+1}X_id\tau
\end{aligned} \tag{6.64}$$

where $Q_i=C_{1i}^TQ_iC_{1i}+(K_i^\kappa)^TW_iK_i^\kappa$. For a fixed control gain K_i^κ , (6.64) can be solved for both the kernel matrix P_i^κ and the improved gain $K_i^{\kappa+1}$, simultaneously. The following Algorithm 1 uses the above Bellman equation to iteratively solve the ARE equation (6.51).

Algorithm 1. Online Off-policy IRL State-feedback algorithm

1. Initialization: Start with a control policy $u_i^\kappa=K_i^0X_i+e$, where K_i^κ is stabilizing and e is the probing noise.
2. Solve the following Bellman equation for P_i^κ and $K_i^{\kappa+1}$ simultaneously.

$$\begin{aligned}
&e^{-\gamma_{c_i}\delta t}X_i(t+\delta t)^TP_i^\kappa X_i(t+\delta t)-X_i(t)^TP_i^\kappa X_i(t)\\
&=-\int_t^{t+\delta t}e^{-\gamma_{c_i}(\tau-t)}X_i^TQ_iX_id\tau+2\int_t^{t+\delta t}e^{-\gamma_{c_i}(\tau-t)}(u_i-K_i^\kappa X_i)^TW_iK_i^{\kappa+1}X_id\tau
\end{aligned} \tag{6.65}$$

3. Stop if convergence is achieved, otherwise set $\kappa=\kappa+1$ and got to 2.
4. On convergence set $K_i=K_i^\kappa$.

In Algorithm 1, the control policy which is applied to the systems, i.e. u_i, can be a fixed stabilizing policy. The data which is gathered by applying this fixed policy to the system is then used in (6.65) to find the value function kernel matrix P_i^κ and the improved policy $u_i^{\kappa+1} = K_i^{\kappa+1} X_i$ corresponds to an updated policy $u_i = K_i X_i$

6.5. OPTIMAL MODEL-FREE OUTPUT REGULATION FOR A MULTI-AGENT HETEROGENEOUS SYSTEM

In this section, the distributed observer and the optimal tracking control from previous two sections are combined, to solve Problem 1. The block diagram representation of the presented approach is shown in Figure 6.1. The developed approach, unlike the stan-

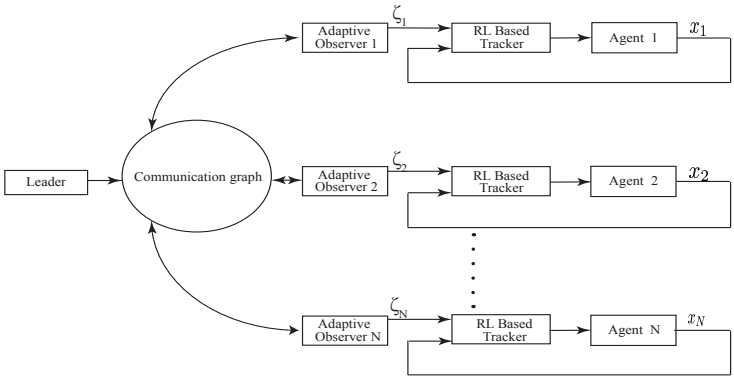


Figure 6.1: Block diagram representation of the proposed approach.

dard method (6.11), does not require the explicit solution of the output regulator equation (6.8). However, it is shown that this distributed reinforcement learning approach implicitly solves the output regulation equations. The optimal control law (6.45) for a single-agent system (6.41) depends on the leader's state ζ_0. But, in a distributed multi-agent network, only few agents will be aware of the leader's trajectory. Hence, the control law (6.45) cannot be used for all the agents. However, as explained in Section 6.3, by using the local adaptive synchronizer (6.14) and the corresponding update law (6.15), every agent can get a local estimation of the leader's state ζ_0 denoted by ζ_i. By using the local estimate ζ_i in (6.45), the modified optimal tracking controller for each agent is

$$u_i = K_{1i} x_i + K_{2i} \zeta_i \equiv K_i X_i \tag{6.66}$$

where K_i is obtained using the online Algorithm 1. Note that the tracking control (6.66) is optimal and does not depend on either the agent's system matrices (A_i, B_i, C_i) or the leader's dynamics matrix S.

The proof of the asymptotic convergence of the distributed observer and the optimal tracker are given in Sections 6.3 and 6.4, respectively. In the following theorem this results are combined to achieve output-synchronization of multi-agent heterogeneous systems.

Theorem 4. Consider the distributed adaptive synchronizer (6.14) and the optimal tracking control solution (6.66) obtained using Algorithm 1 for each agent i. Then the output synchronization problem is solved for $i = 1, \cdots, N$ as $t \to \infty$, i.e., for the given agents $y_i(t) - y_0(t) \to 0 \forall i$, provided that c in (6.14) is sufficiently large and the discount factor γ_{c_i} is less than the bound (6.52).

Proof : Using the control law (6.66), consider the augmented dynamics for a single agent

$$\begin{bmatrix} \dot{x}_i \\ \dot{\zeta}_i \end{bmatrix} = \begin{bmatrix} A_i + B_i K_{1i} & B_i K_{2i} \\ 0 & S_i \end{bmatrix} \begin{bmatrix} x_i \\ \zeta_i \end{bmatrix} + \begin{bmatrix} 0 \\ e_i \end{bmatrix} \tag{6.67}$$

along with the adaptive law given by (6.15)

$$\dot{\hat{S}}_{\text{vec}i} = -\Gamma_{Si}(I_q \otimes \zeta_i) e_i \tag{6.68}$$

where e_i is defined in (6.13). Due to the block-triangular structure, the observer dynamics is independent of the agent state x_i , thus based on the separation principle the observer and the tracking control can be designed independent of each other. In Theorem 1 it is shown, $\zeta_i(t) - \zeta_0(t) \to 0, t \to \infty, \forall i = 1, \cdots, N$. For any full rank matrix R, $R\zeta_i(t) - R\zeta_0(t) \to 0, t \to \infty$, i.e., $R\zeta_i(t) - y_0(t) \to 0, t \to \infty$. Now based on Theorem 2, $y_i(t) - R\zeta_i(t) \to 0, t \to \infty$, this proves $y_i(t) - y_0(t) \to 0, t \to \infty, \forall i$.

Remark 4. This theorem illustrates the separation principle for output regulation of heterogeneous multi-agent systems. It also shows that the explicit solution for the output regulator equation (6.8) is not necessary since tracking is achieved by controller (6.66), which is learned online using Algorithm 1 for each agent. By ensuring the observer is faster than the learning agent we can reduce the effects of transient. Alternatively, first the distributed adaptive synchronizer (6.14) can be used to estimate the leader's state trajectory and then only agents can learn the optimal controller using Algorithm 1.

The following lemma demonstrates that the output regulator equations (6.8) are intrinsically satisfied for the optimal tracking control (6.66).

Lemma 1. Consider a network of heterogeneous multi-agent systems (6.7), and the leader (6.5). The control law (6.66) obtained using Algorithm 1 implicitly solves the output regulation equations

$$\begin{aligned} A_i \Pi_i + B_i \Gamma_i &= \Pi_i S \\ C_i \Pi_i &= R \end{aligned} \tag{6.69}$$

where $\Pi_i \in \mathbb{R}^{n_i \times p}$ and $\Gamma_i \in \mathbb{R}^{m_i \times p}$ are unique nontrivial matrices. Moreover, if the gain K_{1i} in (6.66) and (6.9) are same then $K_{2i} = \Gamma_i - K_{1i}\Pi_i$.

Proof : From Theorem 3, the control law (6.66) obtained using Algorithm 6.4.2,

$$u_i = K_{1i} x_i + K_{2i} \zeta_i \tag{6.70}$$

stabilizes the system (6.7), i.e., $A_i + B_i K_{1i}$ is made Hurwitz, and guarantees output synchronization i.e., $\lim_{t\to\infty} y_i(t) - y_0(t) = 0$. Now based on Assumption 2, there exists unique nontrivial matrix $\Pi_i \in \mathbb{R}^{n_i \times p}$ that satisfies

$$(A_i + B_i K_{1i})\Pi_i + B_i K_{2i} = \Pi_i S \tag{6.71}$$

This is a Sylvester equation and the existence of the solution Π_i is guaranteed since $\sigma(S) \cap \sigma(A_i + B_i K_{1i}) \in \varnothing$ [125]. Also based on Theorem 2, an appropriate value of γ achieves the output regulation for control law (6.66), i.e.,

$$\lim_{t\to\infty} y_i(t) - y_0(t) = \lim_{t\to\infty} = C_i x_i(t) - R\zeta_0(t) = 0. \tag{6.72}$$

This is based on Theorem 1 where $\zeta_i \to \zeta_0$ is shown. Consider the state transformation $\overline{x}_i = x_i - \Pi_i \zeta_0$. The dynamics of the new state $\overline{x}_i$ under (6.70) and (6.71) is

$$\begin{aligned}\dot{\overline{x}}_i &= \dot{x}_i - \Pi_i \dot{\zeta}_0 \\ &= A_i x_i + B_i K_{1i} x_i + B_i K_{2i} \zeta_0 - \Pi_i S \zeta_0 \\ &= (A_i + B_i K_{1i}) \overline{x}_i \end{aligned} \tag{6.73}$$

this $\lim_{t\to\infty} \overline{x}_i = 0$. From Theorem 2,

$$\lim_{t\to\infty} C_i x_i(t) - R\zeta_0(t) = \lim_{t\to\infty} C_i \overline{x}_i(t) + \lim_{t\to\infty} (C_i \Pi_i - R)\zeta_0(t) = 0 \tag{6.74}$$

since $\zeta_0(t)$ is obtained from a marginally stable system (Assumption 2) this implies $C_i \Pi_i - R = 0$. Using the transformation $\Gamma_i = K_{2i} + K_{1i}\Pi_i$ in (6.71) along with (6.74) gives (6.69), this completes the proof.

6.6. SIMULATION RESULTS

In this section, we provide a detailed simulation analysis of the proposed adaptive optimal output synchronization approach. We choose the leader to be sinusoidal trajectory generator and its dynamics is given by

$$\begin{aligned}\dot{\zeta}_0 &= \begin{pmatrix} 0 & 2 \\ -2 & 0 \end{pmatrix} \zeta_0 \\ y_0 &= \begin{pmatrix} 1 & 0 \end{pmatrix} \zeta_0 \end{aligned} \tag{6.75}$$

The heterogeneous followers are given by (6.8) for $i = 1, \cdots, 4$ and their dynamics is

$$\begin{aligned} &A_1 = 0, B_1 = 10, C_1 = 1 \\ &A_2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, B_2 = \begin{pmatrix} 0 \\ 5 \end{pmatrix}, C_2 = \begin{pmatrix} 1 & 0 \end{pmatrix} \\ &A_3 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, B_3 = \begin{pmatrix} 0 \\ 2 \end{pmatrix}, C_3 = \begin{pmatrix} 1 & 0 \end{pmatrix} \\ &A_4 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix}, B_4 = \begin{pmatrix} 5 \\ 0 \\ 10 \end{pmatrix}, C_4 = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \end{aligned} \tag{6.76}$$

The underlying communication network of heterogeneous systems is given in Figure 2 The distributed observer (6.14), (6.15) is implemented for $i = 1, \cdots, 4$, The observer and adaptive gains are chosen as $c = 25, \Gamma_{si} = 15$. For the initial leader's state $\zeta_0(0) = [1 \quad 1]^T$, the error between observer and leader's state along with the Frobenius norm

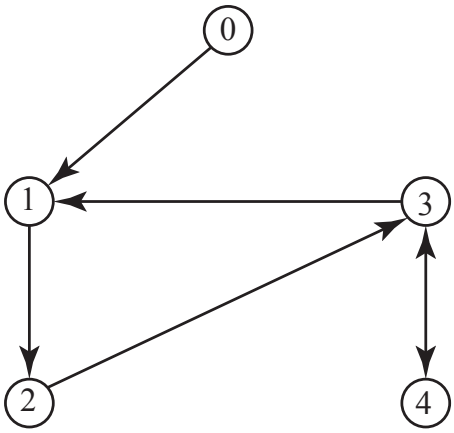


Figure 6.2: Communication graph for the heterogeneous systems.

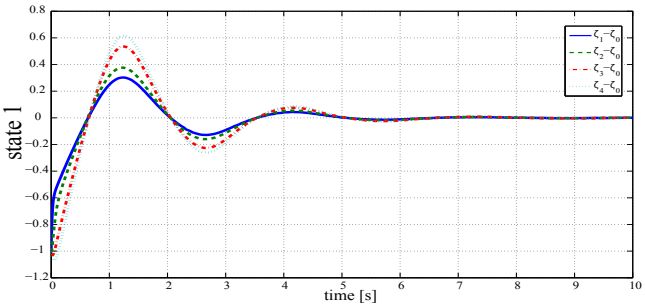


Figure 6.3: Error between adaptive observer and leader's trajectory for state 1.

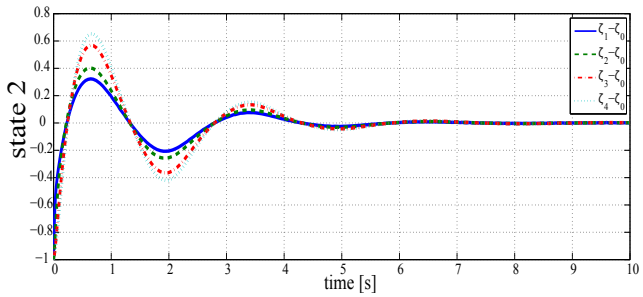


Figure 6.4: Error between adaptive observer and leader's trajectory for state 2.

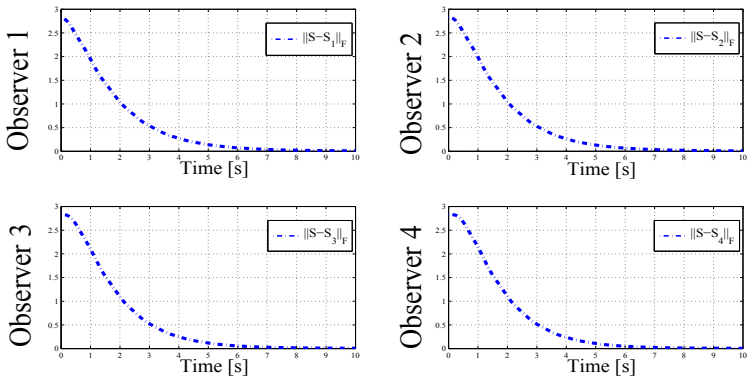


Figure 6.5: Frobenius norm $\|S-\hat{S}_i\|_F$ for the adaptive observers.

$\|S-\hat{S}_i\|_F$ is given in Figure 6.3 and Figure 6.4 and Figure 6.5, respectively. It can be seen from these figures that the introduced distributed observer converges to the leader's state.

The solution of the output regulator equation (6.8) for the given heterogeneous systems (6.76) is

$$\begin{aligned}
\Pi_1 &= \begin{pmatrix} 0 & 1 \end{pmatrix}, \Gamma_1 = \begin{pmatrix} 0 & 0.2 \end{pmatrix} \\
\Pi_2 &= \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \Gamma_2 = \begin{pmatrix} -0.8 & 0 \end{pmatrix} \\
\Pi_3 &= \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \Gamma_3 = \begin{pmatrix} -1.5 & 0 \end{pmatrix} \\
\Pi_4 &= \begin{pmatrix} 0.36 & 0.48 \\ 0.64 & -0.48 \\ 0.96 & 1.28 \end{pmatrix}, \Gamma_4 = \begin{pmatrix} -0.192 & 0.144 \end{pmatrix}
\end{aligned} \tag{6.77}$$

For the following choice of the weight matrices Q_i, R_i the resulting optimal state feedback gain using LQR method for (6.76) is

$$\begin{aligned}
&Q_1 = 100, R_1 = 1, K_{11} = -10 \\
&Q_2 = \begin{pmatrix} 100 & 0 \\ 0 & 100 \end{pmatrix}, R_2 = 1, K_{12} = \begin{pmatrix} -10 & -10.19 \end{pmatrix} \\
&Q_3 = \begin{pmatrix} 100 & 0 \\ 0 & 100 \end{pmatrix}, R_3 = 1, K_{13} = \begin{pmatrix} -9.51 & -10.46 \end{pmatrix} \\
&Q_4 = 100 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, R_4 = 1, K_{14} = \begin{pmatrix} -10 & -12.66 & -6.29 \end{pmatrix}
\end{aligned} \tag{6.78}$$

Using (6.77) and (6.78) the local optimal output regulator control (6.9) can be solved. Instead, the tracking control is obtained online by using the Algorithm 1. The convergence

of the learning controller to their optimal values given by (6.77) and (6.78) for all the agents is given in Figure 6.6. The evaluation of learned optimal tracking control along with the adaptive observer for the given multi-agent heterogeneous network is in Figure 6.7.

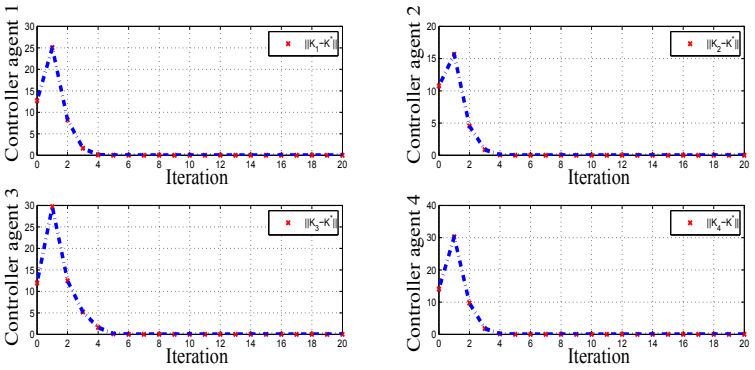


Figure 6.6: Convergence of the learning controller to their optimal values.

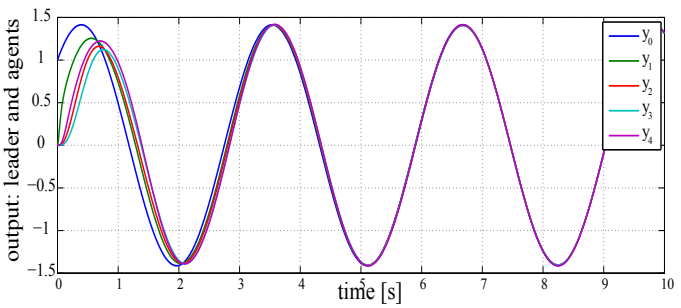


Figure 6.7: Evaluation of the learned controller along with adaptive observer for all 4 heterogeneous agents given in (6.76).

It can be seen from these results that the introduced approach implicitly solves the output regulator equations (6.8) and solves problem 1 without requiring any knowledge of either agent's or leader's dynamics.

6.7. CONCLUSIONS

A novel model-free approach is provided to design a distributed controller output tracking of a heterogeneous network. A local discounted performance function is defined for each agent which penalizes its own control effort and its tracking error. It is shown that minimizing these performance functions leads to solving AREs. It is also shown that the solutions found by solving AREs ensures synchronization provided that the discount factor is small enough. An adaptive distributed observer is designed to estimate the leader

state and reinforcement learning is used to solve the AREs without requiring any knowledge of the dynamics of the agents. A simulation example is provided to show that the proposed approach in fact implicitly solves the output regulator equation for each agent (which is a necessary and sufficient condition to achieve output synchronization).

7

CONCLUSIONS AND RECOMMENDATIONS

In this thesis, reinforcement learning methods to solve passivity-based and distributed control problems have been proposed. The developed methods have various advantages in comparison to the standard approaches, as illustrated by theoretical analysis, numerical simulations and experimental studies. In this chapter the contributions and other findings are summarized. Additionally open research issues and recommendations for possible future work are given. Finally a generic outlook on using a parameterized control law in the reinforcement learning framework is provided.

7.1. SUMMARY OF CONTRIBUTIONS AND CONCLUSIONS

In Chapter 2 the essential theoretical background on port-Hamiltonian theory and passivity-based control was provided. Important system properties, such as stability, passivity, variational symmetry, and self-adjointness are discussed. These properties form the basis for various model-based approaches, such as control-by-interconnection, passivity-based control, canonical transformation, etc. When combining PH models with adaptive and learning approaches, various port-Hamiltonian system properties, such as stability, passivity, etc. are retained. This is a key advantage and is the main motivation for augmenting PH theory with learning and adaptive control. This and additional details are given in Chapter 3.

The adaptive and learning techniques are preferred mechanisms to deal with the model, parameter uncertainties and external disturbances. They are also desired when an analytical model of the dynamical system under consideration is rather expensive to obtain. It is shown that by combining the PH theory with adaptive and learning control results in various advantages. A few of the prominent enhancements are: monotonic error convergence (for ILC), optimal control, guaranteed error bounds, etc. A detailed numerical study is performed to illustrate the effectiveness of the iterative and repetitive control in the PH framework. Simulation results show that augmenting learning

and adaptation with the PH model is an effective mechanism. Thanks to this approach various complex control objectives can be achieved. In many instances the given learning algorithm retains the PH structure, hence it inherently ensures the system properties that are introduced in Chapter 2. The existing state-of-the-art learning methods are listed in Table 3.2. However, the enumerated methods lack the capability to learn an optimal value for a parameterized control law. Although iterative feedback tuning (IFT) and evolutionary strategy IDA-PBC (ES-IDA-PBC) can be used to learn the parameters of the control law, their applicability is limited by the self-adjointness and offline learning assumptions, respectively.

Chapter 4 has introduced various learning algorithms (EBAC, AIDA-AC, CbI-AC, etc.) using the actor-critic framework. These algorithms can learn the parameterized passivity-based control law. The unknown parameter vector of the feedback controller is learned online by directly interacting with the system. In the proposed learning algorithms, the control objective can be specified via a performance measure in the form of a reward function. This solves the requirement for a global Hamiltonian, which is a key obstacle in the existing passivity based methods. Thanks to learning, model and parameter uncertainties can be readily addressed. Importantly, learning avoids computing explicit solutions of the complex partial-differential and algebraic equations present in the model-based passivity control synthesis. The parameters of the controller are then learned online. Additionally, system nonlinearities such as control input saturation can also be handled. This is achieved by setting the gradient update to zero when the control input exceeds the saturation bound. This stops the actor parameter update thus avoiding the saturation. Control saturation can also be achieved, to some extent, by formulating the reward function to penalize a high control effort.

Unlike in prominent critic-only RL methods, continuous-actions can be achieved by using the parameterized control law. Because of the simple framework, the proposed methods can be extended for regulating multi-input multi-output systems. The online learning control methods that are developed in this thesis are approximated by using polynomial, radial and Fourier basis functions. Prior state-space knowledge such as symmetry can be helpful while choosing the type of the basis functions.

Extensive numerical and experimental studies have shown that online learning is an effective mechanism to solve the nonlinear control synthesis problem. Using learning relatively complex control tasks, such as swing up and stabilization of a pendulum, can be achieved by using a single control law. This is an involved control problem if only the model-based port-Hamiltonian control approaches are used. Additionally, for the developed methods a physical meaning can be attributed to the learned control law. This is not possible in the standard actor-critic framework. The developed methods generally require a smaller number of basis functions, thus resulting in faster convergence. For example, compared to the standard actor-critic that needed more than 100 policy parameters, energy balancing actor-critic (EBAC) can achieve the same control objective with only 40 control parameters. Also it must be noted that the proposed methods can learn the control policy much faster when compared to the standard actor-critic algorithm. An improvement of 30-50% in learning speed was noticed for passivity-based actor-critic methods. It was also observed that when the policy gradient was adjusted due to saturation bounds, then the learning algorithm generally required a larger num-

ber of samples to learn the control policy. However, the learned policy was well within the desired saturation limits. The faster convergence of the RL based passivity control methods can be attributed to the available prior knowledge in the form of the PH model.

The proposed methods were used for stabilization and regulation of mechanical and electro-mechanical systems. The simulation and experimental results show a comparable learning curve. The actor and critic learning rates are generally chosen to be one order of magnitude apart. Feasible learning rates were obtained by gridding. The learning rates were fine tuned to ensure nearly zero failure during the numerical simulations. The chosen learning rates are rather conservative which often limits the speed of learning. A higher actor and critic learning rate generally leads to more aggressive learning. However, when doing so a considerable number of failures were observed.

First a set of appropriate learning rates were obtained in simulation. Then, by using these as a basis, the rates were fine tuned during the experimental study. Along with the learning rates, the magnitude of the reward function also plays a key role in the proposed methods. This is in contrast to the standard reinforcement learning framework.

The developed methods were evaluated for parameter and model uncertainties. In simulation this is done by choosing incorrect system parameters in the control policy. Extended evaluation of the proposed methods is done to check for scalability. For example, the methods are evaluated experimentally for the regulation of a manipulator arm having two degrees of freedom. It was observed that due to the symmetry of the state space some systems require a smaller number of Fourier basis functions compared to the polynomial approximation. A smaller number of parameter leads to faster learning.

Chapter 5 discussed the convergence of the developed actor-critic methods. For online learning, exploration is a fundamental component. Using Gaussian noise as an exploration term added to the control input, the parameterized control law can be reformulated as a probability density function. In the stochastic reinforcement learning framework, for the discounted reward setting the policy gradient theorem is used to obtain the gradient of the cost function in terms of action-value function and the control law. On further simplification the policy gradient was approximated by the temporal difference and the actor basis function. Finally, by assuming the actor learning rate to be an order of magnitude higher compared to the critic learning rate, the convergence to a local minima is shown by using the stochastic approximation approach. This is a generic proof and it can be used as a basis to demonstrate the convergence of any parameterized control law in the standard actor-critic setting, provided that the 7 assumptions introduced in Chapter 5 are satisfied.

Chapter 6 introduced a novel reinforcement learning algorithm to solve the optimal distributed tracking problem for a network of linear heterogeneous systems. The existing methods available in the literature are model-based. That is, they require the complete information of leader's and all the agent's dynamics. The existing state-of-the-art adaptive and robust methods for heterogeneous MAS tracking are far from optimal. In contrast the developed method does not require the system information and also it is locally optimal as it learns the tracking controller by directly interacting with the system. Detailed numerical evaluation was conducted to show the feasibility of the developed method. A few prominent applications for the heterogeneous MAS are surveillance, exploration, search and rescue, etc.

7.2. OPEN ISSUES RECOMMENDATIONS FOR FUTURE RESEARCH

In this section some of the open research issues that influence the developed parameter learning RL framework are discussed. Additionally, several fundamental issues encountered when using RL for control will also be listed.

- Although the developed methods in Chapter 4 have been only evaluated for stabilization and regulation of mechanical and electro-mechanical systems, their extension to at least some of the multi-domain complex systems should be possible. A current exception may be electrical systems as they generally require faster sampling compared to a high-inertia mechanical or electro-mechanical system. This scenario might change with the enhancements in the embedded computational capabilities.

- Continuing from the previous observation, in this thesis only the stabilization and regulation control problems are addressed for a nonlinear system in PH form (see Chapter 4). Extension of the developed RL based passivity control methods to address tracking problem would be valuable. Application of these methods for more complex systems such as nonholonomic systems, underactuated systems, etc. would be another logical extension.

- The presented methods in Chapter 4 have been evaluated for systems with up to four state variables. The scalability of the algorithm to systems with larger number of state variables might lead to the curse of dimensionality. Currently, we lack a mechanism that uses the prior system knowledge to decouple the state variables into subgroups. This separation might reduce the computational and memory requirements.

- The reward function is a key element in reinforcement learning. In standard literature, generally a simple choice for the reward function is suggested. The simplest example is to give a reward of 1 when the system is at the desired goal and 0 at other parts of the state-space. However, this simple formulation is not feasible for control as it often results in slow learning. Also complex control objectives cannot be included in this simple form. Hence for control applications the reward function needs to be formulated in a more intelligent and systematic way. Although, initial results are available in [151] a detailed analysis would be useful.

- Extending the previous point, currently no general mechanism exists that can be used to split the control objective (i.e., reward function) into a number of sub goals. This can lead to simple and easier learning objectives and it might result in faster convergence and monotonic learning. Reward-shaping is a novel technique, the inclusion of reward-shaping in RL methods for control would be a valuable extension [152, 153].

- Currently it is not possible to explicitly incorporate the existing knowledge of a safe operating region during policy parameterizations. The influence of this knowledge on the actor formulation might provide a valuable insight.

- In Chapter 4 only discrete-time learning algorithms are used. Modifying the proposed methods for continuous-time RL would be a next logical extension.
- Distributed control of heterogeneous systems has a wide applications, such as advanced surveillance, exploration, search and rescue, etc. These applications generally require multiple physical systems to interact with each other. However, the proposed method cannot be directly used as it assumes the heterogeneous agent to be a linear time-invariant system, whereas many of the robotic systems are nonlinear. Hence a natural extension is to develop a distributed model-free multi-agent heterogeneous tracking algorithm for nonlinear agents.
- In the methods developed in Chapter 6 the communication graph is assumed to be a directional time-invariant graph. However, a more practical assumption would be to consider an undirected but time-varying graph. In multi robotic networks, a bi-directional communication (i.e., an undirected graph) can be justified. This is because if agent i can send information to agent j then communication can also happen in the other direction. An extension of the proposed algorithm to address these scenarios would be a valuable addition.
- Approximate reinforcement learning methods can be used to find local optimal solutions. An alternative would be to learn a control law which satisfies the desired performance criterion but not specifically optimality. This is addressed by the satisficing theory, a heuristic decision making strategy introduced by Herbert Simon in [154]. The notion of satisficing is an upcoming control methodology. In satisficing control instead of an optimal solution, a search is done to obtain an feasible solution which satisfies the acceptable performance threshold. Using a variation of the satisficing framework in reinforcement learning might lead to faster learning [155, 156].

GLOSSARY

The glossary contains summary of symbols, notations and abbreviations used throughout the thesis.

MATH SYMBOLS

$\mathbb{R}$	set of real numbers
E	expectation
P	probability
$\frac{\partial f}{\partial x}$ or $\nabla_x f$	gradient or partial derivative of f w.r.t. x
$\dot{x}$ or $\frac{dx}{dt}$	derivative of x w.r.t. time t
σ	standard deviation
μ	variance
x^T	transpose of x
$g^\perp$	left-annihilator matrix
$\|x\|$	norm of x
k	time or iteration index
$\mathcal{N}(\mu,\sigma)$	normal distribution

PH AND PBC SYMBOLS

$x; X$	state; state space
u	control input
y	system output
J	interconnection matrix
R	dissipation matrix
F	system matrix
H	system Hamiltonian
J_d	desired interconnection matrix
R_d	desired dissipation matrix
F_d	desired system matrix
H_d	desired Hamiltonian
J_d	added interconnection component
R_d	added dissipation component
H_a	added energy component
J_c	controller interconnection matrix
R_c	controller dissipation matrix
F_c	controller system matrix
H_c	controller Hamiltonian

H_{ns}	non shapable component of Hamiltonian
H_{s}	shapable component of Hamiltonian
θ	unknown parameter vector
ϕ	basis function
$\hat{\theta}$	estimated unknown parameter vector
T	kinetic energy in a system
V	potential energy in a system
V_{d}	desired potential energy
K_{p}	proportional gain matrix
K_{d}	damping-injection matrix
Γ	update function
Σ	nonlinear operator
Σ_{v}	variational system
$(x_{\text{v}}, u_{\text{v}}, y_{\text{v}})$	variational state, input and output, respectively
Σ^*	adjoint system
(x^*, u^*, y^*)	adjoint state, input and output, respectively
ξ	parameter vector
ς	saturation function

RL-AC SYMBOLS

$x; X$	state; state space
u, U	action; action space
ρ	reward function
r	reward
π	policy
J^π	cost function for policy π
V^π	state-value function for policy π
Q^π	action-value function for policy π
A^π	advantage function for policy π
$\hat{\pi}$	approximate policy
$\hat{V}^{\hat{\pi}}$	approximate state-value function
υ	critic parameter vector
δ	temporal difference
λ	trace-decay rate
γ	discount factor
θ	actor parameter vector
α	learning rate
Δu	exploration
T_t	trial duration
T_s	sampling time
N_s	number of samples
M	martingale sequence
γ_{c}	continuous-time discount factor

DISTRIBUTED MAS SYMBOLS

$\mathscr{G}$	directed graph
$\mathscr{V}$	set of nodes
$\mathscr{E}$	set of edges
$\mathscr{A}$ or A	graph adjacency matrix
D	in-degree matrix
Π	projection operator

LIST OF ABBREVIATIONS

PH	port-Hamiltonian system
PBC	passivity-based control
PDE	partial differential equation
RL	reinforcement-learning
AC	actor-critic
EB	energy balancing
DI	damping injection
ES	energy shaping
IDA	interconnection and damping assignment
CbI	control by interconnection
ILC	iterative learning control
RC	repetitive control
IFT	iterative feedback tuning
ES	evolutionary strategy
S-AC	standard actor-critic
TD	temporal difference
MDP	Markov decision process
MAS	multi-agent system
IRL	integral reinforcement learning

References

REFERENCES

[1] K. J. Aström and R. M. Murray, *Feedback systems: an introduction for scientists and engineers* (Princeton university press, 2012).

[2] S. Skogestad and I. Postlethwaite, *Multivariable feedback control: analysis and design*, Vol. 2 (Wiley New York, 2007).

[3] S. Sastry, *Nonlinear systems: analysis, stability, and control*, Vol. 10 (Springer Science & Business Media, 2013).

[4] T. Kailath, *Linear systems*, Vol. 1 (Prentice-Hall Englewood Cliffs, NJ, 1980).

[5] R. C. Dorf and R. H. Bishop, *Modern control systems,* (1998).

[6] H. Nijmeijer and A. Van der Schaft, *Nonlinear dynamical control systems* (Springer Science & Business Media, 2013).

[7] H. Khalil, *Nonlinear Systems* (Prentice Hall, 2002).

[8] A. Schaft, *L_2-gain and Passivity Techniques in Nonlinear Control* (Springer-Verlag New York, Inc., 2000).

[9] R. Ortega, J. A. L. Perez, P. J. Nicklasson, and H. Sira-Ramirez, *Passivity-based control of Euler-Lagrange systems: mechanical, electrical and electromechanical applications* (Springer Science & Business Media, 2013).

[10] D. A. White and D. A. Sofge, *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptative Approaches* (Van Nostrand Reinhold Company, 1992).

[11] A. G. Barto, *Reinforcement learning and adaptive critic methods,* Handbook of intelligent control **469**, 491 (1992).

[12] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction* (Cambridge University Press, 1998).

[13] M. Wiering and M. van Otterlo, *Reinforcement Learning: State-of-the-art*, Vol. 12 (Springer Science & Business Media, 2012).

[14] C. Szepesvári, *Algorithms for reinforcement learning,* Synthesis Lectures on Artificial Intelligence and Machine Learning **4**, 1 (2010).

[15] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst, *Reinforcement learning and dynamic programming using function approximators* (CRC Press, 2010).

[16] F. L. Lewis and D. Vrabie, *Reinforcement learning and adaptive dynamic programming for feedback control,* IEEE Circuits and Systems Magazine **9**, 32 (2009).

[17] L. P. Kaelbling, M. L. Littman, and A. W. Moore, *Reinforcement learning: A survey,* Journal of artificial intelligence research , 237 (1996).

[18] I. Grondman, *Online Model Learning Algorithms for Actor-Critic Control*, Ph.D. thesis, TU Delft, Delft University of Technology (2015).

[19] V. Konda and J. Tsitsiklis, *On actor-critic algorithms,* SIAM Journal on Control and Optimization **42**, 1143 (2003).

[20] J. Peters and S. Schaal, *Reinforcement learning of motor skills with policy gradients,* Neural networks **21**, 682 (2008).

[21] J. Kober and J. Peters, *Reinforcement learning in robotics: A survey,* in *Reinforcement Learning* (Springer, 2012) pp. 579–610.

[22] 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* (Springer, 2006) pp. 363–372.

[23] J. Lunze, *Synchronization of heterogeneous agents,* IEEE Transactions on Automatic Control **57**, 2885 (2012).

[24] P. Wieland, R. Sepulchre, and F. Allgöwer, *An internal model principle is necessary and sufficient for linear output synchronization,* Automatica **47**, 1068 (2011).

[25] V. Duindam, A. Macchelli, and S. Stramigioli, *Modeling and control of complex physical systems* (Springer, Berlin Heidelberg, Germany, 2009).

[26] A. van der Schaft and D. Jeltsema, *Port-hamiltonian systems theory: An introductory overview,* **1**, 173 (2014).

[27] R. Ortega and M. W. Spong, *Adaptive motion control of rigid robots: A tutorial,* Automatica **25**, 877 (1989).

[28] J. Á. Acosta and A. Astolfi, *On the pdes arising in ida-pbc,* in *Proceedings of the 48th IEEE Conference on Decision and Control, held jointly with the 28th Chinese Control Conference.* (IEEE, 2009) pp. 2132–2137.

[29] V. S. Borkar, *Stochastic approximation* (Cambridge University Press, 2008).

[30] S. Bhatnagar, R. S. Sutton, M. Ghavamzadeh, and M. Lee, *Natural actor–critic algorithms,* Automatica **45**, 2471 (2009).

[31] A. van der Schaft, *Port-Hamiltonian systems: network modeling and control of nonlinear physical systems,* Advanced Dynamics and Control of Structures and Machines CISM Courses and lectures , 15 (2004).

[32] A. Schaft, *Port-Hamiltonian systems: an introductory survey,* in *Proceedings of the international congress of mathematicians* (European Mathematical Society Publishing House (EMS Ph), Madrid, Spain, 2006) pp. 1339–1365.

[33] K. Fujimoto and T. Sugie, *Canonical transformation and stabilization of generalized Hamiltonian systems,* Systems & Control Letters **42**, 217 (2001).

[34] R. Ortega, A. van der Schaft, F. Castaños, and A. Astolfi, *Control by interconnection and standard passivity-based control of port-Hamiltonian systems,* IEEE Transactions on Automatic Control **53**, 2527 (2008).

[35] J. Koopman and D. Jeltsema, *Casimir-based control beyond the dissipation obstacle,* in *4th IFAC Workshop on Lagrangian and Hamiltonian Methods for Non Linear Control* (IFAC, 2012) pp. 173–177.

[36] R. Ortega, A. J. Van Der Schaft, I. Mareels, and B. Maschke, *Putting energy back in control,* IEEE Control Systems **21**, 18 (2001).

[37] R. Ortega, A. Van Der Schaft, B. Maschke, and G. Escobar, *Interconnection and damping assignment passivity-based control of port-controlled Hamiltonian systems,* Automatica **38**, 585 (2002).

[38] R. Ortega and E. Garcia-Canseco, *Interconnection and damping assignment passivity-based control: A survey,* European Journal of Control **10**, 432 (2004).

[39] B. Jacob and H. J. Zwart, *Linear Port-Hamiltonian systems on infinite-dimensional spaces* (Springer, 2012).

[40] H. Nijmeijer and A. Van der Schaft, *Nonlinear dynamical control systems* (Springer, 1990).

[41] J. C. Willems, *Dissipative dynamical systems part I: General theory,* Archive for rational mechanics and analysis **45**, 321 (1972).

[42] K. Fujimoto and J. M. Sherpen, *Eigenstructure of nonlinear hankel operators,* in *Nonlinear control in the Year 2000* (Springer, 2000) pp. 385–397.

[43] K. Fujimoto and T. Sugie, *On adjoints of Hamiltonian systems,* in *Proc. of IFAC World Congress* (2002) pp. 1602–1607.

[44] K. Fujimoto and T. Sugie, *Iterative learning control of Hamiltonian systems: I/O based optimal control approach,* IEEE Transactions on Automatic Control **48**, 1756 (2003).

[45] J. M. Scherpen, K. Fujimoto, and W. S. Gray, *On adjoints and singular value functions for nonlinear systems,* in *Conference on information science and systems* (2000).

[46] K. Fujimoto and J. M. Scherpen, *Balanced realization and model order reduction for nonlinear systems based on singular value analysis,* SIAM Journal on Control and Optimization **48**, 4591 (2010).

[47] M. Yamakita and K. Furuta, *Iterative generation of virtual reference for a manipulator,* Robotica **9**, 71 (1991).

[48] G. Strang, *Introduction to linear algebra* (SIAM, 2003).

[49] J. E. Marsden and T. S. Ratiu, *Introduction to mechanics and symmetry: a basic exposition of classical mechanical systems*, Vol. 17 (Springer Science & Business Media, 1999).

[50] K. J. Åström and B. Wittenmark, *Adaptive control* (Courier Dover Publications, 2013).

[51] D. Dirksz and J. Scherpen, *Adaptive tracking control of fully actuated port-Hamiltonian mechanical systems,* in *IEEE International Conference on Control Applications (CCA)* (IEEE, 2010) pp. 1678–1683.

[52] I. D. Landau, R. Lozano, M. M'Saad, and A. Karimi, *Adaptive control: algorithms, analysis and applications* (Springer, 2011).

[53] P. A. Ioannou and J. Sun, *Robust adaptive control* (Courier Dover Publications, 2012).

[54] J.-J. E. Slotine and W. Li, *Applied nonlinear control* (Prentice-Hall, 1991).

[55] Z. Bien and J.-X. Xu, *Iterative learning control: analysis, design, integration and applications* (Kluwer Academic Publishers, 1998).

[56] D. A. Bristow, M. Tharayil, and A. G. Alleyne, *A survey of iterative learning control,* IEEE Control Systems **26**, 96 (2006).

[57] Y. Wang, F. Gao, and F. J. Doyle III, *Survey on iterative learning control, repetitive control, and run-to-run control,* Journal of Process Control **19**, 1589 (2009).

[58] H. Hjalmarsson, *Iterative feedback tuning—- an overview,* International journal of adaptive control and signal processing **16**, 373 (2002).

[59] R. Beard, G. Saridis, and J. Wen, *Approximate solutions to the time-invariant hamilton–jacobi–bellman equation,* Journal of Optimization theory and Applications **96**, 589 (1998).

[60] D. Vrabie and F. Lewis, *Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems,* Neural Networks **22**, 237 (2009).

[61] D. Vrabie, K. G. Vamvoudakis, and F. L. Lewis, *Optimal adaptive control and differential games by reinforcement learning principles*, Vol. 81 (IET, 2013).

[62] H.-G. Beyer and H.-P. Schwefel, *Evolution strategies–A comprehensive introduction,* Natural computing **1**, 3 (2002).

[63] D. Dirksz and J. Scherpen, *Power-based adaptive and integral control of standard mechanical systems,* in *49th IEEE Conference on Decision and Control (CDC)* (IEEE, 2010) pp. 4612–4617.

[64] K. Fujimoto and I. Koyama, *Iterative feedback tuning for Hamiltonian systems,* in *Proc. 17th IFAC World Congress* (2008) pp. 15678–15683.

[65] J. C. R. Álvarez, *Port controller Hamiltonian synthesis using evolution strategies,* in *Dynamics, Bifurcations, and Control* (Springer, 2002) pp. 159–172.

[66] Y. Sun, Q. Liu, Y. Song, and T. Shen, *Hamiltonian modelling and nonlinear disturbance attenuation control of TCSC for improving power system stability,* IEE Proceedings-Control Theory and Applications **149**, 278 (2002).

[67] D. A. Dirksz and J. M. Scherpen, *Structure preserving adaptive control of port-Hamiltonian systems,* IEEE Transactions on Automatic Control **57**, 2880 (2012).

[68] Y. Wang, G. Feng, and D. Cheng, *Simultaneous stabilization of a set of nonlinear port-controlled Hamiltonian systems,* Automatica **43**, 403 (2007).

[69] A. Wei and Y. Wang, *Adaptive control of uncertain port-controlled Hamiltonian systems subject to actuator saturation,* International Journal of Control Automation and Systems **9**, 1067 (2011).

[70] Z. Wang and P. Goldsmith, *Modified energy-balancing-based control for the tracking problem,* IET Control Theory & Applications **2**, 310 (2008).

[71] K. Fujimoto, K. Sakurama, and T. Sugie, *Trajectory tracking control of port-controlled Hamiltonian systems via generalized canonical transformations,* Automatica **39**, 2059 (2003).

[72] K. Fujimoto, *On iterative learning control of nonholonomic Hamiltonian systems,* in *Proc. 16th Symposium on Mathematical Theory of Networks and Systems (MTNS2004)* (2004).

[73] J.-J. Slotine and L. Weiping, *Adaptive manipulator control: A case study,* IEEE Transactions on Automatic Control **33**, 995 (1988).

[74] Y. Guo and D. Cheng, *Stabilization of time-varying Hamiltonian systems,* IEEE Transactions on Control Systems Technology **14**, 871 (2006).

[75] Y. Tan, W. Moase, C. Manzie, D. Nesic, and I. Mareels, *Extremum seeking from 1922 to 2010,* in *29th Chinese Control Conference (CCC)* (IEEE, 2010) pp. 14–26.

[76] K. Fujimoto and S. Satoh, *Repetitive control of Hamiltonian systems based on variational symmetry,* Systems & Control Letters **60**, 763 (2011).

[77] S. Satoh, K. Fujimoto, and S.-H. Hyon, *Gait generation for a hopping robot via iterative learning control based on variational symmetry,* in *Lagrangian and Hamiltonian Methods for Nonlinear Control 2006* (Springer, 2007) pp. 197–208.

[78] S. Satoh, K. Fujimoto, and S. Hyon, *Biped gait generation via iterative learning control including discrete state transitions,* in *Proc. 17th IFAC World Congress* (2008) pp. 1729–1734.

[79] K. Fujimoto, *Optimal control of Hamiltonian systems via iterative learning,* in *Proceedings of SICE Annual Conference* (2003) pp. 2617–2622.

[80] K. Fujimoto, T. Horiuchi, and T. Sugie, *Optimal control of Hamiltonian systems with input constraints via iterative learning,* in *Proceedings of 42nd IEEE Conference on Decision and Control*, Vol. 5 (IEEE, 2003) pp. 4387–4392.

[81] Y. Kiyasu, K. Fujimoto, and T. Sugie, *Iterative learning control of nonholonomic Hamiltonian systems: Application to a vehicle system,* in *Proc. 16th IFAC World Congress* (2005).

[82] L. Cuiyan, Z. Dongchun, and Z. Xianyi, *A survey of repetitive control,* in *IEEE/RSJ international conference on intelligent robots and systems* (2004).

[83] H. Hjalmarsson, S. Gunnarsson, and M. Gevers, *A convergent iterative restricted complexity control design scheme,* in *Proceedings of the 33rd IEEE Conference on Decision and Control*, Vol. 2 (IEEE, 1994) pp. 1735–1740.

[84] H. Hjalmarsson, M. Gevers, S. Gunnarsson, and O. Lequin, *Iterative feedback tuning: theory and applications,* IEEE Control Systems **18**, 26 (1998).

[85] K. J. Åström, J. Aracil, and F. Gordillo, *A family of smooth controllers for swinging up a pendulum,* Automatica **44**, 1841 (2008).

[86] A. E. Eiben and J. E. Smith, *Introduction to evolutionary computing* (springer, 2003).

[87] R. W. Longman and Y.-C. Huang, *The phenomenon of apparent convergence followed by divergence in learning and repetitive control,* Intelligent Automation & Soft Computing **8**, 107 (2002).

[88] H.-P. Wen, M. Q. Phan, and R. W. Longman, *Bridging learning control and repetitive control using basis functions,* in *Proceedings of the AAS/AIAA Space Flight Mechanics Meeting 1998* (AIAA, 1998).

[89] J. A. Frueh, *Iterative learning control with basis functions*, Ph.D. thesis, Princeton University (2000).

[90] C. Secchi, S. Stramigioli, and C. Fantuzzi, *Control of Interactive Robotic Interfaces: a port-Hamiltonian approach* (Springer, 2007).

[91] I. Grondman, M. Vaandrager, L. Busoniu, R. Babuska, and E. Schuitema, *Efficient Model Learning Methods for Actor–Critic Control,* IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **42**, 591 (2012).

[92] R. S. Sutton, A. G. Barto, and R. J. Williams, *Reinforcement learning is direct adaptive optimal control,* IEEE Control Systems **12**, 19 (1992).

[93] J. Peters and S. Schaal, *Policy gradient methods for robotics,* in *IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2006) pp. 2219–2225.

[94] M. Deisenroth and C. E. Rasmussen, *Pilco: A model-based and data-efficient approach to policy search,* in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (2011) pp. 465–472.

[95] Y. W. A. Wei, *Stabilization and H^∞ control of nonlinear port-controlled Hamiltonian systems subject to actuator saturation,* Automatica **46**, 2008 (2010).

[96] G. Escobar, R. Ortega, and H. Sira-Ramirez, *Output-feedback global stabilization of a nonlinear benchmark system using a saturated passivity-based controller,* IEEE Transactions on Control Systems Technology **7**, 289 (1999).

[97] A. Macchelli, *Port Hamiltonian Systems: A unified approach for modeling and control finite and infinite dimensional physical systems*, Ph.D. thesis, University of Bologna (2002).

[98] A. Macchelli, C. Melchiorri, C. Secchi, and C. Fantuzzi, *A variable structure approach to energy shaping,* in *Proceedings of the European Control Conference* (2003).

[99] W. Sun, Z. Lin, and Y. Wang, *Global asymptotic and finite-gain L_2 stabilization of port-controlled Hamiltonian systems subject to actuator saturation,* in *Proceedings of the American Control Conference* (St. Louis, 2009).

[100] K. Doya, *Reinforcement learning in continuous time and space,* Neural computation **12**, 219 (2000).

[101] I. Grondman, L. Buşoniu, and R. Babuška, *Model learning actor-critic algorithms: Performance evaluation in a motion control task,* in *Proceedings IEEE Conference on Decision and Control* ((Maui, USA, 2012) pp. 5272–5277.

[102] A. Barto, R. Sutton, and C. Anderson, *Neuronlike adaptive elements that can solve difficult learning control problems,* IEEE Transactions on Systems, Man, and Cybernetics **13**, 835 (1983).

[103] I. Grondman, L. Busoniu, G. A. Lopes, and R. Babuska, *A Survey of Actor-Critic Reinforcement Learning: Standard and Natural Policy Gradients,* IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews **42**, 1291 (2012).

[104] R. Sutton, D. McAllester, S. Singh, and Y. Mansour, *Policy gradient methods for reinforcement learning with function approximation,* Advances in Neural Information Processing Systems **12**, 1057 (2000).

[105] J. Asmuth, L. Li, M. Littman, A. Nouri, and D. Wingate, *A Bayesian sampling approach to exploration in reinforcement learning,* in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (AUAI Press, 2009) pp. 19–26.

[106] G. Konidaris, S. Osentoski, and P. Thomas, *Value function approximation in reinforcement learning using the Fourier basis,* (2008).

[107] R. Hafner and M. Riedmiller, *Reinforcement learning in feedback control,* Machine learning **84**, 137 (2011).

[108] A. Prashanth L and M. Ghavamzadeh, *Actor-critic algorithms for risk-sensitive reinforcement learning,* arXiv preprint arXiv:1403.6530 (2014).

[109] H. Robbins and S. Monro, *A stochastic approximation method,* The annals of mathematical statistics , 400 (1951).

[110] J. Kiefer, J. Wolfowitz, *et al., Stochastic estimation of the maximum of a regression function,* The Annals of Mathematical Statistics **23**, 462 (1952).

[111] J. Peters, *Policy gradient methods for control applications,* .

[112] V. S. Borkar and S. P. Meyn, *The ode method for convergence of stochastic approximation and reinforcement learning,* SIAM Journal on Control and Optimization **38**, 447 (2000).

[113] H. J. Kushner and G. Yin, *Stochastic approximation and recursive algorithms and applications*, Vol. 35 (Springer Science & Business Media, 2003).

[114] S. Bhatnagar, H. Prasad, and L. Prashanth, *Stochastic approximation algorithms,* in *Stochastic Recursive Algorithms for Optimization* (Springer, 2013) pp. 17–28.

[115] P. A. Griffiths, J. W. Morgan, and J. W. Morgan, *Rational homotopy theory and differential forms* (Springer, 1981).

[116] A. Jadbabaie, J. Lin, and A. S. Morse, *Coordination of groups of mobile autonomous agents using nearest neighbor rules,* IEEE Transactions on Automatic Control **48**, 988 (2003).

[117] F. L. Lewis, H. Zhang, K. Hengster-Movric, and A. Das, *Cooperative control of multi-agent systems: optimal and adaptive design approaches* (Springer Science & Business Media, 2013).

[118] R. Olfati-Saber and R. M. Murray, *Consensus problems in networks of agents with switching topology and time-delays,* IEEE Transactions on Automatic Control **49**, 1520 (2004).

[119] W. Ren, R. W. Beard, and E. M. Atkins, *Information consensus in multivehicle cooperative control,* IEEE Control Systems **27**, 71 (2007).

[120] W. Ren and R. W. Beard, *Distributed consensus in multi-vehicle cooperative control* (Springer, 2008).

[121] W. Yu, P. DeLellis, G. Chen, M. di Bernardo, and J. Kurths, *Distributed adaptive control of synchronization in complex networks,* IEEE Transactions on Automatic Control **57**, 2153 (2012).

[122] J. Huang and Z. Chen, *A general framework for tackling the output regulation problem,* IEEE Transactions on Automatic Control **49**, 2203 (2004).

[123] C. De Persis and B. Jayawardhana, *On the internal model principle in the coordination of nonlinear systems,* IEEE Transactions on Control of Network Systems **1**, 272 (2014).

[124] T. Yang, A. Saberi, A. A. Stoorvogel, and H. F. Grip, *Output synchronization for heterogeneous networks of introspective right-invertible agents,* International journal of robust and nonlinear control **24**, 1821 (2014).

[125] Y. Hong, X. Wang, and Z.-P. Jiang, *Distributed output regulation of leader–follower multi-agent systems,* International Journal of Robust and Nonlinear Control **23**, 48 (2013).

[126] A. Das and F. L. Lewis, *Distributed adaptive control for synchronization of unknown nonlinear networked systems,* Automatica **46**, 2014 (2010).

[127] Y. Su and J. Huang, *Cooperative output regulation of linear multi-agent systems,* IEEE Transactions on Automatic Control **57**, 1062 (2012).

[128] C. Huang and X. Ye, *Cooperative output regulation of heterogeneous multi-agent systems: An h_∞ criterion,* IEEE Transactions on Automatic Control **59**, 267 (2014).

[129] E. Peymani, H. F. Grip, A. Saberi, X. Wang, and T. I. Fossen, *h_∞ almost output synchronization for heterogeneous networks of introspective agents under external disturbances,* Automatica **50**, 1026 (2014).

[130] Y. Su, Y. Hong, and J. Huang, *A general result on the robust cooperative output regulation for linear uncertain multi-agent systems,* IEEE Transactions on Automatic Control **58**, 1275 (2013).

[131] J. Xiang, W. Wei, and Y. Li, *Synchronized output regulation of linear networked systems,* IEEE Transactions on Automatic Control **54**, 1336 (2009).

[132] L. Liu and J. Huang, *Global robust output regulation of lower triangular systems with unknown control direction,* Automatica **44**, 1278 (2008).

[133] L. Yu and J. Wang, *Robust cooperative control for multi-agent systems via distributed output regulation,* Systems & Control Letters **62**, 1049 (2013).

[134] X. Wang, Y. Hong, J. Huang, and Z.-P. Jiang, *A distributed control approach to a robust output regulation problem for multi-agent linear systems,* IEEE Transactions on Automatic Control **55**, 2891 (2010).

[135] A. Isidori, L. Marconi, and G. Casadei, *Robust output synchronization of a network of heterogeneous nonlinear agents via nonlinear regulation theory,* IEEE Transactions on Automatic Control **59**, 2680 (2014).

[136] Z. Ding, *Consensus output regulation of a class of heterogeneous nonlinear systems,* IEEE Transactions on Automatic Control **58**, 2648 (2013).

[137] Z. Ding, *Adaptive consensus output regulation of a class of nonlinear systems with unknown high-frequency gain,* Automatica **51**, 348 (2015).

[138] H. Zhang, F. L. Lewis, and Z. Qu, *Lyapunov, adaptive, and optimal design techniques for cooperative systems on directed communication graphs,* IEEE Transactions on Industrial Electronics **59**, 3026 (2012).

[139] H. Zhang, F. L. Lewis, and A. Das, *Optimal design for synchronization of cooperative systems: state feedback, observer and output feedback,* IEEE Transactions on Automatic Control **56**, 1948 (2011).

[140] K. G. Vamvoudakis, F. L. Lewis, and G. R. Hudas, *Multi-agent differential graphical games: Online adaptive learning solution for synchronization with optimality,* Automatica **48**, 1598 (2012).

[141] Y. Cao and W. Ren, *Optimal linear-consensus algorithms: an lqr perspective,* IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **40**, 819 (2010).

[142] H. Zhang, T. Feng, G.-H. Yang, and H. Liang, *Distributed cooperative optimal control for multiagent systems on directed graphs: An inverse optimal approach,* IEEE Transactions on Cybernetics **PP**, 1 (2014).

[143] R. Wheeler Jr and K. Narendra, *Decentralized learning in finite markov chains,* IEEE Transactions on Automatic Control **31**, 519 (1986).

[144] P. Vrancx, K. Verbeeck, and A. Nowé, *Decentralized learning in markov games,* IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **38**, 976 (2008).

[145] H. Lakshmanan and D. P. de Farias, *Decentralized approximate dynamic programming for dynamic networks of agents,* in *American Control Conference* (IEEE, 2006) pp. 1648–1653.

[146] H. S. Chang, *Decentralized learning in finite markov chains: revisited,* IEEE Transactions on Automatic Control **54**, 1648 (2009).

[147] L. Busoniu, R. Babuska, and B. De Schutter, *A comprehensive survey of multiagent reinforcement learning,* IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews **38**, 156 (2008).

[148] J. W. Brewer, *Kronecker products and matrix calculus in system theory,* IEEE Transactions on Circuits and Systems **25**, 772 (1978).

[149] 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**, 3051 (2014).

[150] H. Modares, F. L. Lewis, and Z.-P. Jiang, h_∞ *tracking control of completely-unknown continuous-time systems,* IEEE Transactions on Neural Networks and Learning Systems (2015).

[151] J.-M. Engel and R. Babuška, *On-line reinforcement learning for nonlinear motion control: Quadratic and non-quadratic reward functions,* in *World Congress*, Vol. 19 (2014) pp. 7043–7048.

[152] A. Y. Ng, D. Harada, and S. Russell, *Policy invariance under reward transformations: Theory and application to reward shaping,* in *ICML*, Vol. 99 (1999) pp. 278–287.

[153] A. Harutyunyan, T. Brys, P. Vrancx, and A. Nowé, *Multi-scale reward shaping via an off-policy ensemble,* in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems* (International Foundation for Autonomous Agents and Multiagent Systems, 2015) pp. 1641–1642.

[154] H. A. Simon, *Rational choice and the structure of the environment.* Psychological review **63**, 129 (1956).

[155] M. Goodrich, W. C. Stirling, R. L. Frost, *et al., A theory of satisficing decisions and control,* IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans **28**, 763 (1998).

[156] J. W. Curtis and R. W. Beard, *Satisficing: A new approach to constructive nonlinear control,* IEEE Transactions on Automatic Control **49**, 1090 (2004).

LIST OF PUBLICATIONS

JOURNAL PAPERS

S. P. Nageshrao, G. A. D. Lopes, D. Jeltsema and R. Babuška, Port-Hamiltonian Systems in Adaptive and Learning Control: A Survey, To appear in *IEEE Transaction on Automatic Control*, 2015.

S. P. Nageshrao, G. A. D. Lopes, D. Jeltsema and R. Babuška, Passivity-Based Reinforcement Learning Control of a 2-DOF Manipulator Arm, *Mechatronics*, 24(8): 1001 - 1007, 2014.

H. Modares, S. P. Nageshrao, G. A. D. Lopes, R. Babuška, and F. L. Lewis, Optimal Model-free Output Synchronization of Heterogeneous Systems Using Off-policy Reinforcement Learning, Under Review, 2015.

G. A. D. Lopes, E. Najafi, S. P. Nageshrao, and R. Babuška, Learning Complex Behaviors via Sequential Composition and Passivity-based Control, Accepted for publication by Springer, 2015.

O. Sprangers, R. Babuška, S. P. Nageshrao, G. A. D. Lopes. Reinforcement learning for port-Hamiltonian systems, *IEEE Transactions on Cybernetics*, 45(5):1003-1013, 2014.

CONFERENCE PAPERS

S. P. Nageshrao, G. A. D. Lopes, D. Jeltsema and R. Babuška, Interconnection and Damping Assignment Control via Reinforcement Learning. Proc. of IFAC World Congress, 19(1):1760-1765, 2014.

S. P. Nageshrao, G. A. D. Lopes, D. Jeltsema and R. Babuška, Control by interconnection of a manipulator arm using reinforcement learning. Accepted for publication in IEEE Multi-Systems Conference, 2015.

E. Najafi, G. A. D. Lopes, S. P. Nageshrao, and R. Babuška, Rapid Learning in Sequential Composition Control, The 53rd IEEE Conference on Decision and Control, 2014.

SUMMARY

Over the last couple of decades the demand for high precision and enhanced performance of physical systems has been steadily increasing. This demand often results in miniaturization and complex design, thus increasing the need for complex nonlinear control methods. Some of the state of the art nonlinear methods are stymied by the requirement of full state information, model and parameter uncertainties, mathematical complexity, etc. For many scenarios it is nearly impossible to consider all the uncertainties during the design of a feedback controller. Additionally, while designing a model-based nonlinear control there is no standard mechanism to incorporate performance measures. Some of the mentioned issues can be addressed by using online learning.

Animals and humans have the ability to share, explore, act or respond, memorize the outcome and repeat the task to achieve a better outcome when they encounter the same or a similar scenario. This is called learning from interaction. One instance of this approach is reinforcement learning (RL). However, RL methods are hindered by the curse of dimensionality, non-interpretability and non-monotonic convergence of the learning algorithms. This can be attributed to the intrinsic characteristics of RL, as it is a model-free approach and hence no standard mechanism exists to incorporate à priori model information.

In this thesis, learning methods are proposed which explicitly use the available system knowledge. This can be seen as a new class of approaches that bridge model-based and model-free methods. These methods can address some of the hurdles mentioned earlier. For example, i) a prior system information can speed up the learning, ii) new control objectives can be achieved which otherwise would be extremely difficult to attain using only model-based methods, iii) physical meaning can be attributed to the learned controller.

The developed approach is as follows: the model of the given physical system is represented in the port-Hamiltonian (PH) form. For the system dynamics in PH form a passivity-based control (PBC) law is formulated, which often requires the solution to a set of partial differential equations (PDEs). Instead of finding an analytical solution, the PBC control law is parameterized using an unknown parameter vector. Then, by using a variation of the standard actor-critic learning algorithm, the unknown parameters can be learned online. Using the principles of stochastic approximation theory, a proof of convergence for the developed method is shown.

The proposed methods are evaluated for the stabilization and regulation of mechanical and electro-mechanical systems. The simulation and experimental results show comparable learning curves.

In the final part of the thesis a novel integral reinforcement learning approach is developed to solve for the optimal output tracking control problem for a set of linear heterogeneous multi-agent systems. Unlike existing methods, this approach does not need to solve either the output regulator equation or requires a p-copy of the leader's dynam-

ics in the agent's control law. A detailed numerical evaluation has been conducted to show the feasibility of the developed method.

SAMENVATTING

Gedurende de laatste tientallen jaren is de vraag naar hogere precisie en verbeterde prestaties van fysieke systemen gestaag toegenomen. Deze vraag resulteert vaak in miniturisatie en in complexe ontwerpen, waardoor de noodzaak van het gebruik van complexe niet-lineaire regelmethoden toeneemt. Sommige state-of-the-art niet lineaire methoden zijn onder andere beperkt door hun afhankelijkheid van volledige toestand informatie, onzekerheden in het model en de model parameters en wiskundige complexiteit. In veel gevallen is het bijna onmogelijk om rekening te houden met alle onzekerheden tijdens het ontwerp van de regelaar. Bovendien is er, voor het ontwerpen van een op een model gebaseerde niet lineaire regelaar, geen standaard methodiek om rekening te houden met de prestatiemaatstaven. Sommige van de genoemde problemen kunnen verholpen worden door het gebruik van online learning.

Dieren en mensen hebben het vermogen om te delen, actie te ondernemen of te reageren, de uitkomst van een taak te onthouden en de taak te herhalen om een beter resultaat te behalen als ze de zelfde of een vergelijkbare situatie tegenkomen. Dit heet learning from interaction (leren van interactie). Een voorbeeld van deze aanpak is reinforcement learning (RL). Echter, RL methoden worden gehinderd door de vloek van de dimensionaliteit, de niet interpreteerbaarheid en de niet monotone convergentie van de leer algoritmen. Deze problemen kunnen worden toegeschreven aan de intrinsieke kenmerken van RL, aangezien het een modelvrije aanpak is waardoor er geen standaard mechanisme bestaat om voorkennis te integreren.

In deze thesis worden leermethoden voorgesteld die expliciet de beschikbare systeemkennis gebruiken. Dit kan worden gezien als een nieuwe klasse van methoden die een brug vormt tussen model-gebaseerde en model-vrije methoden. Met deze methoden kunnen sommige van de eerdergenoemde moeilijkheden overkomen worden. Bijvoorbeeld, i) voorkennis over het systeem kan gebruikt worden om het leren te versnellen, ii) nieuwe regel doelstellingen kunnen worden verwezenlijkt die anders - met alleen model-gebaseerde methoden - bijna onhaalbaar zouden zijn, iii) er kan een fysieke betekenis worden toegeschreven aan de geleerde regelaar.

De ontwikkelde methode werkt als volgt: het model van het gegeven fysieke systeem wordt gerepresenteerd in de port-Hamiltonian (PH) vorm. Voor de systeem dynamica in PH vorm wordt een passivity-based control (PBC) wet opgesteld, waarvoor vaak de oplossing van een stelsel van partiële differentiaalvergelijkingen nodig is. Inplaats van een analytische oplossing te vinden wordt de PBC regelwet geparametriseerd met behulp van een onbekende parameter vector. Daarna kunnen, met behulp van een variatie op het standaard actor-critic algoritme, de onbekende parameters online worden geleerd. Met behulp van de principes van stochastische benaderings-theorie wordt het bewijs gegeven voor de ontwikkelde methode.

De voorgestelde methoden zijn geevalueerd voor de stabilizatie en regulatie van mechanische en electromechanische systemen.

In het laatste deel van deze thesis wordt een nieuwe integrale reinforcement learning methode ontwikkelt voor het oplossen van het optimale output tracking regel probleem voor een set van lineaire heterogene multi-agent systemen. In tegenstelling tot bestaande methoden vereist deze aanpak niet het oplossen van de output regulator equation of het gebruiken van een p-copy van de dynamica van de leider in de regelwet van de agent. Een gedetailleerde numerieke evaluatie is uitgevoerd om de haalbaarheid van de ontwikkelde methode aan te tonen.

About the author

Subramanya Prasad Nageshrao was born in Mandya, India on 24th March 1984. He obtained his Bachelor of Engineering (B.E.) degree in electronics and communication from Visvesvaraya Technological University, Belgaum, India, in 2005. Following his graduation he worked as software engineer for Bosch Ltd., Bangalore, India. In 2009 he started his postgraduate studies in mechatronics at Technische Universität Hamburg-Harburg, Hamburg, Germany, he obtained his M.Sc degree in late 2011.

During masters he worked as a research assistant at Technische Universität Hamburg-Harburg, Technische Universität München, and Helmut Schmidt Universität, Hamburg. At TUHH, for his masters, he received financial assistance from Freien und Hansestadt Hamburg (FHH) and Deutscher Akademischer Austauschdienst (DAAD) foundations. His MSc thesis was supervised by Prof. dr. Herbert Werner.

In March 2012 he started working towards his Ph.D. degree on online learning algorithms for control at the Delft Center for Systems and Control, Delft University of Technology, Delft, under the supervision of Dr. Gabriel Lopes and Prof. dr. ir. Robert Babuška. During the course of his PhD he obtained post-graduate certificate from the Dutch Institute of Systems and Control (DISC). At the beginning of 2015, he visited University of Texas at Arlington for four months as a research scholar, where he conducted a part of his research under the supervision of Prof. dr. Frank Lewis.

During the course of his PhD he has assisted in various courses offered in systems and control masters program. He has also supervised several BSc and MSc students. His current research interests include multi-agent systems, distributed control, consensus control, machine learning and nonlinear control, where he is actively combining reinforcement-learning techniques with passivity-based control.