DISCRETE-TIME CONCURRENT LEARNING FOR SYSTEM IDENTIFICATION AND APPLICATIONS: LEVERAGING MEMORY USAGE FOR GOOD LEARNING

Dissertation

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Doctor of Philosophy in Engineering

By

Ouboti Seydou Eyanaa Djaneye-Boundjou

UNIVERSITY OF DAYTON

Dayton, Ohio

December, 2017

DISCRETE-TIME CONCURRENT LEARNING FOR SYSTEM IDENTIFICATION

AND APPLICATIONS: LEVERAGING MEMORY USAGE FOR GOOD LEARNING

Name: Djaneye-Boundjou, Ouboti Seydou Eyanaa

APPROVED BY:

ABSTRACT

DISCRETE-TIME CONCURRENT LEARNING FOR SYSTEM IDENTIFICATION AND APPLICATIONS: LEVERAGING MEMORY USAGE FOR GOOD LEARNING

Name: Djaneye-Boundjou, Ouboti Seydou Eyanaa
University of Dayton

Advisor: Dr. Raúl Ordóñez

Literature on system identification reveals that persistently exiting inputs are needed in order to achieve good parameter identification when using standard learning techniques such as Gradient Descent and/or Least Squares for function approximation. However, realizing persistency of excitation in itself is quite demanding, especially in the context of on-line approximation and adaptive control. Much recently, Concurrent Learning (CL), through its utilization of memory (and, in that regard, quite similarly to human learning), has been shown to be able to yield good learning without the need to resort to persistency of excitation. For all intents and purposes, we refer to "good learning" throughout this work as the ability to reconstruct the function(s) being approximated well when using the estimated parameters.

The continuous-time (CT) domain literature on CL has seen the larger share of researches. For our part, we have focused on the discrete-time (DT) domain. Tough many systems can be modeled as CT systems, usually, controlling such systems, especially real-time (or, rather close to real-time), is done via the use of digital computers and/or micro-controllers, therefore making DT framework studies compelling.

We have shown that, similarly to the CT domain, granted a less restrictive CL condition compared to that of persistency of excitation is verified, analogous CL results to that obtained in the CT domain can also be achieved in the DT domain. Before incorporating and making use of the concept of concurrent learning in our studies, we thoroughly study the Gradient Descent and Least Squares techniques for function approximation and system identification of a dimensionally complex uncertainty, which, to the best our knowledge, is yet to be done in literature. Our main contributions are however the derivations of a DT Normalized Gradient (DTNG) based CL algorithm as well as a DT Normalized Recursive Least Squared (DTNRLS) based CL algorithm for approximation of both DT structured and DT unstructured uncertainties, while showing analytically that our devised algorithms guarantee good parameter identification if the aforesaid CL condition is met.

Numerical simulations are provided to show how well the developed CL algorithms leverage memory usage to achieve good learning. The algorithms are also made use of in two applications: the discrete-time indirect adaptive control of a class of discrete-time single state plant bearing parametric or structured uncertainties and the system identification of a robot.

To my parents and siblings.

ACKNOWLEDGMENTS

First and foremost, I am thankful to have lived up to now to see this dissertation through.

I am also thankful to the members of my committee, Dr. Keigo Hirakawa, Dr. Asari Vijayan, and Dr. Paul Eloe for agreeing to serve as members of my committee as well as the help and support they have given me. I want to personally thank Dr. Raúl Ordóñez for being my advisor, for his exquisite attention to detail, and for guiding me throughout not only my graduate studies but, also, my doctoral studies and this research.

I wish to express my sincere gratitude to Brother Maximin Magnan, SM, and Dr. Amy Anderson without whom I would in all probability not be at this university in the first place.

Last but not least, I want to thank my family (my father Gbandi Djaneye-Boundjou, my mother Akoua Tatcho, who have always been inspirational for me, my siblings Bilaal Djaneye-Boundjou and Jamila Djaneye-Boundjou) and my friends for their love, support, and encouragements.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

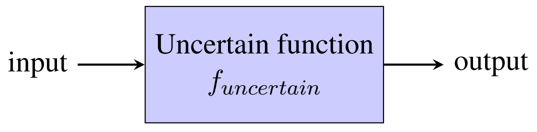
INTRODUCTION

1.1 System Identification

System identification involves making an inference about an unknown or uncertain function based on input and/or output data [1]. It has been widely used in areas such as machine learning and control systems for learning, prediction, adaptation, and various other tasks. Adequate system identification involves generation and collection of input and output data as well as performing good function approximation.

Approximating an unknown function usually takes on two forms. There are:

- The structured uncertainty approximation case (SUAC) or grey-box case;
- The unstructured uncertainty approximation case (UUAC) or black-box case.



In both cases, the approximator is made of a set of parameter estimates, updated **on-line** via an adaptation rule or law so as to provide a better estimation of the unknown (true) parameters, and a regressor, consisting of computable signals. The emphasis here is put on the term **on-line** to differentiate it from **off-line** function approximation. While only one input data point and/or one output data point are available at each instance of time when performing on-line function approximation, off-line function approximation make use of all available input and output data points on the set where approximation is being performed.

Because a structured uncertainty is parameterized by unknown (true) parameters and a known, measurable and/or computable regressor, it is assumed that an approximator designed to estimate a structured uncertainty can closely match that uncertainty so long the true parameters are available. When dealing with a black box case, in addition to parametric uncertainty there also is functional uncertainty. As a result, universal approximators, such as neural networks and fuzzy systems, which allow for construction of a regressor and estimation of the uncertainty (with a relatively small error provided there is enough structure in the regressor), are made use of.

1.2 Need for Discrete-Time Studies

A vast majority of systems can be modeled as continuous-time (CT) systems; and, understandably so, CT systems are well-studied in control literature [2, 3, 4, 5]. However, with the high-tech world that we live in and the demands of real-time performance, CT control algorithms are usually executed by means of digital computers. Derived CT controllers have to be implemented on sampled-data systems that combine a CT system and a discrete-time (DT) controller. As reported in [6, 7], three approaches usually arise.

1. The CT controller can be discretized to obtain an equivalently good DT controller provided the sampling rate is sufficiently large. However, large delays could still be potentially introduced.

2. A DT model of the CT system can be obtain via discretization and the CT controller is recovered by converting a DT controller, developed for controlling the realized DT model, back to the CT domain. It may however not always be possible to get a good DT model of a CT system.

3. Lastly, taking in consideration exact models of the Analog to Digital (AD) and Digital to Analog (DA) converters and based on a sample-data model of the original CT system, a DT controller can be directly implemented. This approach is inherently challenging, since it makes no approximations.

Bearing in mind the importance of DT designs of Case 2 (above) in particular, and, maybe, to a certain extent Case 3, the study and design of DT systems and DT controllers are as crucial as that of CT systems and CT controllers.

Moreover, of particular importance, one of our research interests (and, as a side note, what led to further investigation of function approximation, as we do in the present work) is that of the discrete in nature Particle Swarm Optimization (PSO) algorithm. In the past, an adaptive particle swarm optimization (APSO) algorithm, based on the *inertia factor* PSO algorithm and for which function approximation was employed, was developed [8, 9]. The term *adaptive* here is employed to underline the use of adaptive control in the design of the optimizer as opposed to a self-adjusting optimizer. Bettering the performance of the developed adaptive swarm optimizer, i.e., that in [8, 9], has prompted further analysis of the many aspects that constitute the aforementioned optimizer, with one those aspects being function approximation.

1.3 Persistency of Excitation Condition for Good Learning

Adaptive control, employed when dealing with uncertainties within a plant's system, relies on acquired *a posteriori* knowledge. Through collection of information, it is usually possible to put in place a learning or adaptation mechanism on-line [10] and, given certain conditions, design a control law that can achieve desired system performance even in the presence of the aforesaid uncertainties. The adaptation mechanism performs function approximation via identification. The resulting inferred information is then used to compute the adaptive controller. Two different adaptive control schemes, indirect adaptive control and direct adaptive control, emerge from the way the control action is computed. An indirect adaptive control (IAC) scheme is concerned with, first, approximating the system's uncertainties before combining their estimated values in the definition of the adaptive controller. Conversely, the adaptation mechanism of a direct adaptive control (DAC) scheme directly approximates the adaptive controller.

Some of the standard and widely used learning methods in literature are the Gradient Descent and Recursive Least Squares algorithms, which, when used for standalone approximation, do help minimize the approximation error (defined as the difference between the approximation and the function being approximated). However, they cannot be shown to theoretically guarantee convergence of parameter estimates to their true values or a neighborhood around the aforesaid true values unless the regressor used for approximation is persistently exciting (PE) [2, 3, 4]. Similarly, standard adaptive control, i.e., adaptive control using standard learning methods, can guarantee convergence of the tracking error (defined as the difference between the output to control and its desired value) to the origin or a neighborhood of the origin, but cannot analytically guarantee convergence of the parameter estimates to their true parameters or a neighborhood of their true parameters without the PE condition of the regressor being satisfied [2, 3, 4].

The PE condition can however be hard to achieve, especially when performing closed-loop control. A formal definition of a bounded persistently exciting regressor is given by (B.2) in Appendix B. The PE condition essentially requires complete span of the considered space over which an approximation is made for all predefined time. As an alternative, researchers have studied conditions under which PE conditions on the reference inputs to a system can translate to its states, outputs, and associated regressor(s). In [11], working in the CT framework, the authors show that if the reference input to a plant contains as many spectral lines as the number of unknown (true) parameters to be estimates, then the plant states are PE. For DT linear systems and DT linear switched systems, for instance, respectively, *output reachability* of the system [12] as well as *realization theory* and knowledge of *Markov parameters* [13] are sufficient conditions for realizing persistency of excitation of the regressor granted the reference inputs are PE. However, requiring reference inputs to be PE is equally as restrictive and not easily verifiable on-line.

1.4 Learning Concurrently: Leveraging Memory for Good Learning

Concurrent Learning (CL) is a recently introduced method for CT uncertainty estimation [14], whereby current and stored, past data are used in conjunction to devise an adaptation law. CL is essentially a memory based technique (hence, sharing similarities with human learning) that is capable of helping achieve better learning without necessitating persistency of excitation. In fact, it has been proved in the CT domain that CL guarantees global exponential convergence of the parameter error, when used for stand-alone identification of structured uncertainties, and global exponential convergence of both the tracking error and the parameter error, when used for developing adaptive control of a class of CT linear system bearing an underlying structured uncertainty, without requiring persistency of excitation [15, 16, 14, 17, 18]. Girish Chowdhary, who developed CL, also shows that in the presence of unstructured uncertainties, instead of exponential convergence results, one can get uniform, ultimate boundedness results of the tracking and parameter errors. It needs to be added that though not requiring the PE condition, the CL condition for improved parameter identification is that the memory containing stored data is made of the same number of linearly independent elements as there are (row) dimensions in the regressor used for approximation.

1.5 Research Motivation

System identification and, more precisely, function approximation, as noted before, have many usages in science, engineering, and mathematics. There are a good amount of publications on CL in the CT framework [15, 16, 14, 17, 19, 18, 20, 21, 22, 23, 24, 25, 26, 27] while, to the best of our knowledge, CL studies in DT framework were inexistent at the time. Because, as mentioned above, DT studies and designs are important, the ambition in this research is to study and apply the CL concept for function approximation in DT settings.

In summary:

- The present research aims to formally study system identification and how CL fairs when used for parameter identification, be it for the SUAC or the UUAC;
- Because CL used in CT framework has been fairly studied and mostly because our interest lies largely on DT systems, the focus here will mostly be on applying CL in the DT framework;
- Our developments on system identification in DT settings using CL can then be applied to control systems related problems and/or any problem(s) requiring function approximation.

1.6 Contributions

Mostly due to the fact that function approximation is looked at more in depth here because of our initial research interests, the present work has been mainly geared towards devising CL motivated algorithms in the DT framework and extending CL results of the CT framework to the DT framework. We have developed a normalized gradient based discrete-time CL algorithm, which we have used for standalone parameter identification tasks [28] as well as for uncertainty approximation within a DT adaptive control loop [29]. Literature also suggests that the Recursive Least Squares (RLS) algorithm can potentially yield faster convergence, better function approximation, and better parameter estimation. For that reason and so as to possibly exploit their beneficial sides, we have also worked on merging the concepts of RLS and CL.

In essence, we look at the problem of on-line function approximation in the DT framework from a very fundamental point of view, while deriving and developing learning algorithms using the concept of CL so as to add to its already existing CT literature. On-line function approximation is off particular interest because of its use in controls, especially within an adaptive control loop.

The following points detail our problem formulation and contributions.

- We formulate a very generic and dimensionally complex uncertainty approximation problem, which is not the typical course of action in literature;

- In each of the following points, we investigate both the structured and unstructured uncertainty approximation cases (i.e., SUAC and UUAC). It should be added that, as mentioned above, we are mostly concerned with on-line uncertainty approximation;

  - We study in great detail the use of the Normalized Gradient (NG) descent and Recursive Least Squares learning methods for approximation in DT settings so as to derive conditions for stability of the parameter error. By doing so, we add to the existing literature on function approximation, given the nature of our problem;
  - Our main contributions come from developing and thoroughly studying the DT Normalized Gradient (DTNG) based Concurrent Learning and the DT Normalized Recursive Least Squares (DTNRLS) based algorithms. Here also, we derive conditions for stability of the parameter error;
  - More specifically, we show that both the DTNG based CL and DTNRLS based CL algorithms analytically guarantee better parameter identification properties, thus, potentially, better learning properties, granted the recorded data when employing the concept

of Concurrent Learning is rich enough. Richness of the recorded data is linked with verification of the aforementioned CL condition. In fact, provided the CL condition on the recorded data is satisfied, in the SUAC, we can show exponential convergence and asymptotic convergence of the parameter error to the origin when using the DTNG based CL algorithm and DTNRLS based CL algorithm respectively. With the same condition being verified, in the UUAC, we can show that the parameter error is ultimately, uniformly bounded using either algorithms and, particularly, in the case of the DTNRLS based CL algorithm, asymptotic convergence of the parameter error to the origin. It should be noted that, unlike the PE condition, the CL condition is less demanding as it only concerns the recorded data, thus, only a portion of the past information. Moreover, it is much easier to implement and verify on-line as opposed to the PE condition;

- We use our DTNG based CL algorithm for indirect adaptive control of a single state system bearing structured uncertainties and our DTNRLS based CL algorithm for system identification of a robot. We show via derivations and illustrations the merits of CL. For the indirect adaptive control problem in particular, we show that, besides the parameter error, the tracking error is also exponentially stable.

1.7 Outline

We describe in details the function approximation problem in Chapter II. We pose a function approximation problem that is generic in nature and consider two possible approximation models. The concept of Concurrent Learning is then reviewed in Chapter III. We explore the DTNG algorithm and develop the DTNG based CL algorithm in IV. Both algorithms, their results and properties are summarized in Tables 4.1 and 4.2 respectively. In a similar fashion, we also study the DTNRLS algorithm and derive the DTNRLS based CL algorithm in V, both of which are respectively presented in Tables 5.1 and 5.2 along with their results and properties. Chapter VI presents data recording procedures that can be used when applying CL. Specifically, we describe and compare two ways of recording data in memory. Illustrations of our developed algorithms are provided in VII. In Chapters VIII and IX we apply our developed learning algorithms to two problems. They are the indirect adaptive control of a class of single state systems and the system identification of a Comau Racer robot.

CHAPTER II

MATHEMATICAL PRELIMINARIES: FUNCTION APPROXIMATION, LYAPUNOV ANALYSIS, AND DEFINITIONS

2.1 Discrete-Time Approximation

Letting $k_0 \in \mathbb{N}_+$ and $k_s = k_0 + N - 1 \in \mathbb{N}$, for some $N \in \mathbb{N}_+$ and, thus, $k_s \geq k_0$, throughout this note, we use $k \in \{k_0, k_0 + 1, k_0 + 2, \ldots, k_s\} \cap \mathbb{N}$ to denote a discrete-time step and $x = x(k) \in \mathbb{R}^{r_x}$, $r_x \in \mathbb{N}_+$, to denote a known, computable measurement. Essentially, k_0 is the discrete-time at which we start sampling, k_s is the discrete-time at which we stop sampling, and the strictly positive integer N represents the number of samples from start to finish. Let $f\left(x(k)\right) \in \mathbb{R}^{r_f \times c_f}$, $r_f, c_f \in \mathbb{N}_+$, be an unknown, but, measurable or computable uncertainty to approximate.

Now, let $\theta \in \mathbb{R}^{r_\theta \times c_\theta}$, $r_\theta, c_\theta \in \mathbb{N}_+$, be an unknown, constant parameter matrix and $\phi\left(x(k)\right) \in \mathbb{R}^{r_\phi \times c_\phi}$, $r_\phi, c_\phi \in \mathbb{N}_+$, be a computable regressor. Moreover, let $\mathcal{F}\left(x(k), \theta\right) \in \mathbb{R}^{r_f \times c_f}$ be given by

$$\mathcal{F}\left(x(k), \theta\right) = \mathcal{M}_n(x(k), \theta)\,. \tag{2.1}$$

For some $a_x \in \mathbb{R}^{r_x \times c_x}$, $\phi(a_x) \in \mathbb{R}^{r_\phi \times c_\phi}$, $a_\theta \in \mathbb{R}^{r_\theta \times c_\theta}$, and $n = 1, 2$, in the present work, we consider models

$$\mathcal{M}_n(a_x, a_\theta) = \mathcal{M}_1(a_x, a_\theta) = a_\theta^\top \phi(a_x), \text{ if } n = 1, \text{ and} \tag{2.2a}$$

$$\mathcal{M}_n(a_x, a_\theta) = \mathcal{M}_2(a_x, a_\theta) = \phi^\top(a_x) a_\theta, \text{ if } n = 2. \tag{2.2b}$$

for approximation of $f(x(k))$ with measurement $x(k)$, $k \geq k_0$, kept inside a predefined compact set $D_x \subset \mathbb{R}^{r_x}$. On one hand, using model (2.2a) means $r_\theta = r_\phi$, $c_\theta = r_f$, and $c_\phi = c_f$. On the other, though we still have that $r_\theta = r_\phi$ with model (2.2b), it differs from (2.2a) in that $c_\theta = c_f$ and $c_\phi = r_f$. The previous dimensionality analysis is summarized in Table 2.1.

Table 2.1: Model dimentionality analysis.

Model 1: $\mathcal{M}_n = \mathcal{M}_1$	$r_\theta = r_\phi$, $c_\theta = r_f$, and $c_\phi = c_f$
Model 2: $\mathcal{M}_n = \mathcal{M}_2$	$r_\theta = r_\phi$, $c_\theta = c_f$, and $c_\phi = r_f$

Notice that both models (2.2a) and (2.2b) are linear in the parameter matrix θ; and, with respect to θ, a model in the form of those in (2.2) is known in literature as a Linear Parametric Model (LPM) or a linear in the parameter approximator model.

Taking into account approximation imperfections, we also define the representation error $w(x(k)) \in \mathbb{R}^{r_f \times c_f}$ as

$$w(x(k)) = f(x(k)) - \mathcal{F}(x(k), \theta). \tag{2.3}$$

Let $\|\cdot\|_F$ denote the Frobenius norm operator (see C.7). Error $w(x(k))$, given by (2.3), is typically defined as the subsisting functional reconstruction error when $\theta \in D_\theta \subset \mathbb{R}^{r_\theta \times c_\theta}$ is such that

$$\theta = \arg\min_{\theta \in D_\theta} \|w(x(k))\|_F = \arg\min_{\theta \in D_\theta} \|f(x(k)) - \mathcal{F}(x(k), \theta)\|_F \tag{2.4}$$

represents the best or ideal parameter matrix that can be used for approximating f on D_x, with θ either constrained to the set D_θ or left unconstrained by making $D_\theta = \mathbb{R}^{r_\theta \times c_\theta}$.

For approximations of type SUAC, only parametric uncertainties exist, as $\phi(x(k))$ is perfectly known. Thus, given the appropriate θ, $\mathcal{F}(x(k), \theta)$ can fully match $f(x(k))$. For these reasons, it can be assumed that there is no representation error, i.e., $w(x(k)) \equiv [0]^{r_f \times c_f}$ for all x and k when it comes to approximating structured uncertainties. When dealing with a UUAC problem, because of the added functional uncertainties, i.e., the lack of knowledge of an ideal $\phi\left(x(k)\right)$, universal approximators, such as Radial Basis function Neural Network (RBNN), Multi Layer Perceptron (MLP), Fuzzy systems, are typically made use of to construct $\phi\left(x(k)\right)$. We mathematically define an RBNN structure next.

2.2 Radial Basis Function Neural Network

Using the same notation for the regressor as done above, the regressor of a LPM using a RBNN structure with Gaussian basis functions can be computed as

$$\phi\left(x(k)\right) = \left[\ \zeta_1(x(k))\,,\quad \zeta_2(x(k))\,,\quad \ldots,\quad \zeta_{c_\phi}(x(k))\ \right], \tag{2.5}$$

where the vectors $\zeta_i \in \mathbb{R}^{r_\phi}$, $r_\phi = r_\theta$ and $i = 1, 2, \ldots, c_\phi$, are such that

$$\zeta_i(x(k)) = \begin{bmatrix} b_{\zeta_i} \\ e^{-\frac{\left\|x(k)-\mu_{\zeta_i,1}\right\|^2}{2\sigma^2_{\zeta_i,1}}} \\ e^{-\frac{\left\|x(k)-\mu_{\zeta_i,2}\right\|^2}{2\sigma^2_{\zeta_i,2}}} \\ \vdots \\ e^{-\frac{\left\|x(k)-\mu_{\zeta_i,r_\phi-1}\right\|^2}{2\sigma^2_{\zeta_i,r_\phi-1}}} \end{bmatrix}, \tag{2.6}$$

with $\|\cdot\|$ denoting the 2-norm or ℓ^2-norm operator (see (C.5) in Appendix C for more details). For $i = 1, 2, \ldots, c_\phi$ and $j = 1, 2, \ldots, r_\phi - 1$, $\mu_{\zeta_i,j} \in \mathbb{R}^{r_x}$ and $\sigma_{\zeta_i,j} \in \mathbb{R}_+$ are, respectively, the centroids and the spreads of the Gaussian basis functions $e^{-\frac{\left\|x(k)-\mu_{\zeta_i,j}\right\|^2}{2\sigma^2_{\zeta_i,j}}}$, and $b_{\zeta_i} \in \mathbb{R}$ is a constant bias term that can be used if desired or otherwise replaced by an r_ϕ-th basis function. In this work, we will make use of such a structure to construct the regressor when approximating an unstructured uncertainty.

Per universal approximator properties (see [3, 5]), while making use an RBNN structure, we can assume that a good, ideal θ exists such that the representation error $w\left(x(k)\right)$ that results from approximating $f\left(x(k)\right)$ on D_x ($x \in D_x$) using the models in (2.2) is bounded. That is, there exists an upper bound $W \geq 0$ of $\|w\left(x(k)\right)|_F$ for all k, $x \in D_x$, and $\theta \in D_\theta$ such that,

$$W = \sup_{x \in D_x} \left\|w\left(x(k)\right)\right\|_F,$$

or, so that

$$\left\|w\left(x(k)\right)\right\|_F \leq W. \tag{2.7}$$

Lastly, it is known in literature that if a sufficient number of basis functions are used in an RBNN structured regressor, it is possible to make W arbitrarily small.

2.3 On-Line Function Approximation

With the true parameter θ being unknown, we define its estimate $\hat{\theta}(k) \in \mathbb{R}^{r_\theta \times c_\theta}$ to be used for approximation purposes. We also define the scheme $\mathcal{F}\left(x(k), \hat{\theta}(k)\right) \in \mathbb{R}^{r_f \times c_f}$ modeled as

$$\mathcal{F}\left(x(k), \hat{\theta}(k)\right) = \mathcal{M}_n\left(x(k), \hat{\theta}(k)\right) \tag{2.8}$$

after $\mathcal{F}\left(x(k), \theta\right)$ in (2.1). Let $q(k) \in \mathbb{R}^{r_f \times c_f}$ denote the approximation error, which we can be numerically computed as

$$q(k) = \mathcal{F}\left(x(k), \hat{\theta}(k)\right) - f\left(x(k)\right) \tag{2.9a}$$

$$= \mathcal{M}_n\left(x(k), \hat{\theta}(k)\right) - f\left(x(k)\right), \tag{2.9b}$$

Notice that (2.9b) stems from (2.1).

We now define the parameter error

$$\tilde{\theta}(k) = \hat{\theta}(k) - \theta, \tag{2.10}$$

which means, using (2.9a), that

$$q(k) = \mathcal{F}\left(x(k), \tilde{\theta}(k)\right) - \left(f\left(x(k)\right) - \mathcal{F}\left(x(k), \theta\right)\right), \tag{2.11}$$

where, given the linearity in parameter property of $\mathcal{M}_n$ (and, therefore, $\mathcal{F} = \mathcal{M}_n$),

$$\begin{aligned} \mathcal{F}\left(x(k), \tilde{\theta}(k)\right) &= \mathcal{F}\left(x(k), \hat{\theta}(k)\right) - \mathcal{F}\left(x(k), \theta\right), \\ &= \mathcal{M}_n\left(x(k), \tilde{\theta}(k)\right). \end{aligned} \tag{2.12}$$

Hence, from (2.3) and (2.12), (2.11) becomes

$$q(k) = \mathcal{M}_n\left(x(k), \tilde{\theta}(k)\right) - w\left(x(k)\right). \tag{2.13}$$

Because we are considering two approximation models, i.e, $\mathcal{M}_1$ in (2.2a) and $\mathcal{M}_2$ in (2.2a), for succinctness, we define the error $q_\epsilon(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}}$ (essentially an approximation error), which is to be computed as

$$q_\epsilon(k) = \begin{cases} q^\top(k), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ q(k), & \text{if } \mathcal{M}_n = \mathcal{M}_2, \end{cases} \tag{2.14}$$

and, hence, meaning r_{q_ϵ} and c_{q_ϵ} depend on the model used. Recall that $q \in \mathbb{R}^{r_f \times c_f}$. If using (2.2a), $\mathcal{M}_n = \mathcal{M}_1$ and, subsequently, $r_{q_\epsilon} = c_f$ and $c_{q_\epsilon} = r_f$. Instead, if using (2.2b), $\mathcal{M}_n = \mathcal{M}_2$ and, in this case, $r_{q_\epsilon} = r_f$ and $c_{q_\epsilon} = c_f$. However, no matter which of the two approximation models in (2.2) is used, from (2.9b), (2.13), and (2.14),

$$q_\epsilon(k) = \phi^\top(x(k))\,\hat{\theta}(k) - f_n(k), \tag{2.15a}$$

$$= \phi^\top(x(k))\,\tilde{\theta}(k) - w_\epsilon(k), \tag{2.15b}$$

where the function $f_n(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}}$ and the residual error $w_\epsilon(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}}$ are such that

$$f_n(k) = \begin{cases} f^\top(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ f\left(x(k)\right), & \text{if } \mathcal{M}_n = \mathcal{M}_2, \end{cases} \tag{2.16}$$

and

$$w_\epsilon(k) = \begin{cases} w^\top(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ w\left(x(k)\right), & \text{if } \mathcal{M}_n = \mathcal{M}_2. \end{cases} \tag{2.17}$$

2.4 Normalization Signal

For some (possibly time-varying) bounded scalar $\alpha = \alpha(k) > 0$, a signal $\rho\left(x(k)\right) \in \mathbb{R}^{r_\rho \times c_\rho}$, with $r_\rho, c_\rho \in \mathbb{N}_+$, and a positive semi-definite, symmetric, and (possibly time-varying) bounded matrix $\Xi_m = \Xi_m(k) \in \mathbb{R}^{r_\rho \times r_\rho}$ ($\Xi_m > 0$ and $\Xi_m^\top = \Xi_m$), we also define the normalization signal $m(k) \in \mathbb{R}^{c_\rho \times c_\rho}$ that is such that

$$m^2(k) = \alpha I_{c_\rho} + \rho^\top(x(k))\, \Xi_m \rho\left(x(k)\right), \tag{2.18}$$

or, consequently,

$$m(k) = \Big(\alpha I_{c_\rho} + \rho^\top(x(k))\, \Xi_m \rho\left(x(k)\right)\Big)^{\frac{1}{2}}.$$

On the right hand side of (2.18), I_{c_ρ} denotes a c_ρ-by-c_ρ identity matrix. Given that $\Xi_m > 0$, for all k, $\rho^\top(x(k))\, \Xi_m \rho\left(x(k)\right)$ is at least positive semi-definite, i.e.,

$$\phi^\top(x(k))\, \Xi_m \phi\left(x(k)\right) \geq 0,$$

and, with $\alpha > 0$ and $I_{c_\rho} > 0$, both $m^2(k)$ and $m(k)$ are positive definite, i.e., $m^2(k) > 0$ and $m(k) > 0$. Hence, with $m^2(k) > 0$, its square root $m(k)$ is unique. Further, $\left(m^2(k)\right)^{-1}$ as well as $m^{-1}(k)$ exist and are also both positive definite. Additionally, given (2.18), notice that $m^2(k)$ and $m(k)$ are symmetric. That is $\left(m^2(k)\right)^\top = m^2(k)$, $m^\top(k) = m(k)$. As well, their respective inverse, i.e., $\left(m^2(k)\right)^{-1}$ and $m^{-1}(k)$, are also symmetric. We can bound the signal $m^2(k)$ in (2.18) as

$$0 < \alpha I_{c_\rho} \leq m^2(k) \leq \overline{\lambda}_{m^2,k} I_{c_\rho}, \tag{2.19}$$

with

$$\overline{\lambda}_{m^2,k} = \alpha + \lambda_{\max}\Big(\rho^\top(x(k))\, \Xi_m \rho\left(x(k)\right)\Big), \tag{2.20}$$

from which $\lambda_{\max}(\cdot)$ is used here to denote the maximum eigenvalue operator. From (2.19), we have that

$$\frac{1}{\sqrt{\overline{\lambda}_{m^2,k}}} I_{c_\rho} \leq m^{-1}(k) \leq \frac{1}{\sqrt{\alpha}} I_{c_\rho}$$

and, subsequently,

$$\left\| \frac{1}{\sqrt{\overline{\lambda}_{m^2,k}}} I_{c_\rho} \right\|_F^2 = \frac{1}{\overline{\lambda}_{m^2,k}} \left\| I_{c_\rho} \right\|_F^2 \leq \left\| m^{-1}(k) \right\|_F^2 \leq \left\| \frac{1}{\sqrt{\alpha}} I_{c_\rho} \right\|_F^2 = \frac{1}{\alpha} \left\| I_{c_\rho} \right\|_F^2$$

or, given that $\left\| I_{c_\rho} \right\|_F = \sqrt{c_\rho}$ according to the mathematical definition of the Frobenius norm in (C.7),

$$\frac{c_\rho}{\overline{\lambda}_{m^2,k}} \leq \left\| m^{-1}(k) \right\|_F^2 \leq \frac{c_\rho}{\alpha}. \tag{2.21}$$

Furthermore, noting that

$$\left(m^2(k) \right)^{-1} = m^{-1}(k) m^{-1}(k),$$

using (C.1), (C.9), and with $\lambda_{\left\{ (m^2(k))^{-1} \right\}_r}$, $r = 1, 2, \ldots, c_\phi$, representing the eigenvalues of $\left(m^2(k) \right)^{-1} \in \mathbb{R}^{c_\rho \times c_\rho}$,

$$\begin{aligned} \text{tr} \left\{ \left(m^2(k) \right)^{-1} \right\} = \text{tr} \left\{ m^{-1}(k) m^{-1}(k) \right\} = \left\| m^{-1}(k) \right\|_F^2 &= \sum_{r=1}^{c_\rho} \lambda_{\left\{ (m^2(k))^{-1} \right\}_r}, \\ &\geq \lambda_{\max} \left(\left(m^2(k) \right)^{-1} \right). \end{aligned}$$

Hence, with $\lambda_{\max} \left(\left(m^2(k) \right)^{-1} \right) \leq \left\| m^{-1}(k) \right\|_F^2$ and being able to write

$$0 < \left(m^2(k) \right)^{-1} \leq \lambda_{\max} \left(\left(m^2(k) \right)^{-1} \right) I_{c_\rho},$$

we get that

$$0 < \left(m^2(k) \right)^{-1} \leq \lambda_{\max} \left(\left(m^2(k) \right)^{-1} \right) I_{c_\rho} \leq \left\| m^{-1}(k) \right\|_F^2 I_{c_\rho}, \tag{2.22}$$

Moreover, befitting its definition, $m(k)$ ensures that the signal

$$\rho_m(x(k)) = \rho\left(x(k)\right) m^{-1}(k), \tag{2.23}$$

hence $\rho_m(x(k)) \in \mathbb{R}^{r_\rho \times c_\rho}$, is bounded for all k. It will formally be shown in later chapters that signals defined in the same vein as $\rho_m(x(k))$ are indeed bounded.

2.5 Lyapunov Analysis

Function approximation involves developing and/or applying an adaptation law for update of the parameter estimate $\hat{\theta}(k)$. Boundedness and convergence of the parameter error $\tilde{\theta}(k)$ are key to the success of the approximation scheme.

Letting tr$(\cdot)$ denote the trace operator, throughout this note, we will study the trajectory of $\tilde{\theta}(k)$ using the positive definite, decrescent, and radially unbounded Lyapunov function

$$V(k) = \text{tr}\left\{\tilde{\theta}^\top(k)\Xi_V\tilde{\theta}(k)\right\}, \tag{2.24}$$

with (possibly time-varying) $\Xi_V = \Xi_V(k) \in \mathbb{R}^{r_\theta \times r_\theta}$, with $r_\theta = r_\phi$, chosen as a positive definite, symmetric, and bounded matrix ($\Xi_V > 0$ and $\Xi_V^\top = \Xi_V$). From (2.24), with $\Xi_V > 0$, and because of (C.9), there exist positive scalars $\underline{\beta}_V$ and $\overline{\beta}_V$ that are such that $0 < \underline{\beta}_V < \overline{\beta}_V$ and

$$\underline{\beta}_V \left\|\tilde{\theta}(k)\right\|_F^2 \leq V(k) \leq \overline{\beta}_V \left\|\tilde{\theta}(k)\right\|_F^2. \tag{2.25}$$

Furthermore, consider the rate of change

$$\Delta V(k) = V(k) - V(k-1) \tag{2.26}$$

or, equivalently,

$$\Delta V(k+1) = V(k+1) - V(k)$$

of $V(k)$. If we can show at the very minimum that, for all k, $\Delta V(k) \le 0$, then, according to Lyapunov theory [3, 5]), $V(k)$ is bounded and $\tilde{\theta}(k)$ can also be shown to be bounded. Besides boundedness,

1. If $V(k+1) < V(k)$, then $\tilde{\theta}(k)$ is asymptotically bounded;
2. If $V(k+1) \le \underline{\beta}\, V(k)$, for some $0 < \underline{\beta} < 1$, then $\tilde{\theta}(k)$ is (not only asymptotically bounded, but furthermore) exponentially bounded.

2.6 Some Definitions

The following list highlights noteworthy definitions and properties that we will make use of in the present note. First, however, it is important to remember that, regardless of the model utilized for approximation, i.e., (2.2a) or (2.2b), $r_\theta = r_\phi$.

- Let I_r, $r \in \mathbb{N}_+$, denote an r-by-r identity matrix. According to the definition of the Frobenius norm, i.e., (C.7), we have that $\|I_r\|_F = \sqrt{r}$.
- Let $\Phi_k \in \mathbb{R}^{r_\phi \times r_\phi}$ denote the instantaneous information matrix

  $$\Phi_k = \phi\left(x(k)\right) \phi^\top(x(k))\,. \tag{2.27}$$

  We can see that Φ_k is symmetric, i.e., $\Phi_k^\top = \Phi_k$, and positive semidefinite, i.e., $\Phi_k \ge 0$ for all k. Having previously defined $\lambda_{\max}(\cdot)$ as the maximum eigenvalue operator, we now let $\lambda_{\min}(\cdot)$ denote the minimum eigenvalue operator. As such, for

  $$\underline{\lambda}_{\Phi_k} = \lambda_{\min}(\Phi_k) \text{ and } \overline{\lambda}_{\Phi_k} = \lambda_{\max}(\Phi_k)\,,$$

  we can write that

  $$\underline{\lambda}_{\Phi_k} I_{r_\phi} \le \Phi_k = \phi\left(x(k)\right) \phi^\top(x(k)) \le \overline{\lambda}_{\Phi_k} I_{r_\phi} \tag{2.28}$$

  for all k.

- The rest of the definitions, properties, and propositions, including (but not limited to) persistency of excitation, vector and matrix norm operators, and the use of $\mathrm{tr}(\cdot)$, $\mathrm{vec}(\cdot)$, and $\|\cdot\|_F$ to respectively denote the trace, vectorization, and Frobenius norm operators, are given in the appendices.

For brevity, we will use $\|\cdot\|$ to denote the ℓ^2-norm operator (see (C.5)).

Next, after defining the concept of Concurrent Learning, we study in depth both the Gradient Descent and Recursive Least Squares methods for function approximation in discrete-time settings and then go on to develop their respective Concurrent Learning modifications.

CHAPTER III

CONCURRENT LEARNING

3.1 Concept, Definitions, and Rank Condition

As suitably named, *Concurrent Learning* is a memory based method, whereby carefully selected and recorded past information (saved in memory) is duly combined together with current information during the learning process. Its use of memory makes it therefore quite similar to human learning. The concept of Concurrent Learning was developed by Girish Chowdhary [15, 16, 14, 17, 18], though in the CT framework. The implied, used memory storage when employing Concurrent Learning, coined *history stack*, will be played by matrices Z and/or Z_G defined below in (3.1) and (3.8) respectively.

Recall regressor $\phi\left(x(k)\right) \in \mathbb{R}^{r_\phi \times c_\phi}$, used for approximation purposes and elementally defined in (2.5) as

$$\phi\left(x(k)\right) = \left[\begin{array}{cccc} \zeta_1(x(k)), & \zeta_2(x(k)), & \ldots, & \zeta_{c_\phi}(x(k)) \end{array}\right],$$

with vectors $\zeta_i(x(\tau_j)) \in \mathbb{R}^{r_\phi}$, $i = 1, 2, \ldots c_\phi$, representing the columns of $\phi\left(x(k)\right)$. Let $Z \in \mathbb{R}^{r_Z \times c_Z}$, $r_Z = r_\phi = r_\theta$ and $c_Z \in \mathbb{N}_+$, be a matrix whose columns are composed of vectors $\zeta_i(x(\tau_j))$, $i \in \{1, 2, \ldots, c_\phi\}$, of the regressor $\phi\left(x(\tau_j)\right)$ (see (2.5)), computed at discrete time τ_j,

$k_0 \leq \tau_j < k$, with $j \in \{1, 2, \ldots, c_Z\}$. That is

$$Z = \left[\begin{array}{cccc} \zeta_i(x(\tau_1)) \,, & \zeta_i(x(\tau_2)) \,, & \ldots, & \zeta_i(x(\tau_{c_Z})) \end{array} \right]. \tag{3.1}$$

Given (2.5), substituting $\rho\left(x(k)\right)$ in (2.18) by $\phi\left(x(k)\right)$, and for the scalars $g_{n,p}(k)$ and $\bar{g}_{n,n}(k)$, $n \in \{1, 2, \ldots, c_\phi\}$ as well as $p \in \{1, 2, \ldots, c_\phi\}$, defined as

$$g_{n,p}(k) = \zeta_n^\top(x(k))\, \Xi_m \zeta_p(x(k)) \tag{3.2}$$

and

$$\bar{g}_{n,n}(k) = \alpha + g_{n,n}(k) = \alpha + \zeta_n^\top(x(k))\, \Xi_m \zeta_n(x(k))\,, \tag{3.3}$$

we have that

$$\alpha I_{c_\phi} + \phi^\top(x(k))\, \Xi_m \phi\left(x(k)\right) = \begin{bmatrix} \bar{g}_{1,1}(k) & g_{1,2}(k) & \ldots & g_{1,c_\phi}(k) \\ g_{2,1}(k) & \bar{g}_{2,2}(k) & \ldots & g_{2,c_\phi}(k) \\ \vdots & & \ddots & \vdots \\ g_{c_\phi,1}(k) & g_{c_\phi,2}(k) & \ldots & \bar{g}_{c_\phi,c_\phi}(k) \end{bmatrix}.$$

Given that $\alpha > 0$, $\Xi_m > 0$, and

$$g_{n,n}(k) = \zeta_n^\top(x(k))\, \Xi_m \zeta_n(x(k)) \geq 0,$$

notice that for any k and $n \in \{1, 2, \ldots, c_\phi\}$, the scalar $\bar{g}_{n,n}(k) > 0$. Since a scalar (and though redundant), $\bar{g}_{n,n}(k) = \left\|\bar{g}_{n,n}(k)\right\|_F$, and, while making use of (C.5), (C.7), (C.8), (C.10), and (C.11), we can find a scalar $\overline{\alpha}_g$ such that $0 < \overline{\alpha}_g \leq 1$ and

$$\bar{g}_{n,n}(k) = \left\|\bar{g}_{n,n}(k)\right\|_F \geq \overline{\alpha}_g \left(\alpha + \Lambda^2_{\zeta_n,k}\right), \tag{3.4}$$

where, for

$$\Lambda_{\Xi_m} = \left\|\Xi_m\right\|_F\,, \tag{3.5a}$$

$$\Lambda_{\zeta_n,k} = \left\|\Xi_m\right\|_F^{\frac{1}{2}} \left\|\zeta_n(x(k))\right\|_F = \Lambda_{\Xi_m}^{\frac{1}{2}} \left\|\zeta_n(x(k))\right\|. \tag{3.5b}$$

Expressions (3.4), (3.5a), and (3.5b) allow us to say that

$$\left\|\frac{\zeta_n(x(k))}{\sqrt{\bar{g}_{n,n}(k)}}\right\|_F = \frac{\|\zeta_n(x(k))\|_F}{\left\|\sqrt{\bar{g}_{n,n}(k)}\right\|_F} = \frac{\|\zeta_n(x(k))\|}{\left\|\sqrt{\bar{g}_{n,n}(k)}\right\|_F} \leq \frac{\Lambda_{\Xi_m}^{-\frac{1}{2}}\Lambda_{\zeta_n,k}}{\left(\overline{\alpha}_g\left(\alpha+\Lambda_{\zeta_n,k}^2\right)\right)^{\frac{1}{2}}} = B_g. \tag{3.6}$$

Since Ξ_m is to be picked such that it is bounded, (3.6) means that $\frac{\zeta_n(x(k))}{\sqrt{\bar{g}_{n,n}(k)}} \in \mathbb{R}^{r_\phi}$ is indeed bounded. Actually, similarly to (3.6), we can show that

$$\left\|\frac{\zeta_n(x(k))\,\zeta_n^\top(x(k))}{\bar{g}_{n,n}(k)}\right\|_F \leq \frac{\Lambda_{\Xi_m}^{-\frac{1}{2}}\Lambda_{\zeta_n,k}^2}{\overline{\alpha}_g\left(\alpha+\Lambda_{\zeta_n,k}^2\right)}, \tag{3.7}$$

and conclude that $\frac{\zeta_n(x(k))\,\zeta_n^\top(x(k))}{\bar{g}_{n,n}(k)} \in \mathbb{R}^{r_\phi\times r_\phi}$ is also bounded. It should be added that, for the derived Frobenius norm upper-bounds above, i.e., (3.6) and (3.7), boundedness is contingent upon $\overline{\alpha}_g$, which, according to (3.3) and (3.4), can be made to be far away from zero and closer to 1. In any case, the main reason for going through the previous analysis is to set a basis for the definitions of matrix $Z_G \in \mathbb{R}^{rz\times cz}$, matrix $\bar{Z}_G \in \mathbb{R}^{rz\times cz}$, and row vector $G \in \mathbb{R}^{1\times cz}$, made to respectively contain the bounded vectors $\frac{\zeta_i(x(\tau))}{\sqrt{\bar{g}_{i,i}(\tau_j)}}$, vectors $\frac{\zeta_i(x(\tau))}{\bar{g}_{i,i}(\tau_j)}$, and the positive scalars $\bar{g}_{i,i}(\tau_j)$, with $i \in \{1, 2, \ldots, c_\phi\}$, $j \in \{1, 2, \ldots, c_Z\}$, and DT τ_j being such that $k_0 \leq \tau_j < k$. That is,

$$Z_G = \left[\begin{array}{cccc} \frac{\zeta_i(x(\tau_1))}{\sqrt{\bar{g}_{i,i}(\tau_1)}}, & \frac{\zeta_i(x(\tau_2))}{\sqrt{\bar{g}_{i,i}(\tau_2)}}, & \ldots, & \frac{\zeta_i(x(\tau_{c_Z}))}{\sqrt{\bar{g}_{i,i}(\tau_{c_Z})}} \end{array}\right], \tag{3.8}$$

$$\bar{Z}_G = \left[\begin{array}{cccc} \frac{\zeta_i(x(\tau_1))}{\bar{g}_{i,i}(\tau_1)}, & \frac{\zeta_i(x(\tau_2))}{\bar{g}_{i,i}(\tau_2)}, & \ldots, & \frac{\zeta_i(x(\tau_{c_Z}))}{\bar{g}_{i,i}(\tau_{c_Z})} \end{array}\right], \tag{3.9}$$

and

$$G = \left[\begin{array}{cccc} \bar{g}_{i,i}(\tau_1), & \bar{g}_{i,i}(\tau_2), & \ldots, & \bar{g}_{i,i}(\tau_{c_Z}) \end{array}\right]. \tag{3.10}$$

Notice that Z_G and $\bar{Z}_G$ can actually be constructed by knowing Z and G. Hence, one needs not to put together and carry along all three memory banks, i.e., Z, Z_G, and $\bar{Z}_G$, saved up in memory at all times, as having Z and G would suffice. It is also worthwhile mentioning that Z_G may be practically easier to operate on than Z and, for that matter, $\bar{Z}_G$, because of the boundedness of its elements, i.e., $\dfrac{\zeta_i(x(\tau_j))}{\sqrt{\bar{g}_{i,i}(\tau_j)}}$. On that note, given (3.6), (C.10), and the fact that $Z_G \in \mathbb{R}^{r_Z \times c_Z}$ in (3.9) can be written as a summation of c_Z matrices of size r_Z-by-c_Z, with each matrix containing only one of the c_Z columns of Z_G at a time and the rest of the elements being zero, we can in fact write that

$$\|Z_G\|_F \leq \sum_{j=1}^{c_Z} \left\| \frac{\zeta_i(x(\tau_j))}{\sqrt{\bar{g}_{i,i}(\tau_j)}} \right\|_F \leq c_Z B_g. \tag{3.11}$$

Concerning $\bar{Z}_G$, notice that though, for all k and $n \in \{1, 2, \ldots, c_\phi\}$, $\dfrac{\zeta_n(x(k))}{\sqrt{\bar{g}_{n,n}(k)}}$ is bounded as (3.6) shows, we cannot readily make the same conclusion for

$$\frac{\zeta_n(x(k))}{\bar{g}_{n,n}(k)} = \frac{\zeta_n(x(k))}{\sqrt{\bar{g}_{n,n}(k)}} \frac{1}{\sqrt{\bar{g}_{n,n}(k)}},$$

as $\sqrt{\bar{g}_{n,n}(k)}$ may be so small or, equivalently, $\dfrac{1}{\sqrt{\bar{g}_{n,n}(k)}}$ may be so large that $\left\| \dfrac{\zeta_n(x(k))}{\bar{g}_{n,n}(k)} \right\|_F$ is also large. Obviously, given (3.3), we can avoid having

$$\frac{1}{\sqrt{\bar{g}_{n,n}(k)}} = \frac{1}{\sqrt{\alpha + g_{n,n}(k)}} = \frac{1}{\sqrt{\alpha + \zeta_n^\top(x(k))\, \Xi_m \zeta_n(x(k))}} \geq 1$$

by picking $\alpha \geq 1$.

Furthermore, say for any x and k, $f\left(x(k)\right) \in \mathbb{R}^{r_f \times c_f}$ is such that

$$f\left(x(k)\right) = \left[\begin{array}{cccc} f_1(x(k)), & f_2(x(k)), & \ldots, & f_{c_f}(x(k)) \end{array}\right]$$

and $l\left(x(k)\right) = f^\top(x(k)) \in \mathbb{R}^{c_f \times r_f}$ is such that

$$l\left(x(k)\right) = \left[\begin{array}{cccc} l_1\left(x(k)\right), & l_2\left(x(k)\right), & \ldots, & l_{r_f}\left(x(k)\right) \end{array}\right],$$

where $f_i(x(k)) \in \mathbb{R}^{r_f}$, $i = 1, 2, \ldots, c_f$, and $l_j(x(k)) \in \mathbb{R}^{c_f}$, $j = 1, 2, \ldots, r_f$, are column vectors of matrices $f(x)$ and $l(x)$ respectively. Let $F_n \in \mathbb{R}^{r_F \times c_F}$, $n = 1, 2$ and $r_F, c_F \in \mathbb{N}_+$, be a matrix containing the uncertainty entities in $f\left(x(\tau_j)\right)$. That is, either $f_i(x(\tau_j))$ or $l_i(x(\tau_j))$. If using model (2.2a), then, we set $r_F = r_f$, $c_F = c_Z$, and, for some values of $i = 1, 2, \ldots, c_f$ representing the columns indexes of $f(x)$,

$$F_n = F_1 = \left[\ f_i(x(\tau_1))\,,\quad f_i(x(\tau_2))\,,\quad \ldots,\quad f_i(x(\tau_{c_Z}))\ \right]. \tag{3.12}$$

Similarly, if using model (2.2b) instead, then, $r_F = c_Z$, $c_F = c_f$, and

$$F_n = F_2 = \left[\ l_i(x(\tau_1))\,,\quad l_i(x(\tau_2))\,,\quad \ldots,\quad l_i(x(\tau_{c_Z}))\ \right]^\top, \tag{3.13}$$

for some $i = 1, 2, \ldots, r_f$ denoting the columns indexes of $l(x)$. For conciseness, we choose not to explicitly show time dependency of matrices Z, Z_G, $\bar{Z}_G$, G, and F, i.e., $Z = Z(k)$, $Z_G = Z_G(K)$, $\bar{Z}_G = \bar{Z}_G(k)$, $G = G(k)$ and, for $n = 1, 2$, $F_n = F_n(k)$, though, they could very well be time dependent.

Moving along, let $\Phi_{Z,k} \in \mathbb{R}^{r_Z \times r_Z}$ and $\Phi_{Z_G,k} \in \mathbb{R}^{r_Z \times r_Z}$ be defined as

$$\Phi_{Z,k} = ZZ^\top \tag{3.14}$$

and, based on the definition of Z, Z_G, and $\bar{Z}_G$ in (3.1), (3.8), and (3.9) respectively,

$$\Phi_{Z_G,k} = Z_G Z_G^\top = \bar{Z}_G Z^\top. \tag{3.15}$$

We want to bring attention to the second equality in (3.15) because we will make use of it later. Notice further that both $\Phi_{Z,k}$ and $\Phi_{Z_G,k}$ are symmetric, i.e., $\Phi_{Z,k}^\top = \Phi_{Z,k}$ and $\Phi_{Z_G,k}^\top = \Phi_{Z_G,k}$, and (at least) positive semidefinite, i.e., $\Phi_{Z,k} \geq 0$ and $\Phi_{Z_G,k} \geq 0$. Moreover, based on the definitions of Z, Z_G, and $\bar{Z}_G$ in, respectively, (3.1), (3.8), and (3.9), we have that

$$\Phi_{Z,k} = ZZ^\top = \sum_{j=1}^{c_Z} \zeta_i(x(\tau_j))\,\zeta_i^\top(x(\tau_j)) \tag{3.16}$$

and

$$\Phi_{Z_G,k} = Z_G Z_G^\top = \sum_{j=1}^{c_Z} \frac{\zeta_i(x(\tau_j))}{\sqrt{\bar{g}_{i,i}(\tau_j)}} \frac{\zeta_i^\top(x(\tau_j))}{\sqrt{\bar{g}_{i,i}(\tau_j)}} = \sum_{j=1}^{c_Z} \frac{\zeta_i(x(\tau_j))\,\zeta_i^\top(x(\tau_j))}{\bar{g}_{i,i}(\tau_j)}, \tag{3.17}$$

with $\dfrac{\zeta_i(x(\tau_j))\,\zeta_i^\top(x(\tau_j))}{\bar{g}_{i,i}(\tau_j)}$ being bounded for any $i \in \{1, 2, \ldots, c_\phi\}$ and τ_j that is such that $j \in \{1, 2, \ldots, c_Z\}$ and $k_0 \leq \tau_j < k$, as (3.7) shows. Defining

$$\underline{\lambda}_{\Phi_{Z,k}} = \lambda_{\min}(\Phi_{Z,k})\,,\, \overline{\lambda}_{\Phi_{Z,k}} = \lambda_{\max}(\Phi_{Z,k})\,,$$

$$\underline{\lambda}_{\Phi_{Z_G,k}} = \lambda_{\min}(\Phi_{Z_G,k})\,,\text{ and } \overline{\lambda}_{\Phi_{Z_G,k}} = \lambda_{\max}(\Phi_{Z_G,k})\,,$$

for all k,

$$\underline{\lambda}_{\Phi_{Z,k}} I_{r_Z} = \underline{\lambda}_{\Phi_{Z,k}} I_{r_\phi} \leq \Phi_{Z,k} = ZZ^\top \leq \overline{\lambda}_{\Phi_{Z,k}} I_{r_Z} = \overline{\lambda}_{\Phi_{Z,k}} I_{r_\phi}. \tag{3.18}$$

and

$$\underline{\lambda}_{\Phi_{Z_G,k}} I_{r_Z} = \underline{\lambda}_{\Phi_{Z_G,k}} I_{r_\phi} \leq \Phi_{Z_G,k} = Z_G Z_G^\top \leq \overline{\lambda}_{\Phi_{Z_G,k}} I_{r_Z} = \overline{\lambda}_{\Phi_{Z_G,k}} I_{r_\phi}, \tag{3.19}$$

given that $r_Z = r_\phi$. From (3.16) and (3.17), notice that $\underline{\lambda}_{\Phi_{Z_G,k}}$ and $\overline{\lambda}_{\Phi_{Z_G,k}}$ are functions of $\underline{\lambda}_{\Phi_{Z,k}}$ and $\overline{\lambda}_{\Phi_{Z,k}}$ respectively, as well as the strictly positive scalars $\bar{g}_{i,i}$.

Now, as done in the CT framework, we present the condition on the linear independence of the data in the history stack(s). This condition is also known as the rank condition. We will however formulate it in terms of matrix Z_G, which, as mentioned earlier, might be more appropriate to use in practice for numerical operations due to its columns being bounded (and/or, also, the fact that Z_G is bounded as (3.11) reveals). However, it is important to explicitly mention that, because, as show (3.1), (3.8), and (3.9), Z, Z_G, and $\bar{Z}_G$, are constructed with the same column vectors, a condition on Z_G also applies to Z and $\bar{Z}_G$.

Condition 3.1.1. The history stack $Z_G \in \mathbb{R}^{r_Z \times c_Z}$, $r_Z = r_\phi = r_\theta$ and $c_Z \in \mathbb{N}_+$, given by (3.8) contains r_Z linearly independent columns.

Mathematically speaking, **Condition 3.1.1** means Z_G is full row rank, $c_Z \geq r_Z = r_\phi = r_\theta$, and, considering rank$(\cdot)$ as the rank operator, rank$(Z_G) = r_Z$. Thus, from (3.15), $\Phi_{Z_G,k} = Z_G Z_G^\top$ is positive definite, i.e., $\Phi_{Z_G,k} > 0$. With $\Phi_{Z_G,k} > 0$, its eigenvalues $\underline{\lambda}_{\Phi_{Z_G,k}}$ and $\overline{\lambda}_{\Phi_{Z_G,k}}$ are both strictly positive, therefore making $\underline{\lambda}_{\Phi_{Z,k}}$ and $\overline{\lambda}_{\Phi_{Z,k}}$ also strictly positive when taking in account (3.16), (3.17), and, as shown before, the fact that $\bar{g}_{i,i} > 0$ for any $i \in \{1, 2, \ldots, c_\phi\}$. As a result, $\Phi_{Z_G,k}$ being positive definite also means that $\Phi_{Z,k} = ZZ^\top$ is also positive definite.

3.2 Persistency of Excitation for Good Learning

We know from literature that the standard and widely used learning methods of Gradient Descent and Recursive Least Square, which we will closely look at in the next chapters, cannot theoretically guarantee good parameter identification, i.e., convergence of the parameter estimates to their true, ideal values or a neighborhood about their true, ideal values unless the regressor used for approximation (or its bounded version) is persistently exciting [2, 3, 4]. In Appendix B, we mathematically define what it means for a (bounded) signal to be persistently exciting (see **Definition B.2**). A look at the aforementioned definition shows that the PE condition means realizing complete span, for all time, of the approximation space. For that reason, it is a hard, demanding, and impractical condition to achieve, especially in the context of closed-loop control.

Compared with the PE condition, the rank condition, i.e., **Condition 3.1.1**, is a more moderate condition. It certainly differs from the PE condition in that it only deals with a subset of the past information stored in the history stack Z_G. Moreover, it can be implemented and monitored on-line, and can be attained even with $\dfrac{\zeta_i(x(\tau_j))}{\sqrt{\bar{g}_{i,i}(\tau_j)}}$ in Z_G (refer to (3.8)) being, for instance, exciting, therefore verifying **Definition B.1**, as opposed to being PE, as given by **Definition B.2**.

3.3 Using Memory for Concurrent Learning

Granted the history stack Z in (3.1), we denote $\bar{\mathcal{F}}(Z, \theta) \in \mathbb{R}^{r_F \times c_F}$ given by

$$\bar{\mathcal{F}}(Z, \theta) = \bar{\mathcal{M}}_n(Z, \theta)\,, \tag{3.20}$$

where, depending on the model used, i.e., (2.2a) or (2.2b), for some $a_Z \in \mathbb{R}^{r_Z \times c_Z}$, with $r_Z = r_\phi = r_\theta$ as mentioned before, and, again, $a_\theta \in \mathbb{R}^{r_\theta \times c_\theta}$, we define

$$\bar{\mathcal{M}}_n(a_Z, a_\theta) = \bar{\mathcal{M}}_1(a_Z, a_\theta) = a_\theta^\top a_Z, \text{ if } \mathcal{M}_n = \mathcal{M}_1, \text{ and} \tag{3.21a}$$

$$\bar{\mathcal{M}}_n(a_Z, a_\theta) = \bar{\mathcal{M}}_2(a_Z, a_\theta) = a_Z^\top a_\theta, \text{ if } \mathcal{M}_n = \mathcal{M}_2. \tag{3.21b}$$

Recalling our discussion in Chapter II, notice, from (3.21a) and (3.21b) respectively, that when $n = 1$, $c_\theta = r_f = r_F$, $c_F = c_Z$, and $\bar{\mathcal{M}}_1(Z, \theta) \in \mathbb{R}^{c_\theta \times c_Z}$, while, for $n = 2$, $r_F = c_Z$, $c_\theta = c_f = c_F$, and $\bar{\mathcal{M}}_2(Z, \theta) \in \mathbb{R}^{c_Z \times c_\theta}$. Hence, the dimensions of $\bar{\mathcal{M}}_n(Z, \theta)$ and, according to (3.20), $\bar{\mathcal{F}}(Z, \theta)$ agree with that of F_n in each case (i.e, $n = 1, 2$), which is given by (3.12) or (3.13). In fact, remark that $\bar{\mathcal{F}}(Z, \theta)$ approximates the recorded $F_n \in \mathbb{R}^{r_F \times c_F}$, $n = 1, 2$. Moreover, similarly to (2.8), we also define

$$\bar{\mathcal{F}}\left(Z, \hat{\theta}(k)\right) = \bar{\mathcal{M}}_n\left(Z, \hat{\theta}(k)\right) \tag{3.22}$$

for on-line approximation purposes.

By using memory (Z and F_n) and by comparing $\bar{\mathcal{F}}\left(Z, \hat{\theta}(k)\right)$ to F_n (which contains items pertaining to the actual uncertainty being approximated, i.e., $f(x)$ or $l(x) = f^\top(x)$), we could gauge the learning ability of the approximation algorithm. Such a comparison would tell us, as time evolves, how well $\hat{\theta}(k)$ is able to adjust if it were to be used to reconstruct F_n with $\bar{\mathcal{F}}\left(Z, \hat{\theta}(k)\right)$ or, said differently, how good of a learning generalization is being performed. That information can then be re-used when devising the adaptation law.

As such, based on (3.22), we define the approximation error $q_Z(k) \in \mathbb{R}^{r_{q_Z} \times c_{q_Z}}$ based on recorded data as

$$q_Z(k) = \bar{\mathcal{F}}_n\left(Z, \hat{\theta}(k)\right) - F_n \tag{3.23a}$$

$$= \bar{\mathcal{M}}_n\left(Z, \hat{\theta}(k)\right) - F_n. \tag{3.23b}$$

Notice how similar (for good reasons) the expressions in (3.23) are to the ones in (2.9). According to (3.21a) and (3.21b) respectively, if $\mathcal{M}_n = \mathcal{M}_1$, $r_{q_Z} = c_\theta$ and $c_{q_Z} = c_Z$, whereas, when $\mathcal{M}_n = \mathcal{M}_2$, $r_{q_Z} = c_Z$ and $c_{q_Z} = c_\theta$. It is worthwhile re-emphasizing that Z and F_n, $n = 1, 2$, can vary with k and depend on the column vectors in the regressor $\phi\left(x(\tau_j)\right)$ and the uncertainty $f\left(x(\tau_j)\right)$ or $l\left(x(\tau_j)\right) = f^\top(x(\tau_j))$ respectively, for discrete times τ_j, $k_0 \leq \tau_j < k$, with $j = 1, 2, \ldots, c_Z$, as (3.1), (3.12), and (3.13) show. Subsequently, we then have, just like (2.13), that

$$q_Z(k) = \mathcal{M}_n\Big(Z, \tilde{\theta}(k)\Big) - w_Z(x(k)), \tag{3.24}$$

where the incurred representation error $w_Z(x(k)) \in \mathbb{R}^{r_{q_Z} \times c_{q_Z}}$ is given by

$$w_Z(x(k)) = F_n - \bar{\mathcal{F}}\left(Z, \theta\right) = F_n - \bar{\mathcal{M}}_n(Z, \theta), \tag{3.25}$$

with the second equality due to (3.20). Moreover, letting $W_Z \geq 0$ denote an upper bound to $\|w_Z(x(k))\|_F$ for all k, i.e.,

$$\|w_Z(x(k))\|_F \leq W_Z,$$

since the columns of F_n are made of either columns or rows of $f\left(x(k)\right)$ (see (3.12) and (3.13) for reasons why), due to (2.3), (2.7), (3.25), (C.7), and (C.8), and because $w_Z(x(k)) \in \mathbb{R}^{r_{q_Z} \times c_{q_Z}}$, can be written as a summation of c_{q_Z} matrices of dimension r_{q_Z}-by-c_{q_Z}, with each matrix containing only one column of $w_Z(x(k))$ at a time and the rest of the elements being zero, W_Z is such that

$$\|w_Z(x(k))\|_F \leq \sum_{j=1}^{c_Z} \|w\left(x(\tau_j)\right)\|_F \leq c_Z W = W_Z. \tag{3.26}$$

Lastly, we also define the error $q_{Z,\epsilon}(k) \in \mathbb{R}^{c_Z \times c_\theta}$ as

$$q_{Z,\epsilon}(k) = \begin{cases} q_Z^\top(k), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ q_Z(k), & \text{if } \mathcal{M}_n = \mathcal{M}_2. \end{cases} \tag{3.27}$$

From, respectively, (3.23b) and (3.24), and regardless of the expressions of $\bar{\mathcal{M}}_n$ in (3.21), notice that the error $q_{Z,\epsilon}(k)$ in (3.27) is also

$$q_{Z,\epsilon}(k) = Z^\top \hat{\theta}(k) - F_n, \tag{3.28a}$$

$$= Z^\top \tilde{\theta}(k) - w_{Z,\epsilon}(k), \tag{3.28b}$$

where the matrix $F_n \in \mathbb{R}^{c_Z \times c_\theta}$ and the residual error $w_{Z,\epsilon}(k) \in \mathbb{R}^{c_Z \times c_\theta}$ are respectively given by

$$F_n = \begin{cases} F_1^\top, & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ F_2, & \text{if } \mathcal{M}_n = \mathcal{M}_2, \end{cases} \tag{3.29}$$

and

$$w_{Z,\epsilon}(k) = \begin{cases} w_Z^\top(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ w_Z(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_2. \end{cases} \tag{3.30}$$

CHAPTER IV

DISCRETE-TIME GRADIENT BASED CONCURRENT LEARNING

In the present chapter, we explore the fundamental method of Gradient Descent for discrete-time uncertainty approximation. Doing so, we only consider approximators that are linear in **all parameters** to identify. We study both the structured and unstructured uncertainty approximation cases. From the established foundations, we develop a Gradient Descent Based Concurrent Learning algorithm and show how Concurrent Learning can help achieve better parameter identification.

First, setting $\rho\left(x(k)\right) = \phi\left(x(k)\right)$ and $\Xi_m = I_{r_\phi} > 0$ in (2.18), in this chapter, we will make use of the signal

$$m^2(k) = \alpha I_{c_\phi} + \phi^\top(x(k))\, \phi\left(x(k)\right). \tag{4.1}$$

That means $m(k), m^2(k) \in \mathbb{R}^{c_\phi \times c_\phi}$. Now, consider the scalar cost function $J(k)$ defined as

$$\begin{aligned} J(k) &= \frac{1}{2}\text{tr}\left\{q_\epsilon^\top(k)\left(m^2(k)\right)^{-1} q_\epsilon(k)\right\}, && \text{(4.2a)}\\ &= \frac{1}{2}\text{tr}\left\{\Big(\phi^\top(x(k))\,\hat{\theta}(k) - f_n(k)\Big)^\top \left(m^2(k)\right)^{-1}\Big(\phi^\top(x(k))\,\hat{\theta}(k) - f_n(k)\Big)\right\}, && \text{(4.2b)}\end{aligned}$$

where (4.2b) comes to be because of the expression of q_ϵ in (2.15a). Recall that, according to Table 2.1 and the definition of q_ϵ in (2.14), if $\mathcal{M}_n = \mathcal{M}_1$, $q_\epsilon(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}} = \mathbb{R}^{c_f \times r_f}$, $c_\theta = r_f$, and $c_\phi = c_f$, whereas, when $\mathcal{M}_n = \mathcal{M}_2$, $q_\epsilon(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}} = \mathbb{R}^{r_f \times c_f}$, $c_\theta = c_f$, and $c_\phi = r_f$. That means, in any case, $r_{q_\epsilon} = c_\phi$ and $c_{q_\epsilon} = c_\theta$, and, subsequently, the right hand side of (4.2a) is dimensionally correct.

Now, for some scalar $\eta > 0$, the cost J can be recursively minimized with respect to $\hat{\theta}$ in a steepest descent direction using the update law

$$\hat{\theta}(k) = \hat{\theta}(k-1) - \eta \frac{\partial J(k-1)}{\partial \hat{\theta}(k-1)}$$

or, equivalently,

$$\hat{\theta}(k+1) = \hat{\theta}(k) - \eta \frac{\partial J(k)}{\partial \hat{\theta}(k)} \tag{4.3}$$

An illustration of the concept of Gradient Descent on Figure 4.1 shows a ball moving on the right side along the gradient of the cost J. Depending on its velocity each step of the way, it may be able to avoid falling into crevices and make it all the way to the bottom, where J is absolutely minimized. If that velocity is too high, though it may skip the crevices, it is possible that it gets over to left side and, undesirably, it may exit the enclosure altogether. Parameter η plays the role of the velocity of the ball or, at least, has control over the velocity at which the ball is moving. If η is appropriately set each step of the way then, it is likelier that one is able to guide the ball down the gradient of J without having it falling and dwelling into a crevice.

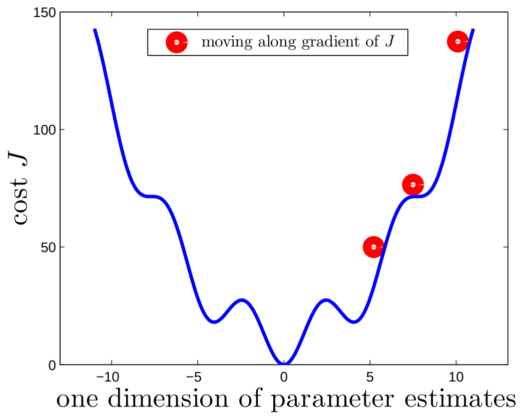


Figure 4.1: Gradient descent illustration

As far as function approximation goes, minimizing J leads to the reduction of the instantaneous approximation error q_ϵ or, equivalently, q (as (4.2a) hints to). The latter update law, i.e., (4.3), is actually what is commonly known as the discrete-time Normalized Gradient (DTNG) descent algorithm. From (4.3), given an initial parameter estimate matrix $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$, where $k_0 \in \mathbb{N}$ is an initial discrete-time step, for $k \geq k_0$, a more general DTNG update law can be formally given as

$$\hat{\theta}(k+1) = \hat{\theta}(k) - \Gamma \Delta \hat{\theta}_{NG}(k), \tag{4.4}$$

where $\Gamma \in \mathbb{R}^{r_\phi \times r_\phi}$, with $r_\phi = r_\theta$, is a positive definite, symmetric matrix ($\Gamma > 0$ and $\Gamma^\top = \Gamma$) and, by deriving the gradient $\dfrac{\partial J(k)}{\partial \hat{\theta}(k)}$ using (4.2b), (C.4), and recalling (2.15a), $\Delta \hat{\theta}_{NG}(k) \in \mathbb{R}^{r_\phi \times c_{q_\epsilon}}$, with $c_{q_\epsilon} = c_\theta$, is given by

$$\Delta \hat{\theta}_{NG}(k) = \phi\left(x(k)\right)\left(m^2(k)\right)^{-1} q_\epsilon(k), \tag{4.5}$$

with the error $q_\epsilon(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}}$ computed as in (2.14), i.e.,

$$q_\epsilon(k) = \begin{cases} q^\top(k), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ q(k), & \text{if } \mathcal{M}_n = \mathcal{M}_2. \end{cases}$$

4.1 Discrete-Time Normalized Gradient for both the Structured and Unstructured Uncertainty Approximation Cases

Consider the DTNG update rule of (4.4), from which, using (4.5), the parameter matrix error given by (2.10) can also be expressed as

$$\tilde{\theta}(k+1) = \tilde{\theta}(k) - \Gamma \phi\left(x(k)\right)\left(m^2(k)\right)^{-1} q_\epsilon(k). \tag{4.6}$$

Since $\Gamma > 0$, by setting

$$\Xi_V = \Xi_V^\top = \Gamma^{-1} = \left(\Gamma^{-1}\right)^\top > 0,$$

the Lyapunov function given by (2.24) becomes

$$V(k) = \text{tr}\left\{\tilde{\theta}^\top(k)\Gamma^{-1}\tilde{\theta}(k)\right\}. \tag{4.7}$$

With Γ being symmetric, Γ^{-1} also is. While letting $\Delta\theta_q(k) \in \mathbb{R}^{r_\theta \times c_\theta}$ ($c_\theta = c_{q_\epsilon}$) be defined as

$$\Delta\theta_q(k) = \phi\left(x(k)\right)\left(m^2(k)\right)^{-1} q_\epsilon(k), \tag{4.8}$$

along (4.6), or, essentially, $\tilde{\theta}(k+1) = \tilde{\theta}(k) - \Gamma\Delta\theta_q(k)$, we write, using (4.7), that

$$V(k+1) = \text{tr}\left\{\tilde{\theta}^\top(k)\Gamma^{-1}\tilde{\theta}(k) - \Upsilon_q - \Upsilon_q^\top + \Upsilon_{\Delta\theta_q}\right\},$$

where $\Upsilon_q \in \mathbb{R}^{c_\theta \times c_\theta}$ and $\Upsilon_{\Delta\theta_q} \in \mathbb{R}^{c_\theta \times c_\theta}$ are defined as

$$\Upsilon_q = \tilde{\theta}^\top(k)\Delta\theta_q(k), \tag{4.9a}$$

$$\Upsilon_{\Delta\theta_q} = \Delta\theta_q^\top(k)\Gamma\Delta\theta_q(k). \tag{4.9b}$$

Thus, given (2.26) and the definition of $V(k)$, i.e., (4.7),

$$\Delta V(k+1) = \text{tr}\left\{-\Upsilon_q - \Upsilon_q^\top + \Upsilon_{\Delta\theta_q}\right\}. \tag{4.10}$$

Let $\overline{q}(k) \in \mathbb{R}^{c_\phi \times c_\theta}$ ($c_\theta = c_{q_\epsilon}$) and $\overline{w}(k) \in \mathbb{R}^{c_\phi \times c_\theta}$ be such that

$$\overline{q}(k) = m^{-1}(k)q_\epsilon(k) \text{ and} \tag{4.11a}$$

$$\overline{w}(k) = m^{-1}(k)w_\epsilon(k). \tag{4.11b}$$

Also, given (4.11a) and (4.11b), let $\mu_{\overline{q},k} > 0$ and $\mu_{\overline{w},k} > 0$ be defined as

$$\mu_{\overline{q},k} = \left\|\overline{q}(k)\right\|_F = \left\|m^{-1}(k)q_\epsilon(k)\right\|_F \text{ and} \tag{4.12a}$$

$$\mu_{\overline{w},k} = \left\|\overline{w}(k)\right\| = \left\|m^{-1}(k)w_\epsilon(k)\right\|_F. \tag{4.12b}$$

Recalling (2.15b), we can write

$$\phi^\top(x(k))\,\tilde{\theta}(k) = q_\epsilon(k) + w_\epsilon(k).$$

or

$$\tilde{\theta}^\top(k)\phi\left(x(k)\right) = q_\epsilon^\top(k) + w_\epsilon^\top(k).$$

Hence, given (4.8), (4.11a), (4.11b), and because the signal $m^{-1}(k)$ is symmetric, Υ_q, defined above in (4.9a), is also

$$\begin{aligned}\Upsilon_q &= q_\epsilon^\top(k)\left(m^2(k)\right)^{-1} q_\epsilon(k) + w_\epsilon^\top(k)\left(m^2(k)\right)^{-1} q_\epsilon(k)\\ &= \left[m^{-1}(k)q_\epsilon(k)\right]^\top \left[m^{-1}(k)q_\epsilon(k)\right] + \left[m^{-1}(k)w_\epsilon(k)\right]^\top \left[m^{-1}(k)q_\epsilon(k)\right]\\ &= \overline{q}^\top(k)\overline{q}(k) + \overline{w}^\top(k)\overline{q}(k).\end{aligned}$$

By making use of (4.12a), (4.12b), (C.2), (C.8), and (C.9),

$$\text{tr}\left\{\Upsilon_q\right\} = \text{tr}\left\{\Upsilon_q^\top\right\} = \mu_{\overline{q},k}^2 + \text{tr}\left\{\overline{w}^\top(k)\overline{q}(k)\right\}. \tag{4.13}$$

Additionally, from (4.8), (4.9b), and (4.11a),

$$\begin{aligned}\Upsilon_{\Delta\theta_q} &= q_\epsilon^\top(k)\left(m^2(k)\right)^{-1}\phi^\top(x(k))\,\Gamma\phi\left(x(k)\right)\left(m^2(k)\right)^{-1} q_\epsilon(k)\\ &= \left[m^{-1}(k)q_\epsilon(k)\right]^\top m^{-1}(k)\phi^\top(x(k))\,\Gamma\phi\left(x(k)\right)m^{-1}(k)\left[m^{-1}(k)q_\epsilon(k)\right]\\ &= \overline{q}^\top(k)m^{-1}(k)\phi^\top(x(k))\,\Gamma\phi\left(x(k)\right)m^{-1}(k)\overline{q}(k).\end{aligned}$$

We define the scalar

$$\eta = \lambda_{\max}(\Gamma)\,, \tag{4.14}$$

meaning $\Gamma \leq \eta I_{r_\phi} = \eta I_{r_\theta}$ since $r_\theta = r_\phi$. With $\eta > 0$ given that $\Gamma > 0$,

$$\Upsilon_{\Delta\theta_q} \leq \eta\,\overline{q}^\top(k)m^{-1}(k)\phi^\top(x(k))\,\phi\left(x(k)\right)m^{-1}(k)\overline{q}(k).$$

Hence, from (C.16) and (4.12a),

$$\text{tr}\left\{\Upsilon_{\Delta\theta_q}\right\} \leq \eta\left\|\phi\left(x(k)\right)\right\|_F^2\left\|m^{-1}(k)\right\|_F^2\mu_{\overline{q}}^2. \tag{4.15}$$

Moreover, given (4.12a) and (4.12b), for all k,

$$\text{tr}\left\{\overline{w}^{\top}(k)\overline{q}(k)\right\} \geq -\left\|\overline{q}(k)\right\|_F \left\|\overline{w}(k)\right\|_F = -\mu_{\overline{q},k}\mu_{\overline{w},k}. \tag{4.16}$$

With $\eta > 0$, based on (C.2), (4.13), (4.15), and (4.16), we get from (4.10) that

$$\begin{aligned}
\Delta V(k+1) &= -2\text{tr}\left\{\Upsilon_q\right\} + \text{tr}\left\{\Upsilon_{\Delta\theta_q}\right\} \\
&\leq -2\mu_{\overline{q}}^2 - 2\text{tr}\left\{\overline{w}^{\top}(k)\overline{q}(k)\right\} + \eta\left\|\phi\left(x(k)\right)\right\|_F^2\left\|m^{-1}(k)\right\|_F^2\mu_{\overline{q}}^2 \\
&\leq -2\mu_{\overline{q}}^2 + 2\mu_{\overline{q}}\mu_{\overline{w},k} + \eta\left\|\phi\left(x(k)\right)\right\|_F^2\left\|m^{-1}(k)\right\|_F^2\mu_{\overline{q}}^2 = Q_E,
\end{aligned} \tag{4.17}$$

where, letting the scalar

$$\beta_u(k) = 2 - \eta\left\|\phi\left(x(k)\right)\right\|_F^2\left\|m^{-1}(k)\right\|_F^2, \tag{4.18}$$

the quadratic expression Q_E in $\mu_{\overline{q},k}$ and $\mu_{\overline{w},k}$ is defined as

$$Q_E = -\beta_u(k)\mu_{\overline{q},k}^2 + 2\mu_{\overline{q}}\mu_{\overline{w},k} = -\mu_{\overline{q},k}\left(\beta_u(k)\mu_{\overline{q},k} - 2\mu_{\overline{w},k}\right). \tag{4.19}$$

Notice that, for all k, since $\eta > 0$, $m^2(k) > 0$, and $\left\|\phi\left(x(k)\right)\right\|_F^2 \geq 0$, then $\beta_u(k) < 2$.

Given (2.25) and bearing in mind that $Q_E \leq 0$ implies

$$\underline{\beta}_V\left\|\tilde{\theta}(k+1)\right\|_F^2 - \overline{\beta}_V\left\|\tilde{\theta}(k)\right\|_F^2 \leq \Delta V(k+1) \leq Q_E \leq 0,$$

with, by the way, $0 < \underline{\beta}_V < \overline{\beta}_V$, which therefore leads to stability results, we are now interested in determining the sign of Q_E as a function of the design parameter η. Recall that $\mu_{\overline{q},k} = \left\|\overline{q}(k)\right\|_F \geq 0$ for all k. We have from (4.19) that $Q_E = 0$ if $\mu_{\overline{q},k} = 0$ or, because of (2.14), (4.11a), and given that $m^{-1}(k) > 0$, effectively, $q_\epsilon(k) = [0]^{r_{q_\epsilon}\times c_{q_\epsilon}}$ or $q(k) = [0]^{r_f\times c_f}$, which cannot always be expected. Hence, we will investigate the case when $\mu_{\overline{q},k} > 0$.

4.1.1 DTNG for the SUAC

Assuming f is a structured uncertainty, then, as pointed out before, we can set $w\left(x(k)\right) = [0]^{r_f \times c_f}$ and, given (2.17), $w_\epsilon(k) = [0]^{r_{q_\epsilon} \times c_{q_\epsilon}}$ for all k. Also, from (2.1) and (2.3), essentially,

$$f\left(x(k)\right) = \mathcal{F}\left(x(k), \theta\right) = \mathcal{M}_n(x(k), \theta)\,, \tag{4.20}$$

and, using (2.13) and (2.15b),

$$q(k) = \mathcal{M}_n\Big(x(k), \tilde{\theta}(k)\Big) \tag{4.21}$$

and

$$q_\epsilon(k) = \phi^\top(x(k))\,\tilde{\theta}(k). \tag{4.22}$$

Moreover, as a result of having $w_\epsilon(k) = [0]^{r_{q_\epsilon} \times c_{q_\epsilon}}$ for all k, according to (4.12b), $\mu_{\overline{w},k} = \|\overline{w}(k)\|_F = 0$. Given (4.19), that means

$$\Delta V(k+1) \leq Q_E = -\beta_u(k)\mu^2_{\overline{q},k}, \tag{4.23}$$

with $Q_E = -\beta_u(k)\mu^2_{\overline{q},k} \leq 0$ if $\beta_u(k) > 0$, which, given (4.18), means picking $\eta = \lambda_{\max}(\Gamma)$ and, subsequently, designing Γ such that

$$0 < \eta < \frac{2}{\|\phi\left(x(k)\right)\|_F^2\,\|m^{-1}(k)\|_F^2} = \overline{\eta}_{NG}(k). \tag{4.24}$$

In the case that $\phi\left(x(k)\right) = [0]^{r_\phi \times c_\phi}$, $\overline{\eta}_{NG}(k)$ becomes undefined. However, when that happens, there is no need picking η and designing for Γ as, for any $\Gamma > 0$, (4.4) reduces in that case to $\hat{\theta}(k+1) = \hat{\theta}(k)$ since, from (4.5) $\Delta\hat{\theta}_{NG}(k) = [0]^{r_\theta \times c_\theta}$. Moreover, with η picked according to (4.24), notice that $0 < \beta_u(k) < 2$.

In all, $\Delta V(k+1) \leq Q_E \leq 0$ implies boundedness of $V(k)$ and $\tilde{\theta}(k)$. In fact, from (2.25), (2.26), and (4.7), $\Delta V(k+1) = V(k+1) - V(k) \leq 0$ means that $V(k+1) \leq V(k)$ and, consequently, for all k, $V(k) \leq V(k_0)$. Thus,

$$\underline{\beta}_V \left\|\tilde{\theta}(k)\right\|_F^2 \leq V(k) \leq V(k_0) \leq \overline{\beta}_V \left\|\tilde{\theta}(k_0)\right\|_F^2,$$

implying, for all k,

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{NG_s} = \sqrt{\frac{\overline{\beta}_V}{\underline{\beta}_V}} \left\|\tilde{\theta}(k_0)\right\|_F \tag{4.25}$$

We could have actually gotten a larger upper bound of η by going back to (4.6) and rewriting it, while using $q_\epsilon(k)$ in (4.22), as

$$\tilde{\theta}(k+1) = \left[I_{r_\theta} - \Gamma\Phi_{m,k}\right]\tilde{\theta}(k), \tag{4.26}$$

where

$$\Phi_{m,k} = \phi\left(x(k)\right)\left(m^2(k)\right)^{-1}\phi^\top(x(k)). \tag{4.27}$$

Notice that $\Phi_{m,k}$ is symmetric, i.e., $\Phi_{m,k}^\top = \Phi_{m,k}$, as $\left(m^2(k)\right)^{-1}$ also is. By evaluating the Lyapunov function (4.7) along (4.26) and using (2.26),

$$\Delta V(k+1) = \operatorname{tr}\left\{\tilde{\theta}^\top(k)\overline{Q}_{NG,k}\tilde{\theta}(k)\right\}, \tag{4.28}$$

where,

$$\overline{Q}_{NG,k} = -\left(2I_{r_\theta} - \Phi_{m,k}\Gamma\right)\Phi_{m,k}. \tag{4.29}$$

From (2.22) and (4.27), we can write that

$$\Phi_{m,k} \leq \left\|m^{-1}(k)\right\|_F^2 \phi\left(x(k)\right) I_{c_\phi}\phi^\top(x(k)) = \left\|m^{-1}(k)\right\|_F^2 \Phi_k,$$

with Φ_k defined in (2.27). Furthermore, given (2.28), $r_\theta = r_\phi$, and $\eta = \lambda_{\max}(\Gamma)$,

$$\Phi_{m,k} \leq \overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2 I_{r_\phi} = \overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2 I_{r_\theta} \tag{4.30}$$

and

$$2I_{r_\theta} - \Phi_{m,k}\Gamma \geq \left(2 - \eta\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2\right) I_{r_\theta}$$

Consequently, for the scalar

$$\beta_s(k) = 2 - \eta\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2, \tag{4.31}$$

$\overline{Q}_{NG,k}$ in (4.71) can be upper bounded as

$$\overline{Q}_{NG,k} \leq -\beta_s(k)\Phi_{m,k}.$$

As a result, given (4.12a), (4.22), (4.27), and (4.28),

$$\begin{aligned}\Delta V(k+1) &\leq -\beta_s(k)\text{tr}\left\{\tilde{\theta}^\top(k-1)\phi\left(x(k)\right)\left(m^2(k)\right)^{-1}\phi^\top(x(k))\,\tilde{\theta}(k-1)\right\}\\ &\leq -\beta_s(k)\text{tr}\left\{q_\epsilon^\top(k)\left(m^2(k)\right)^{-1}q_\epsilon(k)\right\}\\ &\leq -\beta_s(k)\text{tr}\left\{\left[m^{-1}(k)q_\epsilon(k)\right]^\top\left[m^{-1}(k)q_\epsilon(k)\right]\right\}\\ &\leq -\beta_s(k)\mu_{\overline{q},k}^2 \leq 0 \end{aligned} \tag{4.32}$$

if $\beta_s(k) > 0$, which from (4.31), means

$$0 < \eta < \frac{2}{\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2} = \overline{\eta}_{NG_s}(k). \tag{4.33}$$

We formally show in Remark 4.1.3 why, given in (4.24) and (4.33), $\overline{\eta}_{NG}(k) \leq \overline{\eta}_{NG_s}(k)$.

Consider the following remarks.

Remark 4.1.1. Parameter η can vary as k evolves, i.e., $\eta = \eta(k)$, so long as (4.24) (or (4.33)) is met. Hence, Γ, which is such that $\lambda_{\max}(\Gamma) = \eta$, can be set to vary with time also. Moreover, let $\lambda_{\{\Phi_k\}_r}$, $r = 1, 2, \ldots, r_\phi$, denote the eigenvalues of Φ_k.

Remark 4.1.2. Both $\overline{\eta}_{NG_s}(k)$ and $\overline{\eta}_{NG}(k)$ become undefined if $\phi\left(x(k)\right) = [0]^{r_\phi \times c_\phi}$, as, in that case, $\left\|\phi\left(x(k)\right)\right\|_F$ and $\overline{\lambda}_{\Phi_k}$ in, respectively, (4.24) and (4.33) are exactly zero. However, when that is the case, η and Γ need not to be designed for, as (4.4) reduces to $\theta(k+1) = \theta(k)$.

Remark 4.1.3. Because, for all k,

$$\overline{\lambda}_{\Phi_k} = \lambda_{\max}(\Phi_k) = \max_r \lambda_{\{\Phi_k\}_r} \leq \sum_{r=1}^{r_\phi} \lambda_{\{\Phi_k\}_r}$$

and, using (2.27), (C.1), (C.8), and (C.9),

$$\sum_{r=1}^{r_\phi} \lambda_{\{\Phi_k\}_r} = \mathrm{tr}\,(\Phi_k) = \|\phi\,(x(k))\|_F^2$$

then $\overline{\lambda}_{\Phi_k} \leq \|\phi\,(x(k))\|_F^2$. This therefore proves why $\overline{\eta}_{NG}(k)$ in (4.24) and $\overline{\eta}_{NG_s}(k)$ in (4.33) are such that $\overline{\eta}_{NG}(k) \leq \overline{\eta}_{NG_s}(k)$.

Remark 4.1.4. Because we are working here with a dimensionally complicated approximation problem, the upper bound $\overline{\eta}_{NG_s}(k)$ on η in (4.33) may seem different from the one we previously found in [28] (and used in [29]). However the upper bound $\overline{\eta}_{NG_s}(k)$ of η is only written here in a more general form. Notice that, if the normalization signal $m(k)$ is scalar, then (4.24) and (4.33) can be rewritten as

$$0 < \eta < \frac{2m^2(k)}{\|\phi\,(x(k))\|_F^2} \tag{4.34}$$

and

$$0 < \eta < \frac{2m^2(k)}{\overline{\lambda}_{\Phi_k}}, \tag{4.35}$$

as is the case in [28, 29]. Moreover, notice that if $m(k)$ is scalar, then $\phi\,(x(k))$ is a column vector, thus, $c_\phi = 1$, and $m^2(k)$ in (4.1) becomes

$$m^2(k) = \alpha + \phi^\top(x(k))\,\phi\,(x(k)) = \alpha + \|\phi\,(x(k))\|_F^2 = \alpha + \|\phi\,(x(k))\|^2\,,$$

which consequently means that

$$m^2(k) > \|\phi\,(x(k))\|_F^2 = \|\phi\,(x(k))\|^2\,. \tag{4.36}$$

Hence, if for all k, $\overline{\lambda}_{\Phi_k} > 0$, i.e., $\phi\,(x(k)) \neq [0]^{r_\phi \times c_\phi}$, or, with $c_\phi = 1$, $\phi\,(x(k)) \neq [0]^{r_\phi}$,

$$1 < \frac{m^2(k)}{\|\phi\,(x(k))\|^2} \leq \frac{m^2(k)}{\overline{\lambda}_{\Phi_k}}.$$

and, again, given (4.24) and (4.33) or rather (4.34) and (4.35), $\overline{\eta}_{NG_s}(k) \geq \overline{\eta}_{NG}(k) > 2$. The upper bounds $\overline{\eta}_{NG_s}(k)$ and $\overline{\eta}_{NG}(k)$ can indeed be chosen to be constant in general, precisely $\overline{\eta}_{NG_s}(k) = \overline{\eta}_{NG}(k) = 2$ as done in [2, 3] when dealing with a scalar normalization signal. However, we elect not to do so here to allow for less restraining upper bounds of η.

Remark 4.1.5. Using (2.21) and because we have, in this chapter, substituted $\rho\left(x(k)\right)$ by $\phi\left(x(k)\right)$ in the expression of the $m^2(k)$ in (2.18), we have that

$$\overline{\eta}_{NG_s}(k) = \frac{2}{\overline{\lambda}_{\Phi_k} \left\|m^{-1}(k)\right\|_F^2} \leq \frac{2\overline{\lambda}_{m^2,k}}{\alpha c_\rho \overline{\lambda}_{\Phi_k}}, \tag{4.37}$$

where the strictly positive scalar $\overline{\lambda}_{m^2,k}$ is defined in (2.20). If $\phi\left(x(k)\right) \neq [0]^{r_\phi \times c_\phi}$, then $\overline{\lambda}_{\Phi_k} > 0$. We can therefore say from (4.37) that, for all k, if $\phi\left(x(k)\right) \neq [0]^{r_\phi \times c_\phi}$, for which we have to design for Γ as pointed out in Remark 4.1.2, $\overline{\eta}_{NG_s}(k)$ is bounded away from infinity as long as $\alpha \neq 0$ and $c_\rho < \infty$ (which should be the case in practice). With $\overline{\eta}_{NG}(k) \leq \overline{\eta}_{NG_s}(k)$ as we have shown in Remark 4.1.3, that also means $\overline{\eta}_{NG}(k)$ being bounded away from infinity.

In summary, when dealing with structured uncertainties, we are guaranteed to have

$$\Delta V(k+1) \leq Q_E \leq 0$$

provided η is picked according to (4.24) or (4.33). By doing so, $V(k)$ is nonincreasing and, as a result, $\tilde{\theta}(k)$ remains bounded for all k.

Proceeding, the following properties can be obtained. First, given (4.1), (C.8), (C.10), and (C.11), there exists some scalar $\overline{\alpha}_{m,1}$ such that $0 < \overline{\alpha}_{m,1} \leq 1$ and

$$\left\|m(k)\right\|_F \geq \overline{\alpha}_{m,1} \left(\alpha \left\|I_{c_\phi}\right\|_F + \left\|\phi\left(x(k)\right)\right\|_F^2\right)^{\frac{1}{2}}. \tag{4.38}$$

Thus, from (2.14), (4.22), (4.38), (C.8), (C.10), and (C.11),

$$\frac{\left\|q(k)\right\|_F}{\left\|m(k)\right\|_F} = \frac{\left\|q_\epsilon(k)\right\|_F}{\left\|m(k)\right\|_F} \leq \frac{\left\|\tilde{\theta}(k)\right\|_F \left\|\phi\left(x(k)\right)\right\|_F}{\overline{\alpha}_{m,1} \left(\alpha \left\|I_{c_\psi}\right\|_F + \left\|\phi\left(x(k)\right)\right\|_F^2\right)^{\frac{1}{2}}} \tag{4.39}$$

and, using (2.23) to define $\phi_m(x(k)) \in \mathbb{R}^{r_\phi \times c_\phi}$, i.e.,

$$\phi_m(x(k)) = \phi\left(x(k)\right) m^{-1}(k), \tag{4.40}$$

(4.12a), and (C.13), for some scalars β_{q_ϵ} and $\beta_{\phi,1}$, such that $1 < \beta_{q_\epsilon} < \infty$ and $1 < \beta_{\phi,1} < \infty$,

$$\mu_{\overline{q},k} = \left\|\overline{q}(k)\right\|_F = \left\|m^{-1}(k)q_\epsilon(k)\right\|_F \leq \left\|m^{-1}(k)\right\|_F \left\|q_\epsilon(k)\right\|_F < \beta_{q_\epsilon} \left\|I_{c_\phi}\right\|_F \frac{\left\|q_\epsilon(k)\right\|_F}{\left\|m(k)\right\|_F} \tag{4.41}$$

and

$$\begin{aligned}\left\|\phi_m(x(k))\right\|_F = \left\|\phi\left(x(k)\right) m^{-1}(k)\right\|_F \leq \left\|\phi\left(x(k)\right)\right\|_F \left\|m^{-1}(k)\right\|_F < \beta_{\phi,1} \left\|I_{c_\phi}\right\|_F \frac{\left\|\phi(k)\right\|_F}{\left\|m(k)\right\|_F} \\ < \beta_{\phi,1} \left\|I_{c_\phi}\right\|_F \frac{\left\|\phi\left(x(k)\right)\right\|_F}{\overline{\alpha}_{m,1}\left(\alpha \left\|I_{c_\psi}\right\|_F + \left\|\phi\left(x(k)\right)\right\|_F^2\right)^{\frac{1}{2}}}.\end{aligned} \tag{4.42}$$

Hence, with $\tilde{\theta}(k)$ being bounded, given (4.39), $\dfrac{\left\|q(k)\right\|_F}{\left\|m(k)\right\|_F}$ is also bounded and, from (4.41) and (4.42), so are

$$\mu_{\overline{q},k} = \left\|\overline{q}(k)\right\|_F = \left\|m^{-1}(k)q_\epsilon(k)\right\|_F \text{ and } \left\|\phi_m(x(k))\right\|_F = \left\|\phi\left(x(k)\right) m^{-1}(k)\right\|_F.$$

Nevertheless, the previously mentioned boundedness results are dependent on $\overline{\alpha}_{m,1}$. Based on (4.1) and (4.38), we can say that there exists at least an instance of $\overline{\alpha}_{m,1}$ that is much closer to 1 than 0 and for which the aforementioned boundedness results stand. Additionally, we get from (2.26), (4.12a), and (4.32) that

$$V(k+1) = V(k_0) - \sum_{\tau=k_0}^{k} \beta_s(\tau)\mu_{\overline{q},\tau}^2 = V(k_0) - \sum_{\tau=k_0}^{k} \beta_s(\tau) \left\|m^{-1}(\tau)q_\epsilon(\tau)\right\|_F^2,$$

meaning

$$\sum_{\tau=k_0}^{k} \beta_s(\tau) \left\|m^{-1}(\tau)q_\epsilon(\tau)\right\|_F^2 = V(k_0) - V(k+1) \leq V(k_0) = \operatorname{tr}\left\{\tilde{\theta}^\top(k_0)\Gamma^{-1}\tilde{\theta}(k_0)\right\},$$

given that, for all k, $V(k)$, which is defined in (4.7), is nonincreasing,

$$\sum_{\tau=k_0}^{\infty} \beta_s(\tau) \left\|m^{-1}(\tau)q_\epsilon(\tau)\right\|_F^2 = \lim_{k\to\infty} \sum_{\tau=k_0}^{\infty} \beta_s(\tau) \left\|m^{-1}(\tau)q_\epsilon(\tau)\right\|_F^2 \leq V(k_0),$$

and, subsequently, $\left\|m^{-1}(k)q_\epsilon(k)\right\|_F \in \mathcal{L}^2$ or, given (C.9), vec $\left\{m^{-1}(k)q_\epsilon(k)\right\} \in \mathcal{L}^2$. Notice that we could also have used (4.23) to get the same results. For the definitions of $\mathcal{L}^p$, $p \in [1, \infty)$, and $\mathcal{L}^\infty$ signal spaces as well as properties pertaining to those spaces, consult Appendix C. From (4.6), the fact that $\eta = \lambda_{\max}(\Gamma)$, meaning, for $\Gamma \in \mathbb{R}^{r_\phi \times r_\phi}$, $\Gamma \leq \eta I_{r_\phi}$, and given $r_\phi = r_\theta$, we can also bound

$$\Delta\tilde{\theta}(k+1) = \tilde{\theta}(k+1) - \tilde{\theta}(k) \tag{4.43}$$

as

$$\begin{aligned}\left\|\Delta\tilde{\theta}(k+1)\right\|_F &= \left\|\Gamma\phi\left(x(k)\right)\left(m^2(k)\right)^{-1} q_\epsilon(k)\right\|_F = \left\|\Gamma\left[\phi\left(x(k)\right)m^{-1}(k)\right]\left[m^{-1}(k)q_\epsilon(k)\right]\right\|_F \\ &\leq \|\Gamma\|_F \left\|\phi\left(x(k)\right)m^{-1}(k)\right\|_F \left\|m^{-1}(k)q_\epsilon(k)\right\|_F \\ &\leq \eta \left\|I_{r_\phi}\right\|_F \left\|\phi\left(x(k)\right)m^{-1}(k)\right\|_F \left\|m^{-1}(k)q_\epsilon(k)\right\|_F. \end{aligned} \tag{4.44}$$

With η being bounded away from infinity, $\left\|\phi\left(x(k)\right)m^{-1}(k)\right\|_F$ bounded (as shown by (4.42)), and $\left\|m^{-1}(k)q_\epsilon(k)\right\|_F \in \mathcal{L}^2$, (4.44) means that, aside from being bounded, $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$, or, written differently, from (C.9), vec $\left\{\Delta\tilde{\theta}(k)\right\} \in \mathcal{L}^2$.

4.1.2 DTNG for the UUAC

When considering the unstructured uncertainty case, we can no longer assume that $w\left(x(k)\right) = [0]^{r_f \times c_f}$ and/or $w_\epsilon(k) = [0]^{r_{q_\epsilon} \times c_{q_\epsilon}}$. From (4.19) and (4.23), we have that

$$\Delta V(k+1) \leq Q_E = -\beta_u(k)\mu_{\overline{q},k}^2 + 2\mu_{\overline{q}}\mu_{\overline{w},k} = -\mu_{\overline{q},k}\left(\beta_u(k)\mu_{\overline{q},k} - 2\mu_{\overline{w},k}\right). \tag{4.45}$$

For $\mu_{\overline{q},k} \geq 0$ and $\mu_{\overline{w},k} \geq 0$ for all k, $Q_E \leq 0$ (which, as mentioned before, implies stability of $\tilde{\theta}$) if

$$\beta_u(k)\mu_{\overline{q},k} - 2\mu_{\overline{w},k} \geq 0, \tag{4.46}$$

which in turns can only be possible if $\beta_u(k) > 0$ or, consequently, η is picked such that $0 < \eta < \overline{\eta}_{NG}(k)$, i.e., (4.24) is verified. Consider the following analysis.

- It is worth wondering if we can ensure that condition (4.46) holds by adequately designing for η? Recall that $Q_E = 0$ if $\mu_{\overline{q},k} = \left\| m^{-1}(k) q_\epsilon(k) \right\|_F = 0$, which implies that $q_\epsilon(k) = [0]^{r_{q_\epsilon} \times c_{q_\epsilon}}$, and, just like in the case when $\phi\left(x(k)\right) = [0]^{r_\phi \times c_\phi}$, picking η is of no use as, from (4.5), $\Delta\hat{\theta}_{NG}(k) = [0]^{r_\theta \times c_\theta}$ and, as a result, (4.4) simply becomes $\hat{\theta}(k+1) = \hat{\theta}(k)$. For $\mu_{\overline{q},k} > 0$, because of (4.18), (4.46) leads to

$$\beta_u(k) = 2 - \eta \left\| \phi\left(x(k)\right) \right\|_F^2 \left\| m^{-1}(k) \right\|_F^2 \geq \frac{2\mu_{\overline{w},k}}{\mu_{\overline{q},k}}.$$

Noticing that $\frac{2\mu_{\overline{w},k}}{\mu_{\overline{q},k}} = 0$ is a possibility (essentially when $\mu_w = 0$), to make sure $\beta_u(k)$ is strictly greater tan zero, i.e., $\beta_u(k) > 0$, we need

$$\beta_u(k) = 2 - \eta \left\| \phi\left(x(k)\right) \right\|_F^2 \left\| m^{-1}(k) \right\|_F^2 > \frac{2\mu_{\overline{w},k}}{\mu_{\overline{q},k}}$$

instead. That is

$$0 < \eta < \frac{\mu_{\overline{q},k} - \mu_{\overline{w},k}}{\mu_{\overline{q},k}} \left[\frac{2}{\left\| \phi\left(x(k)\right) \right\|_F^2 \left\| m^{-1}(k) \right\|_F^2} \right] = \frac{\mu_{\overline{q},k} - \mu_{\overline{w},k}}{\mu_{\overline{q},k}} \overline{\eta}_{NG}(k), \tag{4.47}$$

given the definition of $\overline{\eta}_{NG}(k)$ in (4.24). Moving along, we can find some constant scalar $\overline{W} \geq 0$ such that

$$\mu_{\overline{w},k} = \left\| \overline{w}(k) \right\|_F = \left\| m^{-1}(k) w_\epsilon(k) \right\|_F \leq \overline{W} \tag{4.48}$$

for all $k > k_0$. In fact, going back to (2.19) we can write that

$$m^{-1}(k) \leq \frac{1}{\sqrt{\alpha}} I_{c_\phi} \text{ and } \left\| m^{-1}(k) \right\|_F \leq \frac{1}{\sqrt{\alpha}} \left\| I_{c_\phi} \right\|_F = \frac{\sqrt{\alpha\, c_\phi}}{\alpha}. \tag{4.49}$$

Hence, given the $\mathcal{L}^\infty$ property of $\left\| w\left(x(k)\right) \right\|_F$, i.e., (2.7), and considering (2.17), (4.48), (4.49), (C.8), and (C.11),

$$\begin{aligned} \mu_{\overline{w},k} = \left\| \overline{w}(k) \right\|_F = \left\| m^{-1}(k) w_\epsilon(k) \right\|_F &\leq \left\| m^{-1}(k) \right\|_F \left\| w_\epsilon(k) \right\|_F \\ &\leq \left\| m^{-1}(k) \right\|_F \left\| w\left(x(k)\right) \right\|_F \leq \frac{\sqrt{\alpha\, c_\phi}}{\alpha} W = \overline{W}. \end{aligned} \tag{4.50}$$

Thus, picking up from (4.47), as a more general rule, η can be picked to verify

$$0 < \eta < \overline{\eta}_{NG_0}(k) = \frac{\mu_{\overline{q},k} - \overline{W}_0}{\mu_{\overline{q},k}} \overline{\eta}_{NG}(k), \tag{4.51}$$

where, for some upper bound $\overline{W}_0 \geq \overline{W} \geq 0$, since $\mu_{\overline{w},k} \leq \overline{W}$,

$$\mu_{\overline{q},k} - \overline{W}_0 \leq \mu_{\overline{q},k} - W \leq \mu_{\overline{q},k} - \mu_{\overline{w},k}.$$

Noticing that, for any $k > k_0$,

$$\frac{\mu_{\overline{q},k} - \overline{W}_0}{\mu_{\overline{q},k}} \leq 1,$$

since $\overline{W}_0 \geq 0$, if η is picked such that (4.51) is met then (4.24) will also be verified. As a side note, remark that $\overline{\eta}_{NG_0}(k) = \overline{\eta}_{NG}(k)$ if, by design, $\overline{W}_0 = 0$. However, an $\overline{W}_0 = 0$ assumes $\overline{W} = 0$, given that $\overline{W}_0 \geq \overline{W}$, and, effectively, that there is no representation error $w\left(x(k)\right)$, i.e., $w\left(x(k)\right) = [0]^{r_f \times c_f}$. Hence, setting $\overline{W}_0 = 0$ would only be a good idea if dealing with the SUAC. Moving on, as hinted to, satisfying (4.51) can only be feasible if $\mu_{\overline{q},k} > \overline{W}_0$. If that is in fact the case then

$$\beta_u(k) > \frac{2\mu_{\overline{w},k}}{\mu_{\overline{q},k}} \text{ or } \beta_u(k)\mu_{\overline{q},k} - 2\mu_{\overline{w},k} > 0,$$

(4.46) is verified, and

$$\Delta V(k+1) = V(k+1) - V(k) \leq Q_E < 0.$$

It should nevertheless be noted that (4.51) requires knowledge of $\overline{W}_0$ for possible implementation and, in practice, it may be difficult to acquire such $\overline{W}_0$. Moreover, even if we could find a viable $\overline{W}_0$, the question still remain as to what happens when $\mu_{\overline{q},k} = \|\overline{q}(k)\|_F \leq \overline{W}_0$ and (4.51) is no longer realizable. Again if we know for a fact that $\overline{W} = 0$ (SUAC), then we need not to worry about the representation error and can proceed to apply the DTNG algorithm as is. Otherwise, in the event $\overline{W} > 0$, (4.51) is no longer valid for values of

$\mu_{\overline{q},k} = \|\overline{q}(k)\|_F \leq \overline{W}_0$ and, therefore, picking η for which $\beta_u(k) > \dfrac{2\mu_{\overline{w},k}}{\mu_{\overline{q},k}}$ cannot be achieved. This analysis can however become useful if we were to implement a *dead-zone modification* [30].

- We thus go back to studying the sign of Q_E and look more closely to what happens when $\mu_{\overline{q},k} > 0$. Remark that if the approximation error $q_\epsilon(k)$ is such that

$$\mu_{\overline{q},k} = \|\overline{q}(k)\|_F = \left\|m^{-1}(k)q_\epsilon(k)\right\|_F > \underline{\varepsilon}_{\overline{q},k}$$

with $\underline{\varepsilon}_{\overline{q},k} = \dfrac{2\mu_{\overline{w},k}}{\beta_u(k)}$, then (4.46) is verified, $\Delta V(k+1) \leq Q_E < 0$ and, consequently, both $V(k)$ and $\left\|\tilde{\theta}(k)\right\|_F$ are bounded. However, when $\mu_{\overline{q},k} \leq \underline{\varepsilon}_{\overline{q},k}$ then

$$\beta_u(k)\mu_{\overline{q},k} - 2\mu_{\overline{w},k} \leq 0,$$

$Q_E \geq 0$, $\Delta V(k+1)$ can be positive, which would lead to $\left\|\tilde{\theta}(k)\right\|_F$ possibly growing unbounded. We can say a few things from this.

1. First, because of (4.48) and since, by picking η to verify (4.24), $\beta_u(k) > 0$ for all $k \geq k_0$, we can find an upper-bound to $\underline{\varepsilon}_{\overline{q},k} = \dfrac{2\mu_{\overline{w},k}}{\beta_u(k)}$. In fact, for some constant scalar $\underline{\beta}_u$ such that, for all $k \geq k_0$, $0 < \underline{\beta}_u \leq \beta_u(k) < 2$, we have

$$\underline{\varepsilon}_{\overline{q},k} = \frac{2\mu_{\overline{w},k}}{\beta_u(k)} \leq \frac{2\overline{W}}{\underline{\beta}_u}.$$

Even though $\Delta V(k+1) < 0$ when $\mu_{\overline{q},k} = \|\overline{q}(k)\|_F > \underline{\varepsilon}_{\overline{q},k}$ (or $\mu_{\overline{q},k} > \dfrac{2\overline{W}}{\underline{\beta}_u}$), we cannot claim ultimate boundedness of $\|\overline{q}(k)\|_F = \left\|m^{-1}(k)q_\epsilon(k)\right\|_F$ or $\|q_\epsilon(k)\|_F$ or, according to (2.14), $\|q(k)\|_F$ in view of the fact that the Lyapunov function $V(k)$ is defined in terms of $\tilde{\theta}$ and not $\overline{q}$ or q_ϵ or q. An equivalent argument can be made by recalling from (2.15b) that

$$q_\epsilon(k) = \phi^\top(x(k))\,\tilde{\theta}(k) - w_\epsilon(k),$$

and, for $\mu_{\overline{q},k} \leq \varepsilon_{\overline{q},k}$ and $\Delta V(k+1) > 0$, $\left\|\tilde{\theta}(k)\right\|_F$ growing large will cause $\|q_\epsilon(k)\|_F$ to also become large. As a result, claiming that $\|q_\epsilon\|_F$ or $\|q\|_F$ are ultimately bounded is not accurate.

2. Second, boundedness of q_ϵ, even if possible, does not directly translate into that of $\tilde{\theta}$. According to (2.15b), (C.8), (C.10), and (C.11),

$$\|q_\epsilon(k)\|_F \leq \|\phi(x(k))\|_F \left\|\tilde{\theta}(k)\right\|_F + \|w_\epsilon(k)\|_F \text{ or } \mu_{q_\epsilon,k} \leq \mu_{\phi,k}\, \mu_{\tilde{\theta},k} + \mu_{w_\epsilon,k}$$

   where we define $\mu_{q_\epsilon,k} = \|q_\epsilon(k)\|_F$, $\mu_{\phi,k} = \|\phi(x(k))\|_F$, $\mu_{\tilde{\theta},k} = \left\|\tilde{\theta}(k)\right\|_F$, and $\mu_{w_\epsilon,k} = \|w_\epsilon(k)\|_F$. For $\mu_{q_\epsilon,k} \neq 0$, solving for $\mu_{\tilde{\theta},k}$ yields

$$\mu_{\tilde{\theta},k} \geq \frac{\mu_{q_\epsilon,k} - \mu_{w_\epsilon,k}}{\mu_{\phi,k}}.$$

   Thus, analytically speaking, a bounded $\mu_{q_\epsilon,k}$ does not necessarily imply a bounded $\mu_{\tilde{\theta},k}$. Actually, even for a small $\mu_{q_\epsilon,k} = \|q_\epsilon(k)\|_F$ (which also means a small $\|q(k)\|_F$) such that $\mu_{\overline{q},k} = \left\|m^{-1}(k)q_\epsilon(k)\right\|_F \leq \varepsilon_{\overline{q},k}$ and $\Delta V(k+1) > 0$ for the remainder of the time, $\mu_{\tilde{\theta},k} = \left\|\tilde{\theta}(k)\right\|_F$ will keep growing indefinitely. When that happens, considering θ is bounded, it implies $\tilde{\theta}(k) = \hat{\theta}(k) - \theta$ and, subsequently, $\hat{\theta}(k)$ possibly drifting to infinity. Caused by the presence of the representation error $w(x(k))$, this phenomena is known as the "*parameter drift*" phenomena [30].

With the NG update rule of (4.4), in the UUAC as opposed to the SUAC, it is therefore, according to the previous analysis, possible that $\|q(k)\|_F$ is small while $\left\|\tilde{\theta}(k)\right\|_F$ grows without bounds. Hence, when dealing with unstructured uncertainties, the DTNG algorithm may therefore lack robustness in certain cases. We will now look more in depth into what needs to happen in order to realize boundedness of $\tilde{\theta}(k)$ given the possibility of ΔV being positive.

First, recall that when $\mu_{\overline{q},k} \leq \varepsilon_{\overline{q},k}$, $0 \leq Q_E \leq \max_{\mu_q} Q_E$. We can actually write

$$\Delta V(k+1) \leq Q_E \leq \max_{\mu_q} Q_E$$

regardless if $\mu_{\overline{q},k} \leq \varepsilon_{\overline{q},k}$ or $\mu_{\overline{q},k} > \varepsilon_{\overline{q},k}$ even though, in the event that $\mu_{\overline{q},k} > \varepsilon_{\overline{q},k}$, as we have seen in the previous analysis, we can with all certainty further say that $\Delta V(k) < 0$. Consider the expression of Q_E in (4.19). Since $\dfrac{\partial Q_E}{\partial \mu_{\overline{q},k}} = 0$ at

$$\mu_{\overline{q},k} = \mu_{\overline{q}_M} = \frac{\mu_{\overline{w},k}}{\beta_u(k)},$$

$\mu_{\overline{q}_M} \geq 0$, and , granted $\beta_u(k) > 0$ as long as η is chosen so as to satisfy (4.24),

$$\frac{\partial^2 Q_E}{\partial \mu_{\overline{q},k}^2} = -2\beta_u(k) < 0,$$

then $\mu_{\overline{q}_M}$ is where Q_E reaches its maximum value. As a result,

$$\max_{\mu_{\overline{q},k}} Q_E = Q_E|_{\mu_{\overline{q},k}=\mu_{\overline{q}_M}} = \frac{\mu_{\overline{w},k}^2}{\beta_u(k)} \geq 0.$$

Because of (2.26) and with $\mu_{\overline{w},k} = \|\overline{w}(k)\| = \left\|m^{-1}(k)w_\epsilon(k)\right\|_F$ (see (4.12b)),

$$\Delta V(k+1) = V(k+1) - V(k) \leq Q_E \leq \max_{\mu_{\overline{q},k}} Q_E = \frac{\mu_{\overline{w},k}^2}{\beta_u(k)} = \frac{\left\|m^{-1}(k)w_\epsilon(k)\right\|_F^2}{\beta_u(k)}$$

implies $V(k+1) \leq V(k_0) + \Lambda_{NG,1}$, where the scalar

$$\Lambda_{NG,1} = \sum_{\tau=k_0}^{k} \frac{2\mu_{\overline{w},\tau}^2}{\beta_u(\tau)} = \sum_{\tau=k_0}^{k} \frac{\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2}{\beta_u(\tau)}.$$

Thus, for all $k \geq k_0$, we can show boundedness of the parameter error $\tilde{\theta}(k)$ provided that there exists a finite scalar $B_{w_\epsilon,1}$ such that

$$\Lambda_{NG,1} \leq \overline{\Lambda}_{NG,1} = \sum_{\tau=k_0}^{\infty} \frac{2\mu_{\overline{w},\tau}^2}{\beta_u(\tau)} = \sum_{\tau=0}^{\infty} \frac{\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2}{\beta_u(\tau)} \leq B_{w_\epsilon,1} < \infty. \tag{4.52}$$

In fact, if (4.52) is verified then, $V(k+1) \leq V(k_0) + \Lambda_{NG,1}$ also means

$$V(k) \leq V(k_0) + B_{w_\epsilon,1}, \tag{4.53}$$

which, due to (2.25), implies

$$\underline{\beta}_V \left\|\tilde{\theta}(k)\right\|_F^2 \leq V(k) \leq V(k_0) + B_{w_\epsilon,1} \leq \overline{\beta}_V \left\|\tilde{\theta}(k_0)\right\|_F^2 + B_{w_\epsilon,1}.$$

Therefore,

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{NG_u} = \sqrt{\frac{\overline{\beta}_V}{\underline{\beta}_V}\left\|\tilde{\theta}(k_0)\right\|_F^2 + \frac{1}{\underline{\beta}_V}B_{w_\epsilon,1}}. \tag{4.54}$$

Hence, if (4.52) is guaranteed, for all k, both $V(k)$ and $\tilde{\theta}(k)$ are bounded.

Here, using (2.14), (2.15b), (4.38), (C.8), (C.10), and (C.11), we get that

$$\frac{\|q(k)\|_F}{\|m(k)\|_F} = \frac{\|q_\epsilon(k)\|_F}{\|m(k)\|_F} \leq \frac{\left\|\tilde{\theta}(k)\right\|_F \|\phi\left(x(k)\right)\|_F + \|w_\epsilon(k)\|_F}{\overline{\alpha}_{m,1}\left(\alpha\left\|I_{c_\psi}\right\|_F + \|\phi\left(x(k)\right)\|_F^2\right)^{\frac{1}{2}}}.$$

Because of (2.7), (2.17), and (C.8), $\|w_\epsilon(k)\|_F = \|w\left(x(k)\right)\|_F \leq W$ and, subsequently,

$$\frac{\|q(k)\|_F}{\|m(k)\|_F} = \frac{\|q_\epsilon(k)\|_F}{\|m(k)\|_F} \leq \frac{\left\|\tilde{\theta}(k)\right\|_F \|\phi\left(x(k)\right)\|_F + W}{\overline{\alpha}_{m,1}\left(\alpha\left\|I_{c_\psi}\right\|_F + \|\phi\left(x(k)\right)\|_F^2\right)^{\frac{1}{2}}} \tag{4.55}$$

Thus, so long as (4.52) is met and $\tilde{\theta}(k)$ is bounded, (4.55) implies that $\dfrac{\|q(k)\|_F}{\|m(k)\|_F}$ is also bounded. Inequalities (4.41), (4.42), and (4.44) remain the same even in the UUAC. Hence, boundedness of $\dfrac{\|q(k)\|_F}{\|m(k)\|_F}$ also means boundedness of $\left\|m^{-1}(k)q_\epsilon(k)\right\|_F$, $\left\|\phi\left(x(k)\right)m^{-1}(k)\right\|_F$, and $\left\|\Delta\tilde{\theta}(k)\right\|_F$.

Going further, by "completing the square," the expression of Q_E in (4.19) can be rewritten as

$$Q_E = \frac{2\mu_{\overline{w},k}^2}{\beta_u(k)} - \frac{\beta_u(k)\mu_{\overline{q},k}^2}{2} - \frac{\beta_u(k)\varepsilon_{u,k}^2}{2}, \tag{4.56}$$

where the scalar signal

$$\varepsilon_{u,k} = \mu_{\overline{q},k} - \frac{2}{\beta_u(k)}\mu_{\overline{w},k}. \tag{4.57}$$

Notice that the first term in (4.56) is indeed $2\max\limits_{\mu_{\overline{q},k}} Q_E$. Recalling, from (2.26) and (4.23), that

$$\Delta V(k+1) = V(k+1) - V(k) \leq Q_E,$$

and proceeding like we did before while using the expression of Q_E in (4.56) this time around, we would get

$$V(k+1) \leq V(k_0) + 2\Lambda_{NG,1} - \Lambda_{NG,2} - \Lambda_{NG,3},$$

where $\Lambda_{NG,1}$ is as defined above, and the scalars

$$\Lambda_{NG,2} = \sum_{\tau=k_0}^{k} \frac{\beta_u(\tau)\mu_{\overline{q},\tau}^2}{2} \text{ and } \Lambda_{NG,3} = \sum_{\tau=k_0}^{k} \frac{\beta_u(\tau)\varepsilon_{u,\tau}^2}{2}.$$

Because both $-\Lambda_{NG,2} \leq 0$ and $-\Lambda_{NG,3} \leq 0$, we could separately write

$$V(k+1) \leq V(k_0) + 2\Lambda_{NG,1} - \Lambda_{NG,2} - \Lambda_{NG,3} \leq V(k_0) + 2\Lambda_{NG,1} - \Lambda_{NG,2} \text{ and}$$

$$V(k+1) \leq V(k_0) + 2\Lambda_{NG,1} - \Lambda_{NG,2} - \Lambda_{NG,3} \leq V(k_0) + 2\Lambda_{NG,1} - \Lambda_{NG,3}.$$

Consequently,

$$\Lambda_{NG,2} \leq V(k_0) - V(k+1) + 2\Lambda_{NG,1} \leq V(k_0) + 2\Lambda_{NG,1} \text{ and}$$

$$\Lambda_{NG,3} \leq V(k_0) - V(k+1) + 2\Lambda_{NG,1} \leq V(k_0) + 2\Lambda_{NG,1},$$

since, by definition, $V(k) \geq 0$ for all k. Granted that θ and $\hat{\theta}(k_0)$ are bounded, $V(k_0)$ is also bounded. Hence, if (4.52) is in fact verified then $2\overline{\Lambda}_{NG,1} < \infty$, therefore leading to

$$\overline{\Lambda}_{NG,2} = \lim_{k\to\infty} \Lambda_{NG,2} = \sum_{\tau=k_0}^{\infty} \frac{\beta_u(\tau)\mu_{\overline{q},\tau}^2}{2} \leq V(k_0) + 2\lim_{k\to\infty} \Lambda_{NG,1} = V(k_0) + 2\overline{\Lambda}_{NG,1} < \infty$$

and

$$\overline{\Lambda}_{NG,3} = \lim_{k\to\infty} \Lambda_{NG,3} = \sum_{\tau=k_0}^{\infty} \frac{\beta_u(\tau)\varepsilon_{u,\tau}^2}{2} \leq V(k_0) + 2\lim_{k\to\infty} \Lambda_{NG,1} = V(k_0) + 2\overline{\Lambda}_{NG,1} < \infty.$$

Because, $\alpha > 0$ and $0 < \beta_u(k) < 2$ for all k, we can thus conclude that

$$\mu_{\overline{q},k} = \|\overline{q}(k)\|_F = \left\|m^{-1}(k)q_\epsilon(k)\right\|_F \in \mathcal{L}^2 \cap \mathcal{L}^\infty \text{ and } \varepsilon_{u,k} \in \mathcal{L}^2 \cap \mathcal{L}^\infty$$

(actually, as Appendix C points out, for a signal $[\cdot]$, $[\cdot] \in \mathcal{L}^2$ causes $[\cdot] \in \mathcal{L}^\infty$). As we have done before, given (4.44), $\left\|m^{-1}(k)q_\epsilon(k)\right\|_F \in \mathcal{L}^2$ also implies $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$.

It is nevertheless worth reminding again that the previously derived stability results and properties are obtained with the assumption that (4.52) is guaranteed, which may not be true for all types of representation errors. In fact, as we found out, (4.52) is verified for $\|w_\epsilon(k)\|_F \in \mathcal{L}^2$ or vec $\{w_\epsilon(k)\} \in \mathcal{L}^2$, which, given (2.17), also means vec $\{w\left(x(k)\right)\} \in \mathcal{L}^2$. However, we have only assumed that $w\left(x(k)\right)$ is given by (2.7). That is, vec $\{w\left(x(k)\right)\} \in \mathcal{L}^\infty$, which is less restrictive than requiring an $\mathcal{L}^2$ property of vec $\{w\left(x(k)\right)\}$. For the class of representation errors $w\left(x(k)\right)$ such that vec $\{w\left(x(k)\right)\} \in \mathcal{L}^\infty$ is the lone assumption made, (4.52) may still work out in practice but cannot be formally proved. One thing that one may consider doing is to force the term $\frac{\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2}{\beta_u(\tau)}$ in (4.52) to decreases exponentially as time evolves so that it sums up over infinite time to a finite aggregate. We may well attempt such a task in the future.

Ideally, we would like to show that $\tilde{\theta}(k) \to [0]^{r_\theta \times c_\theta}$ as time evolves. However, in the SUAC when using the DTNG update rule of (4.4), proving convergence of $\tilde{\theta}(k)$ to the origin is only possible if $\phi_m(x(k))$ in (4.40) is PE [3, 30]. Matters get even more complicated when dealing with unstructured uncertainties due to the presence of the representation error. Nevertheless, in the UUAC, as it turns out, for a PE $\phi_m(x(k))$, regardless of the condition given by (4.52), it could be shown that $\tilde{\theta}(k)$ is bounded [3, 30], with the bound only depending on the upper bound W of $w\left(x(k)\right)$ and the level of excitation [30]. However, the PE condition on $\phi_m(x(k))$ is demanding and possibly even more restrictive than requiring (4.52).

If (4.52) is met, as suggested by (4.54), $\left\|\tilde{\theta}(k)\right\|_F$ dwells inside a bounded neighborhood as large as Θ_{NG_u}, which basically also serves as its ultimate bound. The smaller Θ_{NG_u} is, the closer $\tilde{\theta}(k)$ gets to $[0]^{r_\theta \times r_f}$. It therefore begs the question, how large is Θ_{NG_u}? As (4.54) shows, the size of Θ_{NG_u} is affected by initial conditions, i.e., $\hat{\theta}(k_0)$ since $\tilde{\theta}(k_0) = \hat{\theta}(k_0) - \theta$. The closer $\hat{\theta}(k_0)$ is to θ, the smaller Θ_{NG_u} becomes and the better the learning/prediction performance of the algorithm will be. However, θ being unknown, it is difficult to say how to pick $\hat{\theta}(k_0)$ so as to minimize $\left\|\tilde{\theta}(k_0)\right\|_F^2$ and, consequently, Θ_{NG_u}. What we may be able to control however, is how to minimize $B_{w_\epsilon,1}$ in (4.54) so as to get at least closer to the results obtained in the SUAC, i.e., (4.25).

4.1.3 Concluding Remarks about DTNG

Unlike in the SUAC, the boundedness of $\tilde{\theta}(k)$ in the UUAC is obtained if (4.52) is guaranteed, which is true for $w\left(x(k)\right) \in \mathcal{L}^2$. Notice how conveniently $w\left(x(k)\right) = [0]^{r_f \times c_f} \in \mathcal{L}^2$ for the SUAC.

For performance and robustness sakes, it is desirable that $\hat{\theta}(k)$ converges to θ or $\hat{\theta}(k)$ converges to a (small) neighborhood around θ as $k \to \infty$. However, as far as the NG algorithm is concerned, this is only realizable if $\phi_m(x(k))$ is PE [2, 3, 30]. The PE condition, as already mentioned, is demanding however. Persistency of excitation, as **Definition B.2** in Appendix B and/or expression (B.2) show, imposes conditions on past, current and future regressor vectors, and, for those reasons, is not simple to achieve and/or monitor on-line.

Table 4.1 summarizes the DTNG algorithm, its results and properties.

Table 4.1: DTNG algorithm, results, and properties.

DTNG algorithm	
For $k \geq k_0$, $\phi\left(x(k)\right) \in \mathbb{R}^{r_\phi \times c_\phi}$ being the regressor, $q_\epsilon(k)$ given by (2.14), some $\Gamma \in \mathbb{R}^{r_\phi \times r_\phi}$, and some initial guess $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$, update $\hat{\theta}$ by applying: $m^2(k) = \alpha I_{c_\phi} + \phi^\top(x(k))\,\phi\left(x(k)\right)$, $\Delta\hat{\theta}_{NG}(k) = \phi\left(x(k)\right)\left(m^2(k)\right)^{-1} q_\epsilon(k)$, Design $\Gamma = \Gamma^\top = \Gamma(k) > 0$ such that, $\eta = \lambda_{\max}(\Gamma)$ verifies (4.24) (or (4.33) only in the SUAC), $\hat{\theta}(k+1) = \hat{\theta}(k) - \Gamma\Delta\hat{\theta}_{NG}(k)$.	
SUAC results and properties	**UUAC results and properties**
$\tilde{\theta}(k)$, $\left\Vert m^{-1}(k)q_\epsilon(k)\right\Vert_F$, and $\left\Vert\Delta\tilde{\theta}(k)\right\Vert_F$ are bounded; $\left\Vert m^{-1}(k)q_\epsilon(k)\right\Vert_F$, and $\left\Vert\Delta\tilde{\theta}(k)\right\Vert_F$ belong to $\mathcal{L}^2$.	Same as SUAC only if vec $\{w\left(x(k)\right)\} \in \mathcal{L}^2$.
Convergence Properties	
$\tilde{\theta}(k)$ converges to the origin (meaning $\hat{\theta}(k)$ converges to θ) in the SUAC or $\tilde{\theta}(k)$ converges to a neighborhood of the origin (meaning $\hat{\theta}(k)$ converges to a neighborhood of θ) in the UUAC if $\phi_m(x(k))$ in (4.40) is persistently exciting.	

Next, we present the Normalized Gradient based Concurrent Learning algorithm in DT framework. We show that, given a less constraining condition comparatively to that of persistency of excitation, Concurrent Learning can help achieve better parameter identification.

4.2 Discrete-Time Normalized Gradient based Concurrent Learning for both Structured and Unstructured Uncertainty Approximation Cases

After presenting the discrete-time Normalized Gradient based CL algorithm in Section 4.2.1, we investigate the SUAC and the UUAC when using it.

4.2.1 Discrete-Time Normalized Gradient Based Concurrent Learning Algorithm

Starting with an initial guess $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$ and, with $r_\theta = r_\phi$, given positive definite, symmetric matrices Γ, ξ_{NG}, $\xi_{CL} \in \mathbb{R}^{r_\phi \times r_\phi} = \mathbb{R}^{r_\theta \times r_\theta}$ (i.e., Γ, ξ_{NG}, $\xi_{CL} > 0$ and $\Gamma^\top = \Gamma$, $\xi_{NG}^\top = \xi_{NG}$, $\xi_{CL}^\top = \xi_{CL}$), we put forth for $k \geq k_0$ the DTNG based CL update

$$\hat{\theta}(k+1) = \hat{\theta}(k) - \Gamma \left\{ \xi_{NG} \Delta\hat{\theta}_{NG}(k) + \xi_{CL} \Delta\hat{\theta}_{CL}(k) \right\}. \tag{4.58}$$

The instantaneous update term $\Delta\hat{\theta}_{NG}(k) \in \mathbb{R}^{r_\phi \times c_{q_\epsilon}}$, with $c_{q_\epsilon} = c_\theta$, is as expressed in (4.5). Given matrix $\bar{Z}_G$ and the approximation error based on recorded data $q_{Z,\epsilon}(k)$, both of which are defined defined in (3.27) and (3.9) respectively, we define $\Delta\hat{\theta}_{CL}(k) \in \mathbb{R}^{r_\phi \times c_{q_\epsilon}}$ as

$$\Delta\hat{\theta}_{CL}(k) = Z_G \, q_{Z,\epsilon}(k). \tag{4.59}$$

Notice that the adjustments terms $\Delta\hat{\theta}_{NG}$ and $\Delta\hat{\theta}_{CL}$ in (4.58) do not necessarily contribute equally to the update of the parameter estimate $\hat{\theta}$, as ξ_{NG} does not have to be set equal to ξ_{CL}. Consequently, if need be, either update terms can be prioritized over the other. It should also be added that, just like Γ, ξ_{NG} and ξ_{CL} can be set to vary with time, i.e., $\xi_{NG} = \xi_{NG}(k)$ and $\xi_{CL} = \xi_{CL}(k)$.

The term $\Delta\hat{\theta}_{CL}$ in (4.58) provides adjustments based on recorded data. In doing so, update of the parameter estimate $\hat{\theta}(k)$ (if any) can and/or might continue even when the instantaneous approximation error $q_\epsilon(k) = [0]^{r_{c_{q_\epsilon}} \times c_{q_\epsilon}}$, or, equivalently, $\Delta\hat{\theta}_{NG}(k) = [0]^{r_\phi \times c_{q_\epsilon}}$, since $q_\epsilon(k) = [0]^{r_{c_{q_\epsilon}} \times c_{q_\epsilon}}$

and, thus, from (2.14), $q(k) = [0]^{r_f \times c_f}$ does not necessarily mean that $\hat{\theta}(k)$ has converged to θ.

Let $\lambda_{\max}(\xi_{NG}) = \varepsilon_{NG}$ and $\lambda_{\max}(\xi_{CL}) = \varepsilon_{CL}$. If **Condition 3.1.1** is met, using the CL update rule of (4.58) for

$$0 < \eta < \frac{2}{2\varepsilon_{NG}\overline{\lambda}_{\Phi_k} \left\|m^{-1}(k)\right\|_F^2 + \varepsilon_{CL}\overline{\lambda}_{\Phi_{Z_G,k}}} = \overline{\eta}_{CL}(k) \tag{4.60}$$

and Γ constructed such that $\lambda_{\max}(\Gamma) = \eta$, we can achieve convergence of the parameter error $\tilde{\theta}(k)$ to the origin for the SUAC and to a neighborhood of the origin for the UUAC as time evolves.

We prove our assertion next. However, first, from the expressions of $\Phi_{Z_G,k}$ in (3.15) and $q_{Z,\epsilon}(k)$ in (3.28b), notice that (4.59) can be rewritten as

$$\begin{aligned}\Delta\hat{\theta}_{CL}(k) &= \bar{Z}_G \left(Z^\top \tilde{\theta}(k) - w_{Z,\epsilon}(k)\right) = \bar{Z}_G Z^\top \tilde{\theta}(k) - \bar{Z}_G\, w_{Z,\epsilon}(k) \\ &= \Phi_{Z_G,k}\tilde{\theta}(k) - \varepsilon_{w,k},\end{aligned} \tag{4.61}$$

with $\varepsilon_{w,k} \in \mathbb{R}^{r_\theta \times c_\theta}$ $(r_\theta = r_\phi = r_Z)$ given by

$$\varepsilon_{w,k} = \bar{Z}_G\, w_{Z,\epsilon}(k). \tag{4.62}$$

Let $\Delta\theta_w(k) \in \mathbb{R}^{r_\theta \times c_\theta}$ be such that

$$\Delta\theta_w(k) = \phi\left(x(k)\right)\left(m^2(k)\right)^{-1} w_\epsilon(k). \tag{4.63}$$

Substituting the expression of $q_\epsilon(k)$ in (2.15b) into that of $\Delta\hat{\theta}_{NG}(k)$ in (4.5), using (4.58), (4.61), and letting

$$\bar{\varepsilon}_{w,k} = \Delta\theta_w(k) + \varepsilon_{w,k}, \tag{4.64}$$

the parameter error, fundamentally defined in (2.10), can be rewritten as

$$\tilde{\theta}(k+1) = \left[I_{r_\theta} - \Gamma\xi_{NG}\Phi_{m,k} - \Gamma\xi_{CL}\Phi_{Z_G,k}\right]\tilde{\theta}(k) + \Gamma\xi_{CL}\bar{\varepsilon}_{w,k}. \tag{4.65}$$

where $\Phi_{m,k}$ is defined by (4.27). Now, let matrices $Q(k) \in \mathbb{R}^{r_\theta \times r_\theta}$, $E_k \in \mathbb{R}^{r_\theta \times c_\theta}$, and $\overline{E}_{w,k} \in \mathbb{R}^{c_\theta \times c_\theta}$ be such that

$$Q_k = I_{r_\theta} - \Gamma \xi_{NG} \Phi_{m,k} - \Gamma \xi_{CL} \Phi_{Z_G,k}, \tag{4.66}$$

$$E_k = Q_k^\top \xi_{CL} \bar{\varepsilon}_{w,k}, \text{ and} \tag{4.67}$$

$$\overline{E}_{w,k} = \bar{\varepsilon}_{w,k}^\top \xi_{CL} \Gamma \xi_{CL} \bar{\varepsilon}_{w,k}. \tag{4.68}$$

With $\tilde{\theta}(k+1)$ given by (4.65), $V(k+1)$ obtained using (4.7) expands to

$$V(k+1) = \text{tr}\left\{\tilde{\theta}^\top(k) \overline{Q}_k \tilde{\theta}(k)\right\} + \text{tr}\left\{\Upsilon_{E,k}\right\} + \text{tr}\left\{\overline{E}_{w,k}\right\}, \tag{4.69}$$

where $\overline{Q}_k = Q_k^\top \Gamma^{-1} Q(k)$ and $\Upsilon_{E,k} \in \mathbb{R}^{c_\theta \times c_\theta}$ given by

$$\Upsilon_{E,k} = \tilde{\theta}^\top(k) E_k + E_k^\top \tilde{\theta}(k).$$

Since, for any $m_a \in \mathbb{R}^{r_m \times c_m}$, $r_m \in \mathbb{N}_+$ and $c_m \in \mathbb{N}_+$, tr$(m_a^\top) =$ tr(m_a), we get that

$$\text{tr}\left\{\tilde{\theta}^\top(k) \overline{Q}_k \tilde{\theta}^\top(k)\right\} = \text{tr}\left\{\tilde{\theta}^\top(k) \overline{\overline{Q}}_k \tilde{\theta}^\top(k)\right\},$$

where, based on the expression of $\overline{Q}_k$ above and (4.66),

$$\overline{\overline{Q}}_k = \Gamma^{-1} + \overline{\overline{Q}}_{NG,k} - 2\Phi_{Z_G,k}\, \xi_{CL} + 2\Phi_{Z_G,k} \xi_{CL}\, \Gamma \xi_{NG}\, \Phi_{m,k} + \Phi_{Z_G,k}\, \xi_{CL} \Gamma \xi_{CL}\, \Phi_{Z_G,k}, \tag{4.70}$$

with

$$\overline{\overline{Q}}_{NG,k} = -\left(2I_{r_\theta} - \xi_{NG} \Phi_{m,k} \Gamma\right) \Phi_{m,k} \xi_{NG}, \tag{4.71}$$

as well as

$$\text{tr}\left(\Upsilon_{E,k}\right) = 2\text{tr}\left\{\tilde{\theta}^\top(k) E_k\right\}. \tag{4.72}$$

Requiring that Γ, ξ_{NG}, and ξ_{CL} be positive definite matrices, then, $\eta = \lambda_{\max}(\Gamma) > 0$, $\varepsilon_{NG} = \lambda_{\max}(\xi_{NG}) > 0$, $\varepsilon_{CL} = \lambda_{\max}(\xi_{CL}) > 0$. Notice that $\overline{\overline{Q}}_{NG,k}$ in (4.71) is exactly $\overline{Q}_{NG,k}$ in (4.29) if $\xi_{NG} = I_{r_\phi}$. To find an upper-bound for $\overline{\overline{Q}}_{NG,k}$, we will use (4.30). Doing so while recalling that $r_\theta = r_\phi$, we have that

$$2I_{r_\theta} - \xi_{NG}\Phi_{m,k}\Gamma \geq \left(2 - \eta\varepsilon_{NG}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2\right) I_{r_\theta}$$

and, using (4.71),

$$\overline{\overline{Q}}_{NG,k} \leq -\left(2 - \eta\varepsilon_{NG}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2\right)\varepsilon_{NG}\Phi_{m,k} \leq 0$$

first because $\Phi_{m,k} \geq 0$ for all k and, second, if Γ, with $\eta = \lambda_{\max}(\Gamma)$, is such that, for $\phi\left(x(k)\right) \neq [0]^{r_\phi \times c_\phi}$,

$$0 < \eta < \frac{2}{\varepsilon_{NG}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2} = \overline{\overline{\eta}}_{NG}(k). \tag{4.73}$$

It is worth noting that (4.73) is quite similar to (4.33). Moving along, if we were to pick η (and therefore Γ) such that (4.73) is met, then, by leaving $\overline{\overline{Q}}_{NG,k} \leq 0$ out of the expression of $\overline{\overline{Q}}_k$ in (4.70), we get

$$\overline{\overline{Q}}_k \leq \left(I_{r_\theta} - \Phi_{Z_G,k}\overline{\overline{Q}}_{CL,k}\right)\Gamma^{-1}, \tag{4.74}$$

with

$$\overline{\overline{Q}}_{CL,k} = 2\xi_{CL}\Gamma - 2\xi_{CL}\Gamma\xi_{NG}\Phi_{m,k}\Gamma - \xi_{CL}\Gamma\xi_{CL}\Phi_{Z_G,k}\Gamma.$$

Denoting $\underline{\lambda}_{CL} = \lambda_{\min}(\Gamma)\lambda_{\min}(\xi_{CL})$ and given (4.30), we can say that

$$\overline{\overline{Q}}_{CL,k} \geq \left(2\underline{\lambda}_{CL} - 2\eta^2\varepsilon_{NG}\varepsilon_{CL}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k-1)\right\|_F^2 - \eta^2\varepsilon_{CL}^2\overline{\lambda}_{\Phi_{Z_G,k}}\right) I_{r_\theta}.$$

If Z_G is full row rank (as required by **Condition 3.1.1**), then $\Phi_{Z_G,k} = Z_G Z_G^\top$ is positive definite ($\Phi_{Z_G,k} > 0$), which means the minimum and maximum eigenvalues of $\Phi_{Z_G,k}$, i.e., $\underline{\lambda}_{\Phi_{Z_G,k}}$ and $\overline{\lambda}_{\Phi_{Z_G,k}}$ respectively, are such that $\overline{\lambda}_{\Phi_{Z_G,k}} \geq \underline{\lambda}_{\Phi_{Z_G,k}} > 0$. For

$$\overline{R}_{NG,k} = 1 - \underline{\lambda}_{\Phi_{Z_G,k}}\left(2\underline{\lambda}_{CL} - 2\eta^2\varepsilon_{NG}\varepsilon_{CL}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2 - \eta^2\varepsilon_{CL}^2\overline{\lambda}_{\Phi_{Z_G,k}}\right),$$

from (4.74), a further upper bound of $\overline{\overline{Q}}_k$ is $\overline{\overline{Q}}_k \leq \overline{R}_{NG,k}\Gamma^{-1}$. However, with

$$-\eta\varepsilon_{CL} \leq -\lambda_{\min}(\Gamma)\,\lambda_{\min}(\xi_{CL}) = -\underline{\lambda}_{CL},$$

ensuring $\overline{\overline{Q}}_k \leq R_{NG,k}\Gamma^{-1}$,

$$R_{NG,k} = 1 - \eta\varepsilon_{CL}\underline{\lambda}_{\Phi_{Z_G,k}}\left[2 - \eta\left(2\varepsilon_{NG}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2 + \varepsilon_{CL}\overline{\lambda}_{\Phi_{Z_G,k}}\right)\right]. \tag{4.75}$$

For convenience, as $R_{NG,k} \leq \overline{R}_{NG,k}$, we have that $\overline{\overline{Q}}_k \leq R_{NG,k}\Gamma^{-1} \leq \overline{R}_{NG,k}\Gamma^{-1}$. Finally, with $\overline{\overline{Q}}_k \leq R_{NG,k}\Gamma^{-1}$, from (4.69), and given (4.7), we write

$$V(k+1) \leq R_{NG,k}V(k) + 2\text{tr}\left\{\tilde{\theta}^\top(k)E_k\right\} + \text{tr}\left\{\overline{E}_{w,k}\right\}. \tag{4.76}$$

4.2.2 DTNG based CL for the SUAC

If f is a structured uncertainty then $w\left(x(k)\right) = [0]^{r_f \times c_f}$ and, similarly, $w_Z(x(k)) = [0]^{r_{qZ} \times c_{qZ}}$ for all k, which, according to (2.17), (3.30), (4.62), and (4.63), means $w_\epsilon(k) = [0]^{r_{q\epsilon} \times c_{q\epsilon}}$, $w_{Z,\epsilon}(k) = [0]^{c_Z \times c_\theta}$, $\varepsilon_{w,k} = [0]^{r_\theta \times c_\theta}$, and $\Delta\theta_w(k) = [0]^{r_\theta \times c_\theta}$ respectively. Hence, given (4.64), (4.67), and (4.68), $\bar{\varepsilon}_{w,k} = E_k = [0]^{r_\theta \times r_f}$ and $\overline{E}_{w,k} = [0]^{r_f \times r_f}$. Consequently, (4.76) reduces to

$$V(k+1) \leq R_{NG,k}V(k). \tag{4.77}$$

Now, notice that we effectively have $0 < R_{NG,k} < 1$ if η in the expression of $R_{NG,k}$, i.e., (4.75), verifies (4.60). It is however necessary to add that a value of η that meets (4.60) also verifies the first condition that was imposed onto it while developing this proof, i.e, (4.73), as, given (4.60) and (4.73), $\overline{\eta}_{CL}(k) < \overline{\overline{\eta}}_{NG}(k)$.

In summary, by appropriately designing for η such that $0 < R_{NG,k} < 1$, i.e., making sure that (4.60) is valid, then, as a result of (2.25) and (4.77),

$$\lim_{k\to\infty} \underline{\beta}_V \left\|\tilde{\theta}(k)\right\|_F^2 \leq \lim_{k\to\infty} V(k) = 0$$

if the rank condition on Z_G (as opposed to persistency of excitation) is satisfied. Hence, $\tilde{\theta}(k) = [0]^{r_\theta \times c_\theta}$ is exponentially stable. Additionally, given (4.77) and $0 < R_{NG,k} < 1$, for all k, $V(k) \leq V(k_0)$ and, just like (4.25),

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{NG,CL_s} = \sqrt{\frac{\overline{\beta}_V}{\underline{\beta}_V}} \left\|\tilde{\theta}(k_0)\right\|_F \tag{4.78}$$

The upper bounds of $\frac{\|q(k)\|_F}{\|m(k)\|_F}$, $\left\|m^{-1}(k)q_\epsilon(k)\right\|_F$, and $\left\|\Delta\tilde{\theta}(k)\right\|_F$ are as given in (4.39), (4.41), and (4.44). Hence, with $\tilde{\theta}(k)$ being bounded and converging to the origin exponentially, so do $\frac{\|q(k)\|_F}{\|m(k)\|_F}$, $\left\|m^{-1}(k)q_\epsilon(k)\right\|_F$, and $\left\|\Delta\tilde{\theta}(k)\right\|_F$. Besides, much like in Section 4.1.1, it could be shown that both $\left\|m^{-1}(k)q_\epsilon(k)\right\|_F \in \mathcal{L}^2$ and $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$.

4.2.3 DTNG based CL for the UUAC

Given (2.26) and going back to (4.76), we can write

$$\Delta V(k+1) \leq (R_{NG,k} - 1)V(k) + 2\text{tr}\left\{\tilde{\theta}^\top(k)E_k\right\} + \text{tr}\left\{\overline{E}_{w,k}\right\}.$$

As previously defined and because (C.8), let $\mu_{\tilde{\theta},k} = \left\|\tilde{\theta}(k)\right\|_F = \left\|\tilde{\theta}^\top(k)\right\|_F$. Using the expression of $\overline{E}_{w,k}$ in (4.68), realize that

$$\text{tr}\left\{\overline{E}_{w,k}\right\} = \text{tr}\left\{(E_{w,k})^\top (E_{w,k})\right\},$$

where

$$E_{w,k} = \sqrt{\Gamma}\xi_{CL}\bar{\varepsilon}_{w,k}. \tag{4.79}$$

Finally, given (C.9), (C.15), based on the definition of $V(k)$, i.e., (4.7), with $\lambda_{\max}\left(\Gamma^{-1}\right) = \frac{1}{\lambda_{\min}(\Gamma)}$, which means $0 < \Gamma^{-1} \leq \lambda_{\max}\left(\Gamma^{-1}\right) I_{r_\phi} = \frac{1}{\lambda_{\min}(\Gamma)} I_{r_\phi}$, and letting

$$A_1 = \frac{R_{NG,k} - 1}{\lambda_{\min}(\Gamma)},$$

we have that

$$\Delta V(k+1) \leq A_1 \mu_{\tilde{\theta},k}^2 + 2\left\|E_k\right\|_F \mu_{\tilde{\theta},k} + \left\|E_{w,k}\right\|_F^2. \tag{4.80}$$

We now proceed to upper bounding $\left\|E_k\right\|_F$ and $\left\|E_{w,k}\right\|_F$. Before continuing, it is worth recalling that, according to (4.49), (4.50), and (4.42), for all k,

$$m^{-1}(k) \leq \frac{1}{\sqrt{\alpha}} I_{c_\phi} \text{ and } \left\|m^{-1}(k)\right\|_F \leq \frac{1}{\sqrt{\alpha}} \left\|I_{c_\phi}\right\|_F = \sqrt{\frac{c_\phi}{\alpha}},$$

$$\left\|m^{-1}(k) w_\epsilon(k)\right\|_F \leq \overline{W} = \sqrt{\frac{c_\phi}{\alpha}} W,$$

and

$$\left\|\phi\left(x(k)\right) m^{-1}(k)\right\|_F \leq \beta_{\phi,1} \left\|I_{c_\phi}\right\|_F \frac{\left\|\phi\left(x(k)\right)\right\|_F}{\overline{\alpha}_{m,1}\left(\alpha \left\|I_{c_\psi}\right\|_F + \left\|\phi\left(x(k)\right)\right\|_F^2\right)^{\frac{1}{2}}} = B_{\phi_m}.$$

From (2.7) and (3.26), $\left\|w\left(x(k)\right)\right\| \leq W$ and $\left\|w_Z(x(k))\right\|_F \leq W_Z = c_Z W$, which, given (2.17), (3.30), and (C.8), means $\left\|w_\epsilon(k)\right\| \leq W$ and $\left\|w_{Z,\epsilon}(k)\right\| \leq W_Z$. Also, $\left\|I_r\right\|_F = \sqrt{r}$ for any $r \in \mathbb{N}_+$, $r_\theta = r_\phi = r_Z$, $\Gamma \leq \eta I_{r_\phi} = \eta I_{r_\theta}$, $\xi_{NG} \leq \varepsilon_{NG} I_{r_\phi} = \varepsilon_{NG} I_{r_\theta}$, $\xi_{CL} \leq \varepsilon_{CL} I_{r_\phi} = \varepsilon_{CL} I_{r_\theta}$, and $\Phi_{Z_G,k} = Z_G Z_G^\top \leq \overline{\lambda}_{\Phi_{Z_G,k}} I_{r_Z} = \overline{\lambda}_{\Phi_{Z_G,k}} I_{r_\theta}$. Hence, $\left\|I_{r_\theta}\right\|_F = \sqrt{r_\theta}$, $\left\|\Gamma\right\|_F \leq \eta\sqrt{r_\theta}$, $\left\|\xi_{NG}\right\|_F \leq \varepsilon_{NG}\sqrt{r_\theta}$, and $\left\|\xi_{CL}\right\|_F \leq \varepsilon_{CL}\sqrt{r_\theta}$. Among other properties, we will make use of (C.8), (C.9), (C.10), (C.11), and (C.12) when finding upper bounds for $\left\|E_k\right\|_F$ and $\left\|E_{w,k}\right\|_F$.

First, using the expression of E_k, i.e., (4.67),

$$\left\|E_k\right\|_F \leq \left\|Q_k\right\|_F \left\|\xi_{CL}\right\|_F \left\|\bar{\varepsilon}_{w,k}\right\|_F \leq \varepsilon_{CL}\sqrt{r_\theta} \left\|Q_k\right\|_F \left\|\bar{\varepsilon}_{w,k}\right\|_F.$$

Recalling (4.66), $\Phi_{m,k}$ in (4.27), then

$$\begin{aligned}\left\|Q_k\right\|_F &\leq \left\|I_{r_\theta}\right\|_F + \left\|\Gamma\right\|_F \left\|\xi_{NG}\right\|_F \left\|\phi\left(x(k)\right) m^{-1}(k)\right\|_F^2 + \left\|\Gamma\right\|_F \left\|\xi_{CL}\right\|_F \left\|ZZ^\top\right\|_F \\ &< \sqrt{r_\theta} + \eta r_\theta \left(\varepsilon_{NG} B_{\phi_m}^2 + \varepsilon_{CL} \overline{\lambda}_{\Phi_{Z_G,k}} \sqrt{r_\theta}\right).\end{aligned}$$

Carrying on, from (4.64), $\|\bar{\varepsilon}_{w,k}\|_F \leq \|\Delta\theta_w(k)\|_F + \|\varepsilon_{w,k}\|_F$. Given (3.11, (4.62), and (4.63),

$$\|\varepsilon_{w,k}\|_F \leq \left\|\bar{Z}_G\right\|_F \|w_{Z,\epsilon}(k)\|_F \leq c_Z B_g W_Z = c_Z^2 B_g W$$

and

$$\begin{aligned}\|\Delta\theta_w(k)\|_F &= \left\|\left[\phi\left(x(k)\right) m^{-1}(k)\right]\left[m^{-1}(k) w_\epsilon(k)\right]\right\|_F \\ &\leq \left\|\phi\left(x(k)\right) m^{-1}(k)\right\|_F \left\|m^{-1}(k) w_\epsilon(k)\right\|_F \leq B_{\phi_m}\overline{W} = \sqrt{\frac{c_\phi}{\alpha}} B_{\phi_m} W.\end{aligned}$$

Thus,

$$\|\bar{\varepsilon}_{w,k})\|_F \leq \left\|\bar{Z}_G\right\|_F \|w_{Z,\epsilon}(k)\|_F < \sqrt{\frac{c_\phi}{\alpha}} B_{\phi_m} W + c_Z^2 B_g W = \left(\sqrt{\frac{c_\phi}{\alpha}} B_{\phi_m} + c_Z^2 B_g\right) W$$

and

$$\|E_k\|_F \leq \varepsilon_{CL}\sqrt{r_\theta}\,\|Q_k\|_F \|\bar{\varepsilon}_{w,k}\|_F \leq \frac{B_1}{2},$$

where, for

$$B_2 = 2\sqrt{r_\theta}\left(\sqrt{r_\theta} + \eta r_\theta\left(\varepsilon_{NG} B_{\phi_m}^2 + \varepsilon_{CL}\overline{\lambda}_{\Phi_{Z_G,k}}\sqrt{r_\theta}\right)\right),$$

$$B_1 = \varepsilon_{CL}\left(\sqrt{\frac{c_\phi}{\alpha}} B_{\phi_m} + c_Z^2 B_g\right) W B_2.$$

Further, the expression of $E_{w,k}$ in (4.79) allows us to write

$$\|E_{w,k}\|_F^2 \leq \left\|\sqrt{\Gamma}\right\|_F^2 \|\xi_{CL}\|_F^2 \|\bar{\varepsilon}_{w,k}\|_F^2 \leq C_1,$$

with

$$C_1 = \eta r_\theta\sqrt{r_\theta}\,\varepsilon_{CL}^2\left(\sqrt{\frac{c_\phi}{\alpha}} B_{\phi_m} + c_Z^2 B_g\right)^2 W^2.$$

Notice that $B_2 \neq 0$ as $r_\theta \geq 1$ by definition. Also, with $W \geq 0$, $B_1 \geq 0$ and $C_1 = r_\theta\sqrt{r_\theta}\left(\frac{B_1}{B_2}\right)^2 \geq 0$.

As pointed out before, making sure that η is picked such that (4.60) is verified when applying CL leads to $0 < R_{NG,k} < 1$. As a result, $-1 < A_1 = \dfrac{R_{NG,k} - 1}{\lambda_{\min}(\Gamma)} < 0$. From (4.80), we get

$$\Delta V(k+1) \leq Q_E = A_1 \mu_{\tilde{\theta},k}^2 + B_1 \mu_{\tilde{\theta},k} + C_1, \tag{4.81}$$

given our upper bounds of $\|E_k\|_F$ and $\|E_{w,k}\|_F$. Because $A_1 < 0$ if (4.60) is met and $\mu_{\tilde{\theta},k} \geq 0$, the only valid root (because nonnegative) of the quadratic expression Q_E in (4.81) is

$$B_{NG,CL_u} = \frac{-B_1 - \sqrt{B_1^2 - 4A_1C_1}}{2A_1}.$$

At $\mu_{\tilde{\theta},k} = \mu_{\tilde{\theta}_M} = -\dfrac{B_1}{2A_1}$, $\dfrac{\partial Q_E}{\partial \mu_{\tilde{\theta},k}} = 0$ and $\dfrac{\partial^2 Q_E}{\partial \mu_{\tilde{\theta},k}^2} = 2A_1 < 0$ granted (4.60) is verified. Hence, Q_E attains its maximum at the stationary point $-\dfrac{B_1}{2A_1} \geq 0$. That is,

$$\max_{\mu_{\tilde{\theta},k}} Q_E = Q_E|_{\mu_{\tilde{\theta},k} = \mu_{\tilde{\theta}_M}} = -\frac{B_1^2}{4A_1} + C_1 \geq 0.$$

With $A_1 < 0$, if $\mu_{\tilde{\theta},k} > B_{NG,CL_u}$,

$$\Delta V(k+1) = V(k+1) - V(k) \leq Q_E < 0,$$

whereas, when $\tilde{\theta}$ enters the set

$$S_{NG,CL_u} = \left\{\tilde{\theta} : \left\|\tilde{\theta}(k)\right\|_F \leq B_{NG,CL_u}\right\},$$

$$\Delta V(k) = V(k) - V(k-1) \leq \max_{\mu_{\tilde{\theta},k}} Q_E,$$

meaning that it is possible to have $\Delta V(k) \geq 0$ in that case. However, for discrete times thereafter, $\tilde{\theta}$ stays within the positively invariant set S_{NG,CL_u} (consult [31] to see how S_{NG,CL_u} can be formally proved to be positively invariant). Thus, for all k,

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{NG,CL_u} = \max\left(\left\|\tilde{\theta}(k_0)\right\|_F, B_{NG,CL_u}\right) \tag{4.82}$$

and, based on (2.25),

$$V(k) \leq \overline{\beta}_V \left\|\tilde{\theta}(k)\right\|_F^2 \leq \overline{\beta}_V \max\left(\left\|\tilde{\theta}(k_0)\right\|_F^2, B_{NG,CL_u}^2\right).$$

Here also, without requiring PE condition, provided instead that the rank condition on Z_G is verified and η satisfies (4.60), the DTNG based CL algorithm of (4.58) guarantees that $V(k)$ and $\tilde{\theta}(k)$ are bounded. We do not have convergence of the parameter error matrix to the origin as we did in the SUAC, but, rather, with S_{NG,CL_u} being invariant,

$$\lim_{k \to \infty} \left\| \tilde{\theta}(k) \right\|_F \leq B_{NG,CL_u}.$$

We could find a scalar, constant upper bound to B_{NG,CL_u} and conclude that $\left\| \tilde{\theta}(k) \right\|_F$ is uniformly, ultimately, bounded (UUB).

4.2.4 Concluding Remarks about DTNG based CL

For the more general approximation case, i.e., the UUAC, unlike in Section 4.1.2, we were able to show that the ultimate bound of $\left\| \tilde{\theta}(k) \right\|_F$, i.e, B_{NG,CL_u}, which, by the way, is obtained without necessitating that vec $\{w\left(x(k)\right)\} \in \mathcal{L}^2$, solely depends on entities such as the upper bound W of the representation error w, spectral properties $\overline{\lambda}_{\Phi_k}$, $\underline{\lambda}_{\Phi_{Z_G,k}}$, $\overline{\lambda}_{\Phi_{Z_G,k}}$, and design parameters such as η, α, c_Z, and r_θ. This makes it easier from a design point of view because W might be the only parameter over which there may not be much control. Proper choice of design parameters can help reduce the size of B_{NG,CL_u}, which, ideally, we would want to make as close as possible to zero.

As a result of $\left\| \tilde{\theta}(k) \right\|_F$ being bounded for all k, in the UUAC, $\dfrac{\|q(k)\|_F}{\|m(k)\|_F}$, $\left\| m^{-1}(k) q_\epsilon(k) \right\|_F$, and $\left\| \Delta\tilde{\theta}(k) \right\|_F$, which are as given in (4.55), (4.41), and (4.44) receptively, are bounded and their upper bounds, similarly to that of $\left\| \tilde{\theta}(k) \right\|_F$, can be rewritten so as to show explicit dependency on the previously mentioned controllable parameters. Lastly, we would like to point out that each of $\dfrac{\|q(k)\|_F}{\|m(k)\|_F}$, $\left\| m^{-1}(k) q_\epsilon(k) \right\|_F$, and $\left\| \Delta\tilde{\theta}(k) \right\|_F$ converges to an ultimate bound as $\left\| \tilde{\theta}(k) \right\|_F$ converges to its ultimate bound.

The following theorem summarizes our results.

Theorem 4.2.1. *Let f be an uncertainty to approximate. Consider the approximation model (2.1), the on-line scheme (2.8), and the approximation error (2.9a) (equivalently, (2.9b) or (2.13)). If the rank condition, described by Condition 3.1.1, is met and parameter $\eta = \lambda_{\max}(\Gamma)$ is picked to satisfy (4.60), then, the DTNG based CL adaptation law (4.58) guarantees that:*

- *$\tilde{\theta}(k)$, consequently $\hat{\theta}(k)$ (since θ is considered to be constant), $m^{-1}(k)q_{\epsilon}(k)$ and $\Delta\tilde{\theta}(k) = \tilde{\theta}(k) - \tilde{\theta}(k-1)$ are bounded;*
- *if in the SUAC, $\tilde{\theta}(k) \to [0]^{r_\theta \times r_f}$ exponentially, or, equivalently, $\hat{\theta}(k) \to \theta$ exponentially as $k \to \infty$. Also, both $\left\|m^{-1}(k)q_{\epsilon}(k)\right\|_F \in \mathcal{L}^2$ and $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$;*
- *if in the UUAC, $\tilde{\theta}(k)$ is ultimately, uniformly bounded, as its Frobenius norm converges to a positively invariant set.*

Table 4.2 is a summary of the DTNG based CL algorithm, its results and properties.

We make the following remarks.

Remark 4.2.1. The upper bound $\overline{\eta}_{CL}(k)$ on η given in (4.60) is written in a more general form than what we previously found in [28] (and also used in [29]). For a scalar normalization signal $m(k)$, as is the case in [28, 29], (4.60) can be rewritten as

$$0 < \eta < \frac{2m^2(k)}{2\varepsilon_{NG}\overline{\lambda}_{\Phi_k} + \varepsilon_{CL}\overline{\lambda}_{\Phi_{Z_G,k}} m^2(k)}. \tag{4.83}$$

Remark 4.2.2. Because $\varepsilon_{NG}, \varepsilon_{CL} > 0$, and from (4.33), (4.60), and (4.73), we can say that

$$\overline{\overline{\eta}}_{NG}(k) = \frac{\overline{\eta}_{NG_s}(k)}{\varepsilon_{NG}} \text{ and } \overline{\overline{\eta}}_{NG}(k) \le \overline{\eta}_{CL}(k).$$

Hence, with $\overline{\eta}_{NG_s}(k)$ being bounded away from infinity for all time, as shown in Remark 4.1.5, so are both $\overline{\overline{\eta}}_{NG}(k)$ and $\overline{\eta}_{CL}(k)$.

Table 4.2: DTNG based CL algorithm, results, and properties.

DTNG based CL algorithm	
For $k \geq k_0$, $\phi\left(x(k)\right) \in \mathbb{R}^{r_\phi \times c_\phi}$ being the regressor, the history stack Z_G given by (3.8), $m^2(k)$ given by (4.1), $q_\epsilon(k)$ given by (2.14), $q_{Z,\epsilon}(k)$ given by (3.27), some ξ_{NG}, ξ_{CL}, $\Gamma \in \mathbb{R}^{r_\phi \times r_\phi}$ such that ξ_{NG}, ξ_{CL}, $\Gamma > 0$, and some initial guess, $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$, update $\hat{\theta}$ by applying: $m^2(k) = \alpha I_{c_\phi} + \phi^\top(x(k))\,\phi\left(x(k)\right)$, $\Delta\hat{\theta}_{NG}(k) = \phi\left(x(k)\right)\left(m^2(k)\right)^{-1} q_\epsilon(k)$, $\Delta\hat{\theta}_{CL}(k) = Z_G\, q_{Z,\epsilon}(k)$, Design $\Gamma = \Gamma^\top = \Gamma(k) > 0$ such that, $\eta = \lambda_{\max}(\Gamma)$ verifies (4.60), $\hat{\theta}(k+1) = \hat{\theta}(k) - \Gamma\left\{\xi_{NG}\Delta\hat{\theta}_{NG}(k) + \xi_{CL}\Delta\hat{\theta}_{CL}(k)\right\}$.	
SUAC results and properties	**UUAC results and properties**
Same as DTNG algorithm in the SUAC (see Table 4.1); If **Condition 3.1.1** is verified: $\tilde{\theta}(k) = [0]^{r_\theta \times c_\theta}$ is exponentially stable.	Same as DTNG algorithm in the UUAC (see Table 4.1); If **Condition 3.1.1** is verified: $\tilde{\theta}(k)$ is UUB; $\left\lVert m^{-1}(k) q_\epsilon(k)\right\rVert_F$, and $\left\lVert\Delta\tilde{\theta}(k)\right\rVert_F$ belong to $\mathcal{L}^\infty$.
Convergence Properties	
$\tilde{\theta}(k)$ converges to the origin (meaning $\hat{\theta}(k)$ converges to θ) in the SUAC or $\tilde{\theta}(k)$ converges to a neighborhood of the origin (meaning $\hat{\theta}(k)$ converges to a neighborhood of θ) as long as **Condition 3.1.1** is verified. **Condition 3.1.1** is less demanding than the PE condition on $\phi_m\left(x(k)\right)$ in (4.40) and/or requiring that vec$\left\{w\right\} \in \mathcal{L}^2$).	

Remark 4.2.3. As (4.77) and (4.81) show, $R_{NG,k}$, which is given by (4.75), dictates how fast $V(k)$ converges to zero. Solving $\dfrac{\partial R_{NG,k}}{\partial \eta}(\eta) = 0$ for η,

$$\eta = \eta_m = \frac{\overline{\eta}_{CL}(k)}{2} = \frac{1}{2\varepsilon_{NG}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2 + \varepsilon_{CL}\overline{\lambda}_{\Phi_{Z_G,k}}} \tag{4.84}$$

is the lone stationary point. Notice that $\eta_m \in (0, \overline{\eta}_{CL}(k))$. For $\dfrac{\partial R_{NG,k}}{\partial \eta} < 0$ and $\dfrac{\partial R_{NG,k}}{\partial \eta} > 0$, $\eta < \eta_m$ and $\eta > \eta_m$ respectively. Further,

$$\frac{\partial^2 R_{NG,k}}{\partial \eta^2} = \frac{2\varepsilon_{CL}\underline{\lambda}_{\Phi_{Z_G,k}}}{\eta_m} > 0$$

and, as a result, η_m is a minimum point. At η_m,

$$R_{NG,k}|_{\eta=\eta_m} = 1 - \varepsilon_{CL}\underline{\lambda}_{\Phi_{Z_G,k}}\eta_m = 1 - \frac{\varepsilon_{CL}\underline{\lambda}_{\Phi_{Z_G,k}}}{2\varepsilon_{NG}\overline{\lambda}_{\Phi_k}\left\|m^{-1}(k)\right\|_F^2 + \varepsilon_{CL}\overline{\lambda}_{\Phi_{Z_G,k}}}. \tag{4.85}$$

Because a small $R_{NG,k}$ leads to a faster convergence of $V(k)$ to zero, then, $\eta_m = \dfrac{\overline{\eta}_{CL}(k)}{2}$ is a good choice for η when applying the CL algorithm.

Remark 4.2.4. Based on the expression of $R_{NG,k}|_{\eta=\eta_m}$ above in (4.85), the higher the ratio

$$r_{CL} = \frac{\underline{\lambda}_{\Phi_{Z_G,k}}}{\overline{\lambda}_{\Phi_{Z_G,k}}} \tag{4.86}$$

is, the smaller $R_{NG,k}|_{\eta=\eta_m}$ becomes. This result, though obtained in the DT domain, supports even further the CT framework studies in [32] as wells as [15, 16, 14, 17, 18]. While applying the CL algorithm, maximizing $r_{CL} = \dfrac{\underline{\lambda}_{\Phi_{Z_G,k}}}{\overline{\lambda}_{\Phi_{Z_G,k}}}$ (as the procedures for data recording, detailed Chapter VI, reveal) can therefore be enforced when selecting the data that goes into Z_G and/or Z.

Remark 4.2.5. Given that $\overline{\lambda}_{\Phi_{Z_G,k}} \geq \underline{\lambda}_{\Phi_{Z_G,k}} > 0$ if $Z_G \in \mathbb{R}^{r_Z \times c_Z}$, $r_Z = r_\phi = r_\theta$ and $c_Z \geq r_\theta$, is full row rank, then $r_{CL} \leq 1$. If $r_{CL} = 1$ then, necessarily, $\Phi_{Z_G,k} = Z_G Z_G^\top = \overline{\varepsilon} I_{r_Z}$, for some $\overline{\varepsilon} > 0$, and, consequently, $\dfrac{1}{\sqrt{\overline{\varepsilon}}} Z_G$ is an orthogonal matrix, with Z_G being a square matrix. We thus postulate that it may be possible to get a higher r_{CL} for values of c_Z closer to $r_Z = r_\phi = r_\theta$.

This concludes our study of the DTNG algorithm and its CL modification. We will now look at Least Squares algorithm for uncertainty approximation.

CHAPTER V

DISCRETE-TIME NORMALIZED LEAST SQUARES BASED CONCURRENT LEARNING

The current chapter investigates the use of the Least Squares (LS) technique for discrete-time uncertainty approximation. The LS method minimizes the sum of instantaneous cost $J(k)$, defined in (4.2a), for all time, and, therefore, could potentially lead to a better learning generalization when compared to Gradient Descent. As in Chapter IV, here also, we only consider approximators that are linear in the parameters to identify, study both the structured and unstructured uncertainty approximation cases, as well as develop a DTNRLS based Concurrent Learning algorithm.

5.1 Discrete-Time Normalized Recursive Least-Square Algorithm for both Structured and Unstructured Uncertainty Approximation Cases

Given an initial parameter estimate $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$, for $k \geq k_0$, the discrete-time NRLS algorithm for updating the parameter estimate $\hat{\theta}$ (see (5.1d)) can be given as

$$\psi(k) = \phi\left(x(k)\right), \tag{5.1a}$$

$$K_{LS}(k) = P(k-1)\psi(k)\left(m^2(k)\right)^{-1}, \tag{5.1b}$$

$$\Delta\hat{\theta}_{LS}(k) = K_{LS}(k)Q_\epsilon(k), \tag{5.1c}$$

$$\hat{\theta}(k+1) = \hat{\theta}(k) - \Delta\hat{\theta}_{LS}(k), \tag{5.1d}$$

where, recalling $r_\phi = r_\theta$, $\psi(k) \in \mathbb{R}^{r_\psi \times c_\psi}$, i.e., $r_\psi = r_\phi = r_\theta$ and $c_\psi = c_\phi$ according to (5.1a) in this case, $K_{LS}(k) \in \mathbb{R}^{r_\psi \times c_\psi}$, the adjustment term $\Delta\hat{\theta}_{LS}(k) \in \mathbb{R}^{r_\theta \times c_\theta}$, the gain matrix $P(k) \in \mathbb{R}^{r_\psi \times r_\psi}$ is such that

$$P(k) = P(k-1) - K_{LS}(k)\psi^\top(k)P(k-1), \tag{5.2}$$

$P(k_0 - 1) = P_0 \in \mathbb{R}^{r_\psi \times r_\psi}$, P_0 chosen as a symmetric, positive-definite matrix (i.e., $P_0^\top = P_0$ and $P_0 > 0$) from the start, the normalization matrix signal $m(k) \in \mathbb{R}^{c_\psi \times c_\psi}$ is such that

$$m^2(k) = \alpha I_{c_\psi} + \psi^\top(k)P(k-1)\psi(k), \tag{5.3}$$

for some scalar $\alpha = \alpha(k) > 0$ (meaning α can be time-varying), and, given the error $q_\epsilon(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}}$ computed as in (2.14), i.e.,

$$q_\epsilon(k) = \begin{cases} q^\top(k), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ q(k), & \text{if } \mathcal{M}_n = \mathcal{M}_2, \end{cases}$$

with the approximation error $q(k)$ given by (2.9a) or (2.9b), we define the overall approximation error $Q_\epsilon(k) \in \mathbb{R}^{r_{Q_\epsilon} \times c_{Q_\epsilon}}$ as

$$Q_\epsilon(k) = q_\epsilon(k). \tag{5.4}$$

Having that $q(k) \in \mathbb{R}^{r_f \times c_f}$ and based Table on 2.1, if $\mathcal{M}_n = \mathcal{M}_1$, $r_{q_\epsilon} = c_f = c_\phi$ and $c_{q_\epsilon} = r_f = c_\theta$, whereas, if $\mathcal{M}_n = \mathcal{M}_2$, $r_{q_\epsilon} = r_f = c_\phi$ and $c_{q_\epsilon} = c_f = c_\theta$. Also, given (5.4), $r_{Q_\epsilon} = r_{q_\epsilon}$ and $c_{Q_\epsilon} = c_{q_\epsilon}$. We will see later on how r_{Q_ϵ} and c_{Q_ϵ} vary depending on the definition of Q_ϵ. Moreover, notice that, for $k \geq k_0$, we have set $\rho\,(x(k)) = \psi\,(x(k))$ and $\Xi_m = P(k-1)$ in (2.18) to get the expression of $m^2(k)$ given in this case by (5.3). At $k = k_0$, we have that $\Xi_m = P(k_0 - 1) = P_0 > 0$ by definition. Requiring that $\Xi_m > 0$ as initially intended when defining the normalization signal, we will show later on that $P(k) > 0$ for any $k \geq k_0$.

We introduce the matrix $\Psi_k \in \mathbb{R}^{r_\psi \times r_\psi}$, which we define as

$$\Psi_k = \psi(k)\psi^\top(k). \tag{5.5}$$

Based on (5.5), Ψ_k is symmetric, i.e., $\Psi_k^\top = \Psi_k$, (at least) positive semidefinite, i.e., $\Psi_k \geq 0$, and, for all k, we can thus write

$$\underline{\lambda}_{\Psi_k}(k) I_{r_\psi} \leq \Psi_k \leq \overline{\lambda}_{\Psi_k}(k) I_{r_\psi}, \tag{5.6}$$

for some finite

$$\underline{\lambda}_{\Psi_k}(k) = \lambda_{\min}(\Psi_k)\,, \overline{\lambda}_{\Psi_k}(k) = \lambda_{\max}(\Psi_k)\,.$$

Notice also that, in this case, given (2.27) and (5.5), $\Psi_k = \Phi_k$ if in fact $\psi(k)$ is as given by (5.1a). It should be added that, though seemingly redundant and unnecessary, reasons for defining $\psi(k)$, $Q_\epsilon(k)$, and Ψ_k will become clearer soon, as we look to tie together this section and the next one, where we present our main results.

Expression (5.2) of $P(k)$ will be derived in the upcoming sections. For now, first, notice that $P(k)$ is symmetric for all k. In fact, given that

$$P(k_0 - 1) = P_0 = P_0^\top = P^\top(k_0 - 1)$$

and using (5.2), showing that, for all k, $P(k) = P^\top(k)$ can be done by induction, starting with the computation of $P(k_0)$ and $P^\top(k_0)$. Second, with $P(k)$ being symmetric, then $m(k)$ is also symmetric (see expression (5.3) of $m^2(k)$ for reason as to why), i.e., $m^\top(k) = m(k)$; and, its inverse, which, for now, we can assume to exist for the sake of the argument to follow, is also symmetric, i.e., $\left(m^{-1}(k)\right)^\top = m^{-1}(k)$. Actually, $m(k)$ or $m^2(k)$ can be shown to be invertible by induction, with the premise that $P(k_0 - 1) = P_0 > 0$. Third, $P(k)$ can be rewritten as

$$P(k) = P_A + P_U \left(-I_{c_\psi}\right) P_V,$$

where, given both (5.1b) and (5.2), $P_A \in \mathbb{R}^{r_\psi \times r_\psi}$, $P_U \in \mathbb{R}^{r_\psi \times c_\psi}$, and $P_V \in \mathbb{R}^{c_\psi \times r_\psi}$ are such that

$$\begin{aligned}
&P_A = P(k-1),\\
&P_U = P_A \phi\left(x(k)\right) m^{-1}(k) = P_A \psi(k) m^{-1}(k), \text{ and}\\
&P_V = P_U^\top = m^{-1}(k) \phi^\top\left(x(k)\right) P_A = m^{-1}(k)\psi^\top(k) P_A.
\end{aligned}$$

Hence, assuming for the time being that, for any k, $P(k)$ is invertible (or, alternatively, proceeding with a proof by induction instead to prove invertibility of $P(k)$) so as to be able to use the Woodbury Matrix Identity (A.1), we get

$$P^{-1}(k) = P_A^{-1} - P_A^{-1} P_U \left(-I_{c_\psi} + P_V P_A^{-1} P_U\right)^{-1} P_V P_A^{-1}. \tag{5.7}$$

Let $P_D \in \mathbb{R}^{r_\psi \times c_\psi}$ and $P_E \in \mathbb{R}^{c_\psi \times c_\psi}$ be such that

$$P_D = P_A^{-1} P_U = \psi(k) m^{-1}(k) \text{ and } P_E = \psi^\top(k) P_A \psi(k);$$

and, notice that

$$P_D^\top = P_V P_A^{-1} = m^{-1}(k)\psi^\top(k)$$

and

$$P_V P_A^{-1} P_U = m^{-1}(k) P_E m^{-1}(k).$$

Now, using (5.3), $m^2(k) = \alpha I_{c_\psi} + P_E$ and

$$\begin{aligned}
P_G &= -I_{c_\phi} + P_V P_A^{-1} P_U\\
&= -I_{c_\psi} + m^{-1}(k) P_E m^{-1}(k) = -m^{-1}(k)\left(m^2(k)\right) m^{-1}(k) + m^{-1}(k) P_E m^{-1}(k)\\
&= -m^{-1}(k)\left(m^2(k) - P_E\right) m^{-1}(k) = -m^{-1}(k)\left(\left(\alpha I_{c_\psi} + P_E\right) - P_E\right) m^{-1}(k)\\
&= -\alpha\left(m^2(k)\right)^{-1}.
\end{aligned}$$

Based on our definitions above, (5.5), and given (5.7),

$$P^{-1}(k) = P_A^{-1} - P_D \left(P_G\right)^{-1} P_D^\top = P^{-1}(k-1) + \frac{1}{\alpha}\Psi_k, \tag{5.8}$$

meaning

$$P^{-1}(k) = P^{-1}(k_0 - 1) + \Xi_\Psi(k) = P_0^{-1} + \Xi_\Psi(k), \tag{5.9}$$

where, for $k > k_0$, we define $\Xi_\Psi \in \mathbb{R}^{r_\psi \times r_\psi}$ as

$$\Xi_\Psi(k) = \sum_{\tau=k_0}^{k} \frac{1}{\alpha}\Psi_\tau. \tag{5.10}$$

With $\alpha > 0$ and $\Psi_k \geq 0$ for all k, $\Xi_\Psi(k) \geq 0$. Further, $P_0 > 0$ (and, as a result, $P_0^{-1} > 0$) and given that $P(k)$ is symmetric, (5.9) allows us to write

$$P^{-1}(k) = \left(P^{-1}(k)\right)^\top \geq P^{-1}(k_0 - 1) = P_0^{-1} > 0 \tag{5.11}$$

for all k. Subsequently,

$$0 < P(k) = P^\top(k) \leq P_0. \tag{5.12}$$

That is, $P(k) = P^\top(k)$ is positive definite and bounded for all k so long as P_0 is bounded, which should be the case in practice. With $P(k)$ being positive definite, then, $m(k)$ (or $m^2(k)$ given by (5.3)) is also positive definite. These results thus confirm the existence of both $P^{-1}(k)$ and $m^{-1}(k)$, which, as mentioned above, could also have been proven by induction. Moreover, notice that, given (5.3), (5.6), and (5.12), there exist some scalar $\beta_{IM} \geq 0$ for which

$$\alpha I_{c_\psi} \leq m^2(k) \leq (\alpha + \beta_{IM}) I_{c_\psi}. \tag{5.13}$$

Before looking any further into each approximation case, notice that, with (2.9b), (2.13), (5.1a), (2.14), and regardless of the approximation models in (2.2), the error $Q_\epsilon(k)$ in (5.4) can also be rewritten as

$$Q_\epsilon(k) = q_\epsilon(k) = \phi^\top(x(k))\,\hat{\theta}(k) - f_n(k) \tag{5.14a}$$

$$= \psi^\top(k)\hat{\theta}(k) - f_n(k),$$

$$= \psi^\top(k)\tilde{\theta}(k) - w_\epsilon(k), \tag{5.14b}$$

where the function $f_n(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}} = \mathbb{R}^{r_{Q_\epsilon} \times c_{Q_\epsilon}}$ and the residual error $w_\epsilon(k) \in \mathbb{R}^{r_{Q_\epsilon} \times c_{Q_\epsilon}}$ are as given is (2.16) and (2.17), i.e.,

$$f_n(k) = \begin{cases} f^\top(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ f(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_2, \end{cases}$$

and

$$w_\epsilon(k) = \begin{cases} w^\top(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_1, \\ w(x(k)), & \text{if } \mathcal{M}_n = \mathcal{M}_2, \end{cases}$$

respectively. Additionally, from (5.2),

$$P(k)P^{-1}(k-1) = I_{r_\psi} - K_{LS}(k)\psi^\top(k). \tag{5.15}$$

Thus, considering the DTNRLS update law (5.1), $Q_\epsilon(k)$ in (5.14b), and (5.15), we express the parameter error defined in (2.10) as

$$\tilde{\theta}(k+1) = \left[I_{r_\theta} - K_{LS}(k)\psi^\top(k)\right]\tilde{\theta}(k) + K_{LS}(k)w_\epsilon(k) \tag{5.16a}$$

$$= P(k)P^{-1}(k-1)\tilde{\theta}(k) + K_{LS}(k)w_\epsilon(k). \tag{5.16b}$$

5.1.1 DTNRLS for the SUAC

If f is a structured uncertainty, then, we can set $w(x(k)) = [0]^{r_f \times c_f}$ or, equivalently, given (2.17), $w(x(k)) = [0]^{r_{Q_\epsilon} \times c_{Q_\epsilon}}$ for all k. Based on (2.1) and (2.3), that essentially means $f(x(k))$ is given as in (4.20), i.e.,

$$f(x(k)) = \mathcal{F}(x(k), \theta) = \mathcal{M}_n(x(k), \theta),$$

and, from (5.14b),

$$Q_\epsilon(k) = \psi^\top(k)\tilde{\theta}(k). \tag{5.17}$$

Moreover, the parameter error equations of (5.16) reduce to

$$\tilde{\theta}(k+1) = \left[I_{r_\theta} - K_{LS}(k)\psi^\top(k)\right]\tilde{\theta}(k) \tag{5.18a}$$

$$= P(k)P^{-1}(k-1)\tilde{\theta}(k). \tag{5.18b}$$

For analysis purposes, consider the Lyapunov Function (2.24), with

$$\Xi_V = \Xi_V^\top = P^{-1}(k-1) = \left(P^{-1}(k-1)\right)^\top \geq P_0^{-1} > 0$$

considering (5.11), i.e.,

$$V(k) = \text{tr}\left\{\tilde{\theta}^\top(k)P^{-1}(k-1)\tilde{\theta}(k)\right\}. \tag{5.19}$$

Along (5.18b) and given (5.19),

$$V(k+1) = \text{tr}\left\{\tilde{\theta}^\top(k+1)P^{-1}(k)\tilde{\theta}(k+1)\right\} = \text{tr}\left\{\tilde{\theta}^\top(k+1)P^{-1}(k-1)\tilde{\theta}(k)\right\}.$$

Hence, using (2.26) and (5.19),

$$\Delta V(k+1) = V(k+1) - V(k) = \text{tr}\left\{\Delta\tilde{\theta}^\top(k+1)P^{-1}(k-1)\tilde{\theta}(k)\right\}, \tag{5.20}$$

where $\Delta\tilde{\theta} \in \mathbb{R}^{r_\theta \times r_\theta}$ is as defined in (4.43), i.e., such that, for values of $k \geq k_0$,

$$\Delta\tilde{\theta}(k+1) = \tilde{\theta}(k+1) - \tilde{\theta}(k).$$

From (5.1b) and (5.18a),

$$\Delta\tilde{\theta}(k+1) = -K_{LS}(k)\psi^\top(k)\tilde{\theta}(k) = -P(k-1)\psi(k)\left(m^2(k)\right)^{-1}\psi^\top(k)\tilde{\theta}(k).$$

In light of that, (5.20) can be rewritten as

$$\Delta V(k+1) = -\text{tr}\left\{\tilde{\theta}^\top(k)\psi(k)\left(m^2(k)\right)^{-1}\psi^\top(k)\tilde{\theta}(k)\right\},$$

which, because of (5.17), also means

$$\Delta V(k+1) = -\text{tr}\left\{Q_\epsilon^\top(k)\left(m^2(k)\right)^{-1}Q_\epsilon(k)\right\} = -\text{tr}\left\{\left[m^{-1}(k)Q_\epsilon(k)\right]^\top\left[m^{-1}(k)Q_\epsilon(k)\right]\right\}$$

or, using (C.9),

$$\Delta V(k+1) = -\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F^2 \leq 0. \tag{5.21}$$

The result in (5.21) allows us to conclude that $V(k)$ is bounded for all k. Now, given (5.9) and (5.19), we can write

$$V(k) = \text{tr}\left\{\tilde{\theta}^\top(k)P_0^{-1}\tilde{\theta}(k)\right\} + \text{tr}\left\{\tilde{\theta}^\top(k)\Xi_\Psi(k-1)\tilde{\theta}(k)\right\}.$$

Therefore, with $V(k)$ being bounded and $\Xi_\Psi(k) \geq 0$ for all k,

$$\text{tr}\left\{\tilde{\theta}^\top(k)P_0^{-1}\tilde{\theta}(k)\right\} = V(k) - \text{tr}\left\{\tilde{\theta}^\top(k)\Xi_\Psi(k-1)\tilde{\theta}(k)\right\} \leq V(k)$$

is also bounded and, further, so is $\tilde{\theta}(k)$. Considering (5.3), (C.8), (C.10), and (C.11), we can find some scalar $\overline{\alpha}_{m,2}$ such that $0 < \overline{\alpha}_{m,2} \leq 1$ and

$$\|m(k)\|_F \geq \overline{\alpha}_{m,2}\left(\alpha\left\|I_{c_\psi}\right\|_F + \|P(k-1)\|_F\|\psi(k)\|_F^2\right)^{\frac{1}{2}}. \tag{5.22}$$

Hence, from (5.22), as well as (2.14), (5.4), (5.17), (C.8), (C.10), and (C.11),

$$\frac{\|Q_\epsilon(k)\|_F}{\|m(k)\|_F} = \frac{\|q(k)\|_F}{\|m(k)\|_F} \leq \frac{\left\|\tilde{\theta}(k)\right\|_F\Lambda_{P,k}^{-\frac{1}{2}}\Lambda_{\psi,k}}{\overline{\alpha}_{m,2}\left(\alpha\left\|I_{c_\psi}\right\|_F + \Lambda_{\psi,k}^2\right)^{\frac{1}{2}}} \tag{5.23}$$

and, using (C.13), for some scalars β_{q_ϵ} and $\beta_{\psi,1}$, such that $1 < \beta_{q_\epsilon} < \infty$ and $1 < \beta_{\psi,1} < \infty$,

$$\begin{aligned}\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F &\leq \left\|m^{-1}(k)\right\|_F\|Q_\epsilon(k)\|_F \\ &< \beta_{q_\epsilon}\left\|I_{c_\psi}\right\|_F\frac{\|Q_\epsilon(k)\|_F}{\|m(k)\|_F}\end{aligned} \tag{5.24}$$

and, for $\rho(x(k)) = \psi(k)$ in (2.23), $\psi_m(k) \in \mathbb{R}^{r_\psi \times c_\psi}$ is defined as

$$\psi_m(k) = \psi(k)m^{-1}(k), \tag{5.25}$$

with

$$\|\psi_m(k)\|_F = \left\|\psi(k)m^{-1}(k)\right\|_F \leq \|\psi(k)\|_F \left\|m^{-1}(k)\right\|_F < \beta_{\psi,1} \left\|I_{c_\psi}\right\|_F \frac{\|\psi(k)\|_F}{\|m(k)\|_F}$$
$$< \beta_{\psi,1} \left\|I_{c_\psi}\right\|_F \frac{\Lambda_{P,k}^{-\frac{1}{2}} \Lambda_{\psi,k}}{\overline{\alpha}_{m,2} \left(\alpha \left\|I_{c_\psi}\right\|_F + \Lambda_{\psi,k}^2\right)^{\frac{1}{2}}}, \tag{5.26}$$

where

$$\Lambda_{P,k} = \|P(k-1)\|_F \text{ and} \tag{5.27a}$$

$$\Lambda_{\psi,k} = \|P(k-1)\|_F^{\frac{1}{2}} \|\psi(k)\|_F = \Lambda_{P,k}^{\frac{1}{2}} \|\psi(k)\|_F . \tag{5.27b}$$

We also have from (2.26) and (5.21) that, for values of $k > k_0$,

$$V(k+1) = V(k_0) - \sum_{\tau=k_0}^{k} \left\|m^{-1}(\tau)Q_\epsilon(\tau)\right\|_F^2 ,$$

and, thus, with $V(k)$ being nonincreasing for all k as a result of (5.21),

$$\sum_{\tau=k_0}^{k} \left\|m^{-1}(\tau)Q_\epsilon(\tau)\right\|_F^2 = V(k_0) - V(k+1) \leq V(k_0), \tag{5.28}$$

where, according to (5.19),

$$V(k_0) = \text{tr}\left\{\tilde{\theta}^\top(k_0) \left(P_0\right)^{-1} \tilde{\theta}(k_0)\right\}$$

is finite so long as $\left\|\hat{\theta}(k_0)\right\|_F$ is. Consequently, first, given that $\tilde{\theta}(k)$ and $P(k)$ are bounded for all k, $\dfrac{\|q(k)\|_F}{\|m(k)\|_F}$, $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F$, and $\left\|\psi(k)m^{-1}(k)\right\|_F$ also are. As a side note, given (5.3) and (5.22), there exists at least an instance of $\alpha_{m,2}$ that is closer to 1 tan 0 so that the previously mentioned boundedness results hold. Also, ensuring the boundedness of $\psi_m(k) = \psi(k)m^{-1}(k)$ is the reasoning behind the use of the normalization signal $m(k)$. Second, even better than its $\mathcal{L}^\infty$ property is the $\mathcal{L}^2$ property of $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F$, or, equivalently, given (C.9), vec $\left\{m^{-1}(k)Q_\epsilon(k)\right\} \in \mathcal{L}^2$, which is obtained from (5.28). Refer to Appendix C for definitions of $\mathcal{L}^p$, with $p \in [1, \infty)$, and $\mathcal{L}^\infty$ signal spaces. Continuing and referring back to the algorithm in (5.1), while also using (2.10) and

(4.43), we have that

$$\left\|\Delta\tilde{\theta}(k+1)\right\|_F = \left\|\tilde{\theta}(k+1) - \tilde{\theta}(k)\right\|_F = \left\|P(k-1)\psi(k)\left(m^2(k)\right)^{-1} Q_\epsilon(k)\right\|_F$$
$$\leq \Lambda_{P,k}\left\|\psi(k)m^{-1}(k)\right\|_F \left\|m^{-1}(k)Q_\epsilon(k)\right\|_F. \tag{5.29}$$

Hence, not only is $\left\|\Delta\tilde{\theta}(k)\right\|_F$ bounded since $\left\|\psi(k)m^{-1}(k)\right\|_F \in \mathcal{L}^\infty$ and $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F \in \mathcal{L}^\infty$, but, with $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F \in \mathcal{L}^2$, we also have $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$ or, correspondingly, due to (C.9), vec $\left\{\Delta\tilde{\theta}(k)\right\} \in \mathcal{L}^2$.

Now, notice, using (5.2), that, for all k,

$$P(k) = P(k_0 - 1) - \Xi_{P,1}(k) = P_0 - \Xi_{P,1}(k), \tag{5.30}$$

where, with $\Xi_{P,2} \in \mathbb{R}^{r_\psi \times r_\psi}$ given by

$$\Xi_{P,2}(k) = K_{LS}(k)\psi^\top(k)P(k-1).$$

which, from the expression of $K_{LS}(k)$ in (5.1b) and because, for all k, $P(k) > 0$ and $m^2(k) > 0$ as (5.12) and (5.13) respectively show, also means

$$\Xi_{P,2}(k) = P(k-1)\psi(k)\left(m^2(k)\right)^{-1}\psi^\top(k)P(k-1) \geq 0,$$

we define $\Xi_{P,1} \in \mathbb{R}^{r_\psi \times r_\psi}$ as

$$\Xi_{P,1}(k) = \sum_{\tau=k_0}^{k} \Xi_{P,2}(\tau).$$

Let $z_a \in \mathbb{R}^{r_\psi}$ be an arbitrary, constant vector. Also, we define the scalar function

$$h_{z_a}(k) = z_a^\top \Xi_{P,1}(k) z_a.$$

With $P(k) > 0$ for all k, we get from (5.30) that

$$z_a^\top P(k) z_a = z_a^\top P_0 z_a - h_{z_a}(k) \geq 0; \tag{5.31}$$

thus, resulting in h_{z_a} being bounded from above, as (5.31) leads to $h_{z_a}(k) \leq z_a^\top P_0 z_a$. It can also be shown that

$$h_{z_a}(k+1) - h_{z_a}(k) = z_a^\top \Xi_{P,2}(k+1) z_a \geq 0,$$

since $\Xi_{P,2}(k) \geq 0$ for any k, and, therefore, h_{z_a} is monotonically increasing. Both properties of h_{z_a}, i.e., boundedness from above and monotonic growth, imply that $\lim_{k\to\infty} h_{z_a}(k)$ exist and is finite. We have essentially applied **Proposition A.1** from Appendix A. Further, given that P_0 is bounded (at least in practice), $z_a^\top P_0 z_a$ is also bounded and finite. Hence, given (5.31), $\lim_{k\to\infty} z_a^\top P(k) z_a$ exists and is finite. Since z_a is constant and arbitrary, we can conclude that there exists a constant $P_{ss} \in \mathbb{R}^{r_\psi \times r_\psi}$ such that

$$\lim_{k\to\infty} P(k) = P_{ss}. \tag{5.32}$$

Now, considering (5.18b), for all $k > k_0$,

$$\tilde{\theta}(k) = P(k-1)P^{-1}(k_0-1)\tilde{\theta}(k_0) = P(k-1)P_0^{-1}\tilde{\theta}(k_0). \tag{5.33}$$

Recalling the definition of $\tilde{\theta}(k)$ in (2.10),

$$\lim_{k\to\infty} \hat{\theta}(k) = \theta + \lim_{k\to\infty} \tilde{\theta}(k) = \theta + \lim_{k\to\infty} P(k-1)P_0^{-1}\tilde{\theta}(k_0),$$

with θ, as defined in (2.4), being constant. As a result, because P_0 is constant and due to (5.32), there also exists a constant $\hat{\theta}_{ss} \in \mathbb{R}^{r_\theta \times c_\theta}$ that is such that

$$\lim_{k\to\infty} \hat{\theta}(k) = \hat{\theta}_{ss}. \tag{5.34}$$

This however does not mean that $\hat{\theta}_{ss} = \theta$ is to be expected. From (5.12), $P(k-1)P_0^{-1} \leq I_{r_\psi}$, and, using (5.33), for all k,

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{LS_s} = \left\|I_{r_\psi}\right\|_F \left\|\tilde{\theta}(k_0)\right\|_F = \sqrt{r_\psi}\left\|\tilde{\theta}(k_0)\right\|_F. \tag{5.35}$$

5.1.2 DTNRLS for the UUAC

Here, we can no longer assume that, for all k, $w\left(x(k)\right) = [0]^{r_f \times c_f}$ or, equivalently, $w_\epsilon(k) = [0]^{r_{Q_\epsilon} \times c_{Q_\epsilon}}$. We go back to $Q_\epsilon(k)$ and $\tilde{\theta}(k)$ respectively given by (5.14b) and (5.16). Analyzing the trajectory of $\tilde{\theta}(k)$ is nevertheless done in a similar fashion to what we did in Section 5.1.1.

With (5.16b) and (5.19), we get that

$$V(k+1) = \text{tr}\left\{\tilde{\theta}^\top(k+1)P^{-1}(k-1)\tilde{\theta}(k) + \Upsilon_{\epsilon,k}\right\},$$

where

$$\Upsilon_{\epsilon,k} = \tilde{\theta}^\top(k+1)\left(P(k)\right)^{-1}K_{LS}(k)w_\epsilon(k). \tag{5.36}$$

That means, according to (2.26) and (5.19), that

$$\Delta V(k+1) = \text{tr}\left\{\Delta\tilde{\theta}^\top(k+1)P^{-1}(k-1)\tilde{\theta}(k) + \Upsilon_{\epsilon,k}\right\}. \tag{5.37}$$

From (5.1b) and (5.16a), while using (4.43),

$$\begin{aligned}\Delta\tilde{\theta}(k+1) &= -K_{LS}(k)\psi^\top(k)\tilde{\theta}(k) + K_{LS}(k)w_\epsilon(k)\\ &= -P(k-1)\psi(k)\left(m^2(k)\right)^{-1}\psi^\top(k)\tilde{\theta}(k) + P(k-1)\psi(k)\left(m^2(k)\right)^{-1}w_\epsilon(k).\end{aligned}$$

Therefore, (5.37) can be rewritten as

$$\begin{aligned}\Delta V(k+1) = &-\text{tr}\left\{\tilde{\theta}^\top(k)\psi(k)\left(m^2(k)\right)^{-1}\psi^\top(k)\tilde{\theta}(k)\right\}\\ &+\text{tr}\left\{w_\epsilon^\top(k)\left(m^2(k)\right)^{-1}\psi^\top(k)\tilde{\theta}(k) + \Upsilon_{\epsilon,k}\right\}.\end{aligned} \tag{5.38}$$

Let $\bar{\Upsilon}_{P,k} \in \mathbb{R}^{c_\psi \times c_\psi}$ be defined as

$$\bar{\Upsilon}_{P,k} = m^{-1}(k)\psi^\top(k)\Upsilon_{P,k}\psi(k)m^{-1}(k) = \left[\psi(k)m^{-1}(k)\right]^\top\Upsilon_{P,k}\left[\psi(k)m^{-1}(k)\right],$$

where $\Upsilon_{P,k} \in \mathbb{R}^{r_\psi \times r_\psi}$ is

$$\Upsilon_{P,k} = P(k-1)\left(P(k)\right)^{-1}P(k-1),$$

which, because of (5.8), also means

$$\Upsilon_{P,k} = P(k-1) + \frac{1}{\alpha} P(k-1)\Psi_k P(k-1).$$

Since, for all k, $\Psi_k \geq 0$, given (5.12), with P_0 being bounded, and because $\left\| \psi(k) m^{-1}(k) \right\|_F \in \mathcal{L}^\infty$ (as shown above, towards the end of Section 5.1.2) means $\psi(k)m^{-1}(k)$ is bounded, we can say that there exists some scalar $\beta_P > 0$ for which

$$\bar{\Upsilon}_{P,k} \leq \beta_P I_{c_\psi}.$$

While using (5.1b) and substituting the expression of $\tilde{\theta}(k+1)$ given by (5.16b) in (5.36), we get that

$$\begin{aligned}\Upsilon_{\epsilon,k} &= \tilde{\theta}^\top(k)\psi(k)\left(m^2(k)\right)^{-1} w_\epsilon(k) + w_\epsilon^\top(k) m^{-1}(k) \bar{\Upsilon}_{P,k} m^{-1}(k) w_\epsilon(k) \\ &\leq \tilde{\theta}^\top(k)\psi(k)\left(m^2(k)\right)^{-1} w_\epsilon(k) + \beta_P w_\epsilon^\top(k)\left(m^2(k)\right)^{-1} w_\epsilon(k),\end{aligned}$$

which, going back to (5.38) and "completing the square", in turn means

$$\begin{aligned}\Delta V(k+1) &\leq -\text{tr}\left\{Q_\epsilon^\top(k)\left(m^2(k)\right)^{-1} Q_\epsilon(k)\right\} + \beta(k)\,\text{tr}\left\{w_\epsilon^\top(k)\left(m^2(k)\right)^{-1} w_\epsilon(k)\right\} \\ &\leq -\text{tr}\left\{\left[m^{-1}(k)Q_\epsilon(k)\right]^\top \left[m^{-1}(k)Q_\epsilon(k)\right]\right\} \\ &\quad + \beta(k)\,\text{tr}\left\{\left[m^{-1}(k)w_\epsilon(k)\right]^\top \left[m^{-1}(k)w_\epsilon(k)\right]\right\},\end{aligned}$$

with $Q_\epsilon(k)$ given by (5.14b) and $\beta(k) = 1 + \beta_P > 0$. Further, using (C.9),

$$\Delta V(k+1) \leq -\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F^2 + \beta(k)\left\|m^{-1}(k)w_\epsilon(k)\right\|_F^2, \tag{5.39}$$

$$\leq \beta(k)\left\|m^{-1}(k)w_\epsilon(k)\right\|_F^2, \tag{5.40}$$

since $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F^2 \geq 0$. From (2.26) and (5.40), it can be shown for $k \geq k_0$ that

$$V(k+1) \leq V(k_0) + \sum_{\tau=k_0}^{k} \beta(\tau)\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2.$$

Provided that there exists a finite scalar $B_{w_\epsilon,2} \geq 0$ that is such that

$$\sum_{\tau=0}^{\infty} \beta(\tau)\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2 \leq B_{w_\epsilon,2} < \infty, \tag{5.41}$$

then, for all $k > k_0$,

$$V(k+1) \leq V(k_0) + B_{w_\epsilon,2}, \tag{5.42}$$

meaning $V(k)$ is bounded, which, as shown in Section 5.1.1, leads to $\tilde{\theta}(k)$ being also bounded. Based on (5.42), we use the definition of $V(k)$ in (5.19), the lower bound of $P^{-1}(k)$ in (5.11), and relationship (C.9) together with the fact that, with both $P_0 > 0$ and $P_0^{-1} > 0$,

$$\underline{\lambda}_0 I_{r_\psi} \leq P_0 \leq \overline{\lambda}_0 I_{r_\psi} \text{ and } \overline{\lambda}_0 I_{r_\psi} \leq P_0^{-1} \leq \underline{\lambda}_0 I_{r_\psi},$$

where we denote

$$\underline{\lambda}_0 = \lambda_{\min}(P_0) = \lambda_{\max}\left(P_0^{-1}\right), \overline{\lambda}_0 = \lambda_{\max}(P_0) = \lambda_{\min}\left(P_0^{-1}\right),$$

to arrive at

$$\overline{\lambda}_0 \left\|\tilde{\theta}(k+1)\right\|_F^2 \leq V(k+1) \leq V(k_0) + B_{w_\epsilon,2} \leq \underline{\lambda}_0 \left\|\tilde{\theta}(k_0)\right\|_F^2 + B_{w_\epsilon,2}.$$

Therefore, if in fact (5.41) is verified,

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{LS_u} = \sqrt{\overline{\lambda}_0^{-1} \underline{\lambda}_0 \left\|\tilde{\theta}(k_0)\right\|_F^2 + \overline{\lambda}_0^{-1} B_{w_\epsilon,2}} \tag{5.43}$$

for all k.

Given that $\beta(k)$ is finite and, according to (5.13), $m(k)$ is bounded from above and below, condition (5.41) implies that both vec $\{w_\epsilon(k)\}$ and vec $\{w\left(x(k)\right)\}$ (recall (2.17)) possess $\mathcal{L}^2$ properties, i.e., vec $\{w_\epsilon(k)\} \in \mathcal{L}^2$ and, equivalently, vec $\{w\left(x(k)\right)\} \in \mathcal{L}^2$. We have previously only required that vec $\{w\left(x(k)\right)\} \in \mathcal{L}^\infty$ (see (2.7)). Hence, necessitating (5.41) is quite restrictive.

Notice also that Θ_{LS_u} in (5.43), which, in this case, also serves as an ultimate bound to $\left\|\tilde{\theta}(k)\right\|_F$, is function of the initial value $\left\|\tilde{\theta}(k_0)\right\|_F$. Hence, the larger $\left\|\tilde{\theta}(k_0)\right\|_F$ is or, equivalently, the farther $\hat{\theta}_0$ is picked (unknowingly) from the true θ, the larger the size of the neighborhood (at least in theory) in which $\left\|\tilde{\theta}(k)\right\|_F$ could ultimately end up in.

Assuming (5.41) is satisfied, which means $\tilde{\theta}(k)$ is bounded, as done in Section 5.1.1, we now proceed to bounding other approximation related signals. The process is similar to before. The only difference, in this case, is the presence of the representation error. Recall (2.7), which features W as the upper bound to $\|w(x(k))\|_F$. Because of (2.17) and the fact that, according to (C.8), $\|w(x(k))\|_F = \|w^\top(x(k))\|_F$, W is also the upper bound to $\|w_\epsilon(k)\|_F$. Given (5.3), (2.14), (5.4), (5.14b), (5.22), (C.8), (C.10), and (C.11),

$$\frac{\|Q_\epsilon(k)\|_F}{\|m(k)\|_F} = \frac{\|q(k)\|_F}{\|m(k)\|_F} \leq \frac{\left\|\tilde{\theta}(k)\right\|_F \Lambda_{P,k}^{-\frac{1}{2}} \Lambda_{\psi,k} + W}{\overline{\alpha}_{m,2}\left(\alpha \left\|I_{c_\psi}\right\|_F + \Lambda_{\psi,k}^2\right)^{\frac{1}{2}}},$$

where $\Lambda_{P,k}$ and $\Lambda_{\psi,k}$ are as defined in (5.27a) and (5.27b) respectively. Consequently, we have that $\frac{\|Q_\epsilon(k)\|_F}{\|m(k)\|_F}$ is also bounded here. With (5.24), (5.26), and (5.29), which, by the way, remain unchanged even in this case, we conclude that $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F$, $\left\|\psi(k)m^{-1}(k)\right\|_F$, and $\left\|\Delta\tilde{\theta}(k)\right\|_F$ are all bounded. If we were to go back to (5.39), while making use of (2.26), we could write that

$$V(k+1) \leq V(k_0) - \sum_{\tau=k_0}^{k} \left\|m^{-1}(\tau)Q_\epsilon(\tau)\right\|_F^2 + \sum_{\tau=k_0}^{k} \beta(\tau)\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2$$

and

$$\begin{aligned}\sum_{\tau=k_0}^{k} \left\|m^{-1}(\tau)Q_\epsilon(\tau)\right\|_F^2 &\leq -V(k+1) + V(k_0) + \sum_{\tau=k_0}^{k} \beta(\tau)\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2 \\ &\leq V(k_0) + \sum_{\tau=k_0}^{k} \beta(\tau)\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2,\end{aligned}$$

since, for all k, $V(k)$ in (5.19) is positive definite. Hence, if (5.41) is verified,

$$\begin{aligned}\sum_{\tau=k_0}^{\infty} \left\|m^{-1}(\tau)Q_\epsilon(\tau)\right\|_F^2 &= \lim_{k\to\infty} \sum_{\tau=k_0}^{k} \left\|m^{-1}(\tau)Q_\epsilon(\tau)\right\|_F^2 \\ &\leq V(k_0) + \lim_{k\to\infty} \sum_{\tau=k_0}^{k} \beta(\tau)\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2 \\ &\leq V(k_0) + \sum_{\tau=k_0}^{\infty} \beta(\tau)\left\|m^{-1}(\tau)w_\epsilon(\tau)\right\|_F^2 \leq V(k_0) + B_{w_\epsilon,2},\end{aligned}$$

which with $V(k_0)$ being bounded means $\left\|m^{-1}(\tau)Q_\epsilon(\tau)\right\|_F \in \mathcal{L}^2$.

Having $P(k) \to P_{ss}$ as $k \to \infty$ also applies in this case, as the manner in which $P(k)$ is computed here, i.e., (5.2), remains unchanged. As for the convergence of $\hat{\theta}(k)$ in this case, from (5.16b), we get, for all $k > k_0$, that

$$\begin{aligned}\tilde{\theta}(k) &= P(k-1)P^{-1}(k_0-1)\tilde{\theta}(k_0) + \Xi_{\epsilon,1}(k) \\ &= P(k-1)P_0^{-1}\tilde{\theta}(k_0) + \Xi_{\epsilon,1}(k), \end{aligned} \tag{5.44}$$

where $\Xi_{\epsilon,1} \in \mathbb{R}^{r_\theta \times c_\theta}$ is defined as

$$\Xi_{\epsilon,1}(k) = P(k-1)\sum_{\tau=k_0}^{k-2} \Xi_{\epsilon,2}(\tau) w_\epsilon(\tau) + K_{LS}(k-1)w_\epsilon(k-1), \tag{5.45}$$

with $\Xi_{\epsilon,2} \in \mathbb{R}^{r_\psi \times c_\psi}$ given by

$$\Xi_{\epsilon,2}(k) = P^{-1}(k)K_{LS}(k).$$

Making use of the definition of $K_{LS}(k)$ in (5.1b) and, then, relationship (5.15),

$$\begin{aligned}\Xi_{\epsilon,2}(k) &= \left[P^{-1}(k)P(k-1)\right]\psi(k)\left(m^2(k)\right)^{-1} \\ &= \psi(k)\left(m^2(k)\right)^{-1} - K_{LS}(k)\psi^\top(k)\psi(k)\left(m^2(k)\right)^{-1}. \end{aligned} \tag{5.46}$$

Based on (5.1b), (5.46), (C.10), and (C.11) ,

$$\begin{aligned}\Lambda_{k,1} &= \left\|\psi(k)\left(m^2(k)\right)^{-1}\right\|_F \le \left\|\psi(k)m^{-1}(k)\right\|_F \left\|m^{-1}(k)\right\|_F, \\ \Lambda_{k,2} &= \left\|K_{LS}(k)\right\|_F \le \left\|P(k-1)\right\|_F \left\|\psi(k)\left(m^2(k)\right)^{-1}\right\|_F = \left\|P(k-1)\right\|_F \Lambda_{k,1},\end{aligned}$$

and

$$\left\|\Xi_{\epsilon,2}(k)\right\|_F \le \Lambda_{k,1} + \Lambda_{k,2}\left\|\psi^\top(k)\psi(k)\left(m^2(k)\right)^{-1}\right\|_F.$$

Additionally, given (5.22), (C.7), and (C.13), we could find some $\beta_{\psi,2}$ such that $1 < \beta_{\psi,2} < \infty$ and

$$\begin{aligned}\left\|\psi^\top(k)\psi(k)\left(m^2(k)\right)^{-1}\right\|_F &\le \left\|\psi(k)\right\|_F^2 \left\|m^{-1}(k)\right\|_F^2 < \beta_{\psi,2}\left\|I_{c_\psi}\right\|_F \left(\frac{\left\|\psi(k)\right\|_F}{\left\|m(k)\right\|_F}\right)^2 \\ &< \beta_{\psi,2}\left\|I_{c_\psi}\right\|_F \frac{\Lambda_{P,k}^{-1}\Lambda_{\psi,k}^2}{\overline{\alpha}_{m,2}^2\left(\alpha\left\|I_{c_\psi}\right\|_F + \Lambda_{\psi,k}^2\right)},\end{aligned}$$

Therefore, given (5.12), (5.13), and (5.26), for all k, $\psi(k)\left(m^2(k)\right)^{-1}$, $\psi^\top(k)\psi(k)\left(m^2(k)\right)^{-1}$, and, subsequently, $K_{LS}(k)$ and $\Xi_{\epsilon,2}(k)$ are all bounded from above. Continuing, from (2.10), with θ being constant, and, because, as stipulated above, $P(k) \to P_{ss}$ as time evolves, (5.44) allows us to say that

$$\lim_{k\to\infty} \hat{\theta}(k) = \theta + \lim_{k\to\infty} \tilde{\theta}(k) = \theta + \lim_{k\to\infty} P(k-1)P_0^{-1}\tilde{\theta}(k_0) + \Xi_{\epsilon,1}(k)$$

is finite and constant if $\lim\limits_{k\to\infty} \Xi_{\epsilon,1}(k)$ is finite and constant. Given the expression $\Xi_{\epsilon,1}(k)$ in (5.45) and because $P(k)$, $K_{LS}(k)$, and $\Xi_{\epsilon,2}(k)$ are bounded from above for all k, $\lim\limits_{k\to\infty} \Xi_{\epsilon,1}(k)$ is finite and constant if, for instance, vec $\{w_\epsilon(k)\} \in \mathcal{L}^p$ or, correspondingly, vec $\{w\left(x(k)\right)\} \in \mathcal{L}^p$, $p \in [1, \infty)$, because that would mean $\lim\limits_{k\to\infty} w_\epsilon(k) = [0]^{r_{Q_\epsilon} \times c_{Q_\epsilon}}$ or $\lim\limits_{k\to\infty} w\left(x(k)\right) = [0]^{r_f \times c_f}$ (see properties of $\mathcal{L}^p$, with $p \in [1, \infty)$, signal spaces in Appendix C). Hence, if (5.41) stands, then vec $\{w_\epsilon(k)\} \in \mathcal{L}^2$ or, equivalently, vec $\{w\left(x(k)\right)\} \in \mathcal{L}^2$ and, as $k \to \infty$, $\hat{\theta}(k) \to \hat{\theta}_{ss}$.

Better learning is achieved by having $\hat{\theta}(k)$ converge to θ (or a neighborhood of it) as time evolves. When it comes to the NRLS algorithm, this is only possible if $\psi_m(k) = \psi(k)m^{-1}(k) \in \mathbb{R}^{r_\psi \times c_\psi}$ is PE [30, 2, 3]. The PE condition on $\psi_m(k)$, as mentioned before, is quite demanding however. Because it imposes conditions on past, current, and future regressor states, persistency of excitation is not simple to achieve and/or monitor on-line.

We present a summary of the DTNRLS algorithm, its results and properties in Table 5.1.

Next, we present our main contribution, i.e., the NRLS-based Concurrent Learning algorithm in DT framework. For a less demanding condition than the PE condition and/or the $\mathcal{L}^2$ property of vec $\{w\left(x(k)\right)\}$, we show that CL leads to good learning.

Table 5.1: DTNRLS algorithm, results, and properties.

DTNRLS algorithm	
For $k \geq k_0$, $\phi\left(x(k)\right) \in \mathbb{R}^{r_\phi \times c_\phi}$ being the regressor, $q_\epsilon(k)$ given by (2.14), some positive definite $P(k_0 - 1) = P_0 \in \mathbb{R}^{r_\phi \times c_\psi}$, and some $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$, update $\hat{\theta}$ by applying: $\psi(k) = \phi\left(x(k)\right) \in \mathbb{R}^{r_\psi \times c_\psi}$, $m^2(k) = \alpha I_{c_\psi} + \psi^\top(k) P(k-1) \psi(k)$, $K_{LS}(k) = P(k-1)\psi(k)\left(m^2(k)\right)^{-1}$, $Q_\epsilon(k) = q_\epsilon(k)$, $\Delta\hat{\theta}_{LS}(k) = K_{LS}(k) Q_\epsilon(k)$, $\hat{\theta}(k+1) = \hat{\theta}(k) - \Delta\hat{\theta}_{LS}(k)$, and, for next iteration, $P(k) = P(k-1) - K_{LS}(k)\psi^\top(k)P(k-1)$.	
SUAC results and properties	**UUAC results and properties**
$\tilde{\theta}(k)$, $\left\Vert m^{-1}(k) q_\epsilon(k) \right\Vert_F$, and $\left\Vert \Delta\tilde{\theta}(k) \right\Vert_F$ are bounded; $\left\Vert m^{-1}(k) q_\epsilon(k) \right\Vert_F$, and $\left\Vert \Delta\tilde{\theta}(k) \right\Vert_F$ belong to $\mathcal{L}^2$; $P(k) = P^\top(k) > 0$ for all $k \geq k_0$, $P(k)$ remains bounded and converges to a constant $P_{ss} \in \mathbb{R}^{r_\psi \times r_\psi}$ as $k \to \infty$; $\tilde{\theta}(k)$ converges to a constant $\tilde{\theta}_{ss} \in \mathbb{R}^{r_\theta \times c_\theta}$ as $k \to \infty$;	$P(k) = P^\top(k) > 0$ for all $k \geq k_0$, $P(k)$ remains bounded and converges to a constant $P_{ss} \in \mathbb{R}^{r_\psi \times r_\psi}$ as $k \to \infty$; Same as SUAC (if not already mentioned) only if $\operatorname{vec}\left\{w\left(x(k)\right)\right\} \in \mathcal{L}^2$.
Convergence Properties	
$\tilde{\theta}(k)$ converges to the origin (meaning $\hat{\theta}(k)$ converges to θ) in the SUAC or $\tilde{\theta}(k)$ converges to a neighborhood of the origin (meaning $\hat{\theta}(k)$ converges to a neighborhood of θ) in the UUAC if $\psi_m(k)$ in (5.25) is persistently exciting.	

5.2 Discrete-Time Normalized Recursive Least Squares Based Concurrent Learning for Both Structured and Unstructured Uncertainty Approximation Cases

Similarly to what is done in [3], it can be shown that the NRLS algorithm of (5.1) could be obtained via minimization with respect to $\hat{\theta}(k)$ of the scalar cost function

$$J_{LS}(k) = \frac{1}{2}\mathrm{tr}\left\{ \sum_{\tau=k_0}^{k-1} \frac{1}{\alpha}\Upsilon_{LS}(\tau) + \Upsilon_0(k) \right\},$$

where $\alpha = \alpha(k)$ is the same positive scalar that figures in the expression of $m^2(k)$, i.e., (5.3), we define $\Upsilon_{LS} \in \mathbb{R}^{c_{q_\epsilon} \times c_{q_\epsilon}}$ and $\Upsilon_0 \in \mathbb{R}^{c_\theta \times c_\theta}$ as

$$\Upsilon_{LS}(k) = Q_\epsilon^\top(k) Q_\epsilon(k),$$

with $Q_\epsilon(k)$ defined in (5.4), and

$$\Upsilon_0(k) = \left(\hat{\theta}(k) - \hat{\theta}_0\right)^\top P_0^{-1} \left(\hat{\theta}(k) - \hat{\theta}_0\right). \tag{5.47}$$

Notice that $\Upsilon_{LS}(k)$ is an instantaneous cost, alike the one minimized when applying the Gradient Descent algorithm. The Least Squares algorithm, instead, minimizes a summation of instantaneous costs.

As aforementioned (see Section 9.4.1), given (2.14) and (5.4), if $\mathcal{M}_n = \mathcal{M}_1$, i.e., model (2.2a), $c_{Q_\epsilon} = c_{q_\epsilon} = r_f = c_\theta$, while, when $\mathcal{M}_n = \mathcal{M}_2$, i.e., model (2.2b), $c_{Q_\epsilon} = c_{q_\epsilon} = c_f = c_\theta$. There are therefore no dimensional ambiguities in the definition of $J_{LS}(k)$ above. Moreover, proving that (5.1) is derived through minimization of $J_{LS}(k)$ will be indirectly shown through the derivation of our CL law.

5.2.1 Discrete-Time Normalized Recursive Least Squares Based Concurrent Learning Algorithm

Recalling that $r_{q_\epsilon} = c_\phi$ and $c_{q_\epsilon} = c_\theta$ no matter which model, i.e., (2.2a) or (2.2b), is used for approximation, let $\xi_\phi \in \mathbb{R}^{r_{q_\epsilon} \times r_{q_\epsilon}}$ and $\xi_Z \in \mathbb{R}^{c_Z \times c_Z}$ be positive definite matrices, i.e., $\xi_\phi > 0$ and

$\xi_Z > 0$. Also, let $\bar{\Upsilon}_{LS}$, $\Upsilon_{CL} \in \mathbb{R}^{c_\theta \times c_\theta}$ be defined as

$$\bar{\Upsilon}_{LS}(k) = q_\epsilon^\top(k)\, \xi_\phi \xi_\phi^\top\, q_\epsilon(k), \tag{5.48}$$

with instantaneous approximation error $q_\epsilon(k)$ defined in (2.14), and

$$\Upsilon_{CL}(k) = q_{Z,\epsilon}^\top(k)\, \xi_Z \xi_Z^\top\, q_{Z,\epsilon}(k), \tag{5.49}$$

where the approximation error based on recorded data $q_{Z,\epsilon}(k)$ is defined in (3.27). Consider the scalar cost function

$$J_{LS,CL}(k) = \frac{1}{2}\text{tr}\left\{ \sum_{\tau=k_0}^{k-1} \frac{1}{\alpha} \left[\bar{\Upsilon}_{LS}(\tau) + \Upsilon_{CL}(\tau)\right] + \Upsilon_0(k) \right\}.$$

Notice that $J_{LS,CL}$ is a modification of J_{LS} (previously defined). To minimize $J_{LS,CL}(k)$ with respect to $\hat{\theta}(k)$, we first express the gradient $\dfrac{\partial J_{LS,CL}}{\partial \hat{\theta}(k)}$ and equate it to the origin, i.e., $\dfrac{\partial J_{LS,CL}}{\partial \hat{\theta}(k)} = [0]^{r_\theta \times c_\theta}$.

For the following calculation of the gradient, we use the expression of $J_{LS,CL}(k)$ above in conjunction with the expressions of $q_\epsilon(k)$ in (5.14a), $\bar{\Upsilon}_{LS}(k)$ in (5.48), $\Upsilon_0(k)$ in (5.47), $q_{Z,\epsilon}(k)$ in (3.28a), $\Upsilon_{CL}(k)$ in (5.49), as well as relationships (C.3) and (C.4). As it turns out,

$$\begin{aligned} \frac{\partial J_{LS,CL}(k)}{\partial \hat{\theta}(k)} &= \sum_{\tau=k_0}^{k-1} \frac{1}{\alpha} \left[\phi\left(x(\tau)\right)\, \xi_\phi \xi_\phi^\top\, q_\epsilon(\tau) + Z\, \xi_Z \xi_Z^\top\, q_{\epsilon,Z}(\tau)\right] + P_0^{-1}\left(\hat{\theta}(k) - \hat{\theta}_0\right) \\ &= P^{-1}(k-1)\hat{\theta}(k) - \nu(k-1), \end{aligned}$$

where, with $r_Z = r_\phi = r_\theta$ and, $c_{q_\epsilon} = c_\theta$ whether model (2.2a) or (2.2b) is used, because of (5.14a) and (3.28a), and denoting $\bar{\Phi}_k \in \mathbb{R}^{r_\phi \times r_\phi}$, $\bar{\Phi}_{Z,k} \in \mathbb{R}^{r_Z \times r_Z}$, $\bar{\Phi}_{f_n,k} \in \mathbb{R}^{r_\phi \times c_\theta}$, and $\bar{\Phi}_{f_n,k} \in \mathbb{R}^{r_\phi \times c_\theta}$ as

$$\bar{\Phi}_k = \phi\left(x(k)\right)\, \xi_\phi \xi_\phi^\top\, \phi^\top(x(k)) = \left[\phi\left(x(k)\right)\xi_\phi\right]\left[\phi\left(x(k)\right)\xi_\phi\right]^\top, \tag{5.50}$$

$$\bar{\Phi}_{Z,k} = Z\, \xi_Z \xi_Z^\top\, Z^\top = \left[Z\, \xi_Z\right]\left[Z\, \xi_Z\right]^\top, \tag{5.51}$$

$$\bar{\Phi}_{f_n,k} = \phi\left(x(k)\right)\, \xi_\phi \xi_\phi^\top\, f_n(k) = \left[\phi\left(x(k)\right)\xi_\phi\right]\left[f_n^\top(k)\, \xi_\phi\right]^\top, \tag{5.52}$$

and

$$\bar{\Phi}_{f_n,k} = \phi\left(x(k)\right)\, \xi_Z \xi_Z^\top f_n = \left[\phi\left(x(k)\right)\, \xi_Z\right] \left[f_n^\top \xi_Z\right]^\top, \tag{5.53}$$

for $k > k_0$,

$$P^{-1}(k-1) = P_0^{-1} + \sum_{\tau=k_0}^{k-1} \frac{1}{\alpha}\left[\bar{\Phi}_\tau + \bar{\Phi}_{Z,\tau}\right] \tag{5.54}$$

and $\nu \in \mathbb{R}^{r_\theta \times c_\theta}$ is such that

$$\nu(k-1) = P_0^{-1}\hat{\theta}_0 + \sum_{\tau=k_0}^{k-1} \frac{1}{\alpha}\left[\bar{\Phi}_{f_n,\tau} + \bar{\Phi}_{f_n,\tau}\right]. \tag{5.55}$$

Hence,

$$\frac{\partial J_{LS,CL}(k)}{\partial \hat{\theta}(k)} = P^{-1}(k-1)\hat{\theta}(k) - \nu(k-1) = [0]^{r_\theta \times c_\theta}$$

means

$$\hat{\theta}(k) = P(k-1)\nu(k-1). \tag{5.56}$$

Recall that $P(k_0-1) = P_0 > 0$ means $P_0^{-1} > 0$. Therefore, because, for all k, both $\bar{\Phi}_k \geq 0$ and $\bar{\Phi}_{Z,k} \geq 0$, judging by (5.50) and (5.51) respectively, it is not wrong to assume (as (5.54) hints to) that the inverse to $P(k)$ exists for all k since, based on (5.54), $P^{-1}(k) > 0$.

Unlike in Section 9.4.1, let $\psi(k) \in \mathbb{R}^{r_\psi \times c_\psi} = \mathbb{R}^{r_Z \times (c_\phi + c_Z)}$, i.e., $r_\psi = r_Z = r_\phi = r_\theta$ and $c_\psi = c_\phi + c_Z$, be given by

$$\psi(k) = \left[\ \phi\left(x(k)\right)\xi_\phi, \quad Z\,\xi_Z\ \right], \tag{5.57}$$

with $\phi\left(x(k)\right)$ and Z defined in (2.5) and (3.1) respectively. Given the definitions of $\bar{\Phi}_k$ in (5.50), $\bar{\Phi}_{Z,k}$ in (5.51), and based on (5.57),

$$\Psi_k = \psi(k)\psi^\top(k) = \bar{\Phi}_k + \bar{\Phi}_{Z,k}. \tag{5.58}$$

We thus have from (5.54) that

$$P^{-1}(k) = P_0^{-1} + \sum_{\tau=k_0}^{k-1} \frac{1}{\alpha}\Psi_\tau + \frac{1}{\alpha}\Psi_k = P^{-1}(k-1) + \frac{1}{\alpha}\Psi_k,$$

which is exactly what we got before in (5.8). Therefore, working backwards and using the Woodbury Matrix Identity (A.1) to invert the expression of $P^{-1}(k)$ above (or (5.8)), we get $P(k)$ given by (5.2), with both $K_{LS}(k)$ and $m^2(k)$ defined respectively in (5.1b) and (5.3). This is the derivation of $P(k)$ we alluded to back in Section 9.4.1. Further, recalling that $f_n(k) \in \mathbb{R}^{r_{q_\epsilon} \times c_{q_\epsilon}}$ (see (2.16) for definition), with, as discussed before, $r_{q_\epsilon} = c_\phi$ and $c_{q_\epsilon} = c_\theta$ regardless of the model used, and $f_n \in \mathbb{R}^{c_Z \times c_\theta}$ (see (3.29) for definition), we denote $f(k) \in \mathbb{R}^{c_\theta \times (c_\phi + c_Z)}$ as

$$f(k) = \left[\; f_n^\top(k)\,\xi_\phi, \quad f_n^\top \xi_Z \;\right]. \tag{5.59}$$

Hence, notice, from (5.52), (5.53), (5.57), and (5.59), that

$$\psi(k) f^\top(k) = \bar{\Phi}_{f_n,k} + \bar{\Phi}_{f_n,k}.$$

As a result, from (5.55), we can write

$$\begin{aligned} \nu(k) &= P_0^{-1}\hat{\theta}_0 + \sum_{\tau=k_0}^{k-1} \frac{1}{\alpha}\psi(\tau) f^\top(\tau) + \frac{1}{\alpha}\psi(k) f^\top(k) \\ &= \nu(k-1) + \frac{1}{\alpha}\psi(k) f^\top(k). \end{aligned} \tag{5.60}$$

Going back to (5.56) and using (5.60), we have that

$$\hat{\theta}(k+1) = P(k)\nu(k) = P(k)\left[\nu(k-1) + \frac{1}{\alpha}\psi(k) f^\top(k)\right].$$

Because, for all k, $P^{-1}(k)$ exists, then (5.56) means

$$\nu(k-1) = P^{-1}(k-1)\hat{\theta}(k).$$

We also get from (5.8) that

$$P^{-1}(k-1) = P^{-1}(k) - \frac{1}{\alpha}\Psi_k.$$

Subsequently, $\hat{\theta}(k+1)$ above could also be expressed as

$$\begin{aligned}\hat{\theta}(k+1) &= P(k)\left[P^{-1}(k-1)\hat{\theta}(k) + \frac{1}{\alpha}\psi(k)f^{\top}(k)\right] \\ &= P(k)\left[\left(P^{-1}(k) - \frac{1}{\alpha}\Psi_k\right)\hat{\theta}(k) + \frac{1}{\alpha}\psi(k)f^{\top}(k)\right]\end{aligned}$$

or, given the definition of Ψ_k in (2.27),

$$\hat{\theta}(k+1) = \hat{\theta}(k) - \frac{1}{\alpha}P(k)\psi(k)Q_{\epsilon}(k), \tag{5.61}$$

where, here, the overall approximation error $Q_{\epsilon}(k) \in \mathbb{R}^{r_{Q_\epsilon}\times c_{Q_\epsilon}} = \mathbb{R}^{(c_\phi + c_Z)\times c_\theta}$, i.e., now $r_{Q_\epsilon} = c_\phi + c_Z$ and $c_{Q_\epsilon} = c_\theta$, is

$$Q_{\epsilon}(k) = \psi^{\top}(k)\hat{\theta}(k) - f^{\top}(k), \tag{5.62}$$

with $\psi(k)$ and $f(k)$ respectively defined in (5.57) and (5.59). Hence, given the definitions of $q_\epsilon(k)$ and $q_{Z,\epsilon}(k)$ in (5.14a) and (3.28a) respectively, (5.62) can be rewritten as

$$Q_{\epsilon}(k) = \xi_\phi^{\top} q_\epsilon(k) + \xi_Z^{\top} q_{Z,\epsilon}(k). \tag{5.63}$$

Because (5.3) allows us to write

$$\psi^{\top}(k)P(k-1)\psi(k) = m^2(k) - \alpha I_{c_\psi},$$

we also have, using (5.1b) and (5.2), that

$$\begin{aligned}P(k)\psi(k) &= P(k-1)\psi(k) - K_{LS}(k)\left(m^2(k) - \alpha I_{c_\phi}\right) \\ &= P(k-1)\psi(k)\left[I_{c_\phi} - \left(m^2(k)\right)^{-1}\left(m^2(k) - \alpha I_{c_\phi}\right)\right] \\ &= \alpha P(k-1)\psi(k)\left(m^2(k)\right)^{-1}.\end{aligned}$$

As a result, (5.61) becomes

$$\hat{\theta}(k+1) = \hat{\theta}(k) - P(k-1)\psi(k)\left(m^2(k)\right)^{-1}Q_{\epsilon}(k).$$

We summarize our NRLS-based CL algorithm as follows. Starting with an initial parameter estimate vector $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$, given a positive definite, symmetric matrix $P(k_0 - 1) = P_0 \in \mathbb{R}^{r_\psi \times c_\psi}$, and given positive definite matrices $\xi_\phi \in \mathbb{R}^{r_{q_\epsilon} \times r_{q_\epsilon}}$, $r_{q_\epsilon} = c_\phi$, and $\xi_Z \in \mathbb{R}^{c_Z \times c_Z}$, the discrete-time NRLS-based CL algorithm for updating the parameter estimate $\hat{\theta}$ for $k \geq k_0$ is

$$\Delta\hat{\theta}_{LS,CL}(k) = K_{LS}(k)Q_\epsilon(k), \tag{5.64a}$$

$$\hat{\theta}(k+1) = \hat{\theta}(k) - \Delta\hat{\theta}_{LS,CL}(k), \tag{5.64b}$$

where $\psi(k)$, which is to be used in the computation of $K_{LS}(k)$ in (5.1b), $P(k)$ in (5.2), and $m^2(k)$ in (5.3), is given by (5.57) and the error $Q_\epsilon(k)$ is given by (5.62).

Matrices ξ_ϕ and ξ_Z are essentially weighting matrices, as portrayed by (5.63). Hence, the designer could choose to emphasize the contribution of either the instantaneous approximation error $q_\epsilon(k)$, especially, on suspicion that the history stack Z may contain outdated or corrupted information, or the error $q_{Z,\epsilon}(k)$, which is based on recorded data, for possibly better learning generalization. For numerical consideration however, because $r_{q_\epsilon} = c_\theta$ or c_Z could be quite large, in practice, both ξ_ϕ and ξ_Z could be reduced to just scalars. This can essentially be done by setting $\xi_\phi = \varepsilon_\phi I_{r_{q_\epsilon}}$ and $\xi_Z = \varepsilon_Z I_{c_Z}$, for some scalars $\varepsilon_\phi > 0$ and $\varepsilon_Z > 0$. It should be noted that the previous derivations do remain the same if this is opted for.

Notice that, form wise, the initial DTNRLS algorithm (5.1) and the DTNRLS based CL algorithm (5.64) look identical. The only differences between the two are the manner in which the regressor $\psi(k)$ ((5.1a) versus (5.57)) and the overall estimation error $Q_\epsilon(k)$ ((5.4) versus (5.62)) are computed. Consequently, when it comes to the DTNRLS based CL algorithm (5.64), proving the boundedness of $\tilde{\theta}(k)$ and other properties that follow, be it in the SUAC or the UUAC, has already been established in Section 9.4.1 and, thus, need not be repeated again.

Nevertheless, as we will show in the next section, when using the DTNRLS based CL algorithm and faced with the UUAC, boundedness of $\tilde{\theta}(k)$ can be proven without necessitating thatvec $\{w\,(x(k))\} \in \mathcal{L}^2$ as it was needed in Section 5.1.2. Moreover, we will investigate how the DTNRLS based CL algorithm can guarantee good parameter identification even in the absence of the PE condition.

5.2.2 Convergence of the Gain Matrix

Let the rank condition, i.e., **Condition 3.1.1**, be verified. If so, the history stack Z is full row rank and, as a result, $\Phi_{Z,k}$ (defined in (3.14)) is positive definite. Given that both $\xi_\psi > 0$ and $\xi_Z > 0$, then $\xi_\psi \xi_\psi^\top > 0$ and $\xi_Z \xi_Z^\top > 0$ also. Letting

$$\underline{\lambda}_\psi = \lambda_{\min}\Big(\xi_\psi \xi_\psi^\top\Big) > 0,\, \overline{\lambda}_\psi = \lambda_{\max}\Big(\xi_\psi \xi_\psi^\top\Big) > 0,$$

$$\underline{\lambda}_Z = \lambda_{\min}\Big(\xi_Z \xi_Z^\top\Big) > 0, \text{ and } \overline{\lambda}_Z = \lambda_{\max}\Big(\xi_Z \xi_Z^\top\Big) > 0,$$

granted $\Phi_k \geq 0$ for all k and $r_\psi = r_\phi = r_\theta$, we can write, from (2.27), (3.14), (5.50), (5.51), and (5.58), that

$$\underline{\lambda}_Z \underline{\lambda}_{\Phi_{Z,k}} I_{r_\psi} \leq \underline{\lambda}_\psi \Phi_k + \underline{\lambda}_Z \Phi_{Z,k} \leq \Psi_k \leq \overline{\lambda}_\psi \Phi_k + \overline{\lambda}_Z \Phi_{Z,k}. \tag{5.65}$$

With $\Phi_{Z,k} > 0$, $\underline{\lambda}_{\Phi_{Z,k}} > 0$ and (5.65) imply $\Psi_k > 0$.

Now, with $\alpha > 0$, $P_0 > 0$, and $\Psi_k > 0$, (5.9) and (5.10) allow us to write

$$P^{-1}(k) = P_0^{-1} + \sum_{\tau=k_0}^{k} \frac{1}{\alpha}\Psi_\tau > \sum_{\tau=k_0}^{k} \frac{1}{\alpha}\Psi_\tau > 0.$$

Moreover, we can find a small scalar $\underline{\beta}_\Psi > 0$ such that, given (5.6),

$$\underline{\beta}_\Psi I_{r_\psi} \leq \underline{\lambda}_{\Psi_k}(k) I_{r_\psi} \leq \Psi_k \leq \overline{\lambda}_{\Psi_k}(k) I_{r_\psi} \tag{5.66}$$

and

$$P^{-1}(k) > \lambda_{\min}\left(P^{-1}(k)\right) I_{r_\psi} \geq \sum_{\tau=k_0}^{k} \underline{\beta}_\Psi I_{r_\psi} = \left(\sum_{\tau=k_0}^{k} \underline{\beta}_\Psi\right) I_{r_\psi}.$$

Hence,

$$\begin{aligned}\lim_{k\to\infty}\lambda_{\min}\left(P^{-1}(k)\right) &\geq \lim_{k\to\infty}\sum_{\tau=k_0}^{k}\underline{\beta}_{\Psi} = \lim_{k\to\infty}(k-k_0+1)\underline{\beta}_{\Psi} \\ &= \infty.\end{aligned} \tag{5.67}$$

With

$$\lambda_{\min}\left(P^{-1}(k)\right) = \left(\lambda_{\max}\left(P(k)\right)\right)^{-1} \to \infty,$$

then $\lambda_{\max}\left(P(k)\right) \to 0$, and, consequently, regardless of the approximation case,

$$\lim_{k\to\infty} P(k) = P_{ss} = [0]^{r_\psi \times r_\psi}. \tag{5.68}$$

Now, recall the dynamics of $\tilde{\theta}(k)$ given by (5.44) in the UUAC and, in particular, for the SUAC, with $w_\epsilon(k) = [0]^{r_{Q_\epsilon}\times c_{Q_\epsilon}}$ thus $\Xi_{\epsilon,1} \equiv [0]^{r_\theta\times c_\theta}$ (see (5.45) for the reason why), (5.44) is reduced to (5.33).

5.2.3 Convergence of the Parameter Error in the SUAC

As $P(k) \to [0]^{r_\psi\times r_\psi}$, we get, from (5.33), that

$$\lim_{k\to\infty}\tilde{\theta}(k) = \lim_{k\to\infty}P(k-1)P_0^{-1}\tilde{\theta}(k_0) = [0]^{r_\theta\times c_\theta}.$$

Thus, we conclude, in this case, that $\tilde{\theta}(k) = [0]^{r_\theta\times c_\theta}$ is asymptotically stable. Given (5.33) and, using (5.12), $P(k-1)P_0^{-1} \leq I_{r_\psi}$, alike in the case of standard DTNRLS algorithm, we have that

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{LS,CL_s} = \left\|I_{r_\psi}\right\|_F\left\|\tilde{\theta}(k_0)\right\|_F = \sqrt{r_\psi}\left\|\tilde{\theta}(k_0)\right\|_F \tag{5.69}$$

for all k.

As discussed above, because of the similarity of the standard DTNRLS algorithm and its CL modification, boundedness of $\tilde{\theta}(k)$ can be attained with the same analysis as in Section 5.1.1. The asymptotic convergence of $\tilde{\theta}(k)$ to the origin as well as its boundedness can then be used to deduce all approximation related properties of Section 5.1.1, namely the boundedness of $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F$,

$\|\psi_m(k)\|_F = \left\|\psi(k)m^{-1}(k)\right\|_F$, and $\left\|\Delta\tilde{\theta}(k)\right\|_F$, whose upper bounds are respectively expressed in (5.24), (5.26), and (5.29). In particular, given that $\tilde{\theta}(k) \to [0]^{r_\theta \times c_\theta}$, using (5.23), for some scalars $B_{q,1} > 0$ and $B_{q,2} \geq 0$,

$$\sum_{\tau=k_0}^{\infty} \left(\frac{\|Q_\epsilon(\tau)\|_F}{\|m(\tau)\|_F}\right)^2 \leq \sum_{\tau=k_0}^{\infty} B_{q,1} \left\|\tilde{\theta}(k)\right\|_F^2 \leq B_{q,2} < \infty.$$

That is to say $\dfrac{\|Q_\epsilon(k)\|_F}{\|m(k)\|_F} \in \mathcal{L}^2$ and, as a result, considering (5.24) and (5.29) respectively, both $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F \in \mathcal{L}^2$ and $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$.

5.2.4 Convergence of the Parameter Error in the UUAC

We have shown earlier that $\psi(k)\left(m^2(k)\right)^{-1}$, $K_{LS}(k)$, and $\Xi_{\epsilon,2}(k)$ are all bounded from above. If (2.7) is verified, then W is an upper bound to $\|w\left(x(k)\right)\|_F$ and, equivalently, $\|w_\epsilon(k)\|_F$ for all k. With $P(k) \to [0]^{r_\psi \times r_\psi}$ as $k \to \infty$, given (5.1b), we also have that $K_{LS}(k) \to [0]^{r_\psi \times c_\psi}$ as time evolves. Given the previously cited attributes and properties, not only are both

$$\left\|P(k-1)\sum_{\tau=k_0}^{k-2} \Xi_{\epsilon,2}(\tau)w_\epsilon(\tau)\right\|_F \quad \text{and} \quad \|K_{LS}(k-1)w_\epsilon(k-1)\|_F$$

bounded from above and for all time, but also

$$\lim_{k\to\infty} P(k-1)\sum_{\tau=k_0}^{k-2} \Xi_{\epsilon,2}(\tau)w_\epsilon(\tau) = [0]^{r_\theta \times c_\theta}$$

and

$$\lim_{k\to\infty} K_{LS}(k-1)w_\epsilon(k-1) = [0]^{r_\theta \times c_\theta}.$$

Hence, given (5.45), there exists a scalar $B_{w_\epsilon,3} \geq 0$ that is such that $\|\Xi_{\epsilon,1}(k)\|_F \leq B_{w_\epsilon,3}$ and $\lim_{k\to\infty} \Xi_{\epsilon,1}(k) = [0]^{r_\theta \times c_\theta}$. Now, using (5.12), $P(k-1)P_0^{-1} \leq I_{r_\psi}$, and, recalling (5.44),

$$\left\|\tilde{\theta}(k)\right\|_F \leq \Theta_{LS,CL_u} = \left\|I_{r_\psi}\right\|_F \left\|\tilde{\theta}(k_0)\right\|_F + B_{w_\epsilon,3} = \sqrt{r_\psi}\left\|\tilde{\theta}(k_0)\right\|_F + B_{w_\epsilon,3}. \tag{5.70}$$

Moreover, keeping with (5.44),

$$\lim_{k\to\infty} \tilde{\theta}(k) = \lim_{k\to\infty} P(k-1)P_0^{-1}\tilde{\theta}(k_0) + \Xi_{\epsilon,1}(k) = [0]^{r_\theta \times c_\theta}.$$

That is, $\tilde{\theta}(k)$ is bounded for all k and $\tilde{\theta}(k) = [0]^{r_\theta \times c_\theta}$ is asymptotically stable. Notice that we did not need vec $\{w_\epsilon(k)\}$ to belong to an $\mathcal{L}^p$, $p \in [1, \infty)$, space to get this result. Instead, if Z satisfies the rank condition, then, we can achieve boundedness of $\tilde{\theta}(k)$ as well as its ultimate convergence to the origin in the UUAC. Subsequently, what this allows us to do is to deduce the other approximation related properties. Using (5.23), boundedness of $\tilde{\theta}(k)$ implies boundedness of $\frac{\|Q_\epsilon(k)\|_F}{\|m(k)\|_F}$, which can in turn be used to justify boundedness of both $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F \in \mathcal{L}^2$ and $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$ expressed respectively in (5.24) and (5.29).

5.2.5 Concluding Remarks about DTNRLS based CL

The following theorem summarizes our results.

Theorem 5.2.1. *Let f be an uncertainty to approximate. Consider the approximator $\mathcal{F}$ in (2.1), which can be modeled by (2.2a) or (2.2b). Also, consider the on-line scheme (2.8) and the approximation error (2.9a) (equivalently, (2.9b)). If the rank condition (described by Condition 3.1.1) is satisfied, then the discrete-time Normalized Recursive Least Squares based Concurrent Learning adaptation scheme (5.64) guarantees:*

- $P(k) = P^\top(k) > 0$ *and* $P(k)$ *is bounded;*
- $\tilde{\theta}(k)$*, consequently* $\hat{\theta}(k)$ *(since* θ *is considered to be constant),* $m^{-1}(k)Q_\epsilon(k)$ *and* $\Delta\tilde{\theta}(k) = \tilde{\theta}(k) - \tilde{\theta}(k-1)$ *are bounded;*
- $\lim_{k\to\infty} P(k) = [0]^{r_\psi \times r_\psi}$*;*
- $\tilde{\theta}(k) \to [0]^{r_\theta \times c_\theta}$ *asymptotically or, equivalently,* $\hat{\theta}(k) \to \theta$ *asymptotically as* $k \to \infty$*;*

- *if in the SUAC, both* $\left\|m^{-1}(k)Q_\epsilon(k)\right\|_F \in \mathcal{L}^2$ *and* $\left\|\Delta\tilde{\theta}(k)\right\|_F \in \mathcal{L}^2$.

We also make the remarks that follow.

Remark 5.2.1. An argument can be made that the DTNRLS algorithm of (5.1) already uses memory because, given (5.2), as time goes by, $\psi(k)$ at different k values is "absorbed" into $P(k)$. By exclusively picking the memory items that form the history stack Z, the DTNRLS based CL algorithm of (5.64) can possibly help deliver better approximation results.

Remark 5.2.2. Recall that having $\lambda_{\min}\left(P^{-1}(k)\right) \to \infty$ (see (5.67)) is what then leads to all the convergence results. It would thus be ideal, for instance, that $\underline{\beta}_\Psi$ or, subsequently, given (5.66), $\underline{\lambda}_{\Psi_k}$ is very large for faster convergence. From (5.65), notice that the larger $\underline{\lambda}_{\Phi_{Z,k}}$ is, the larger $\underline{\lambda}_{\Psi_k}$ would also be. Hence, when running the CL algorithm, it would be a good idea to compute, monitor, and maximize either the minimum eigenvalue

$$r_{CL,1} = \underline{\lambda}_{\Phi_{Z,k}} \tag{5.71}$$

of $\Phi_{Z,k} = ZZ^\top$ or, done differently, the inverse condition number

$$r_{CL,2} = \frac{\underline{\lambda}_{\Phi_{Z,k}}}{\overline{\lambda}_{\Phi_{Z,k}}} \tag{5.72}$$

of $\Phi_{Z,k} = ZZ^\top$ also, by selective selection of the data that goes into the history stack Z. As highlighted later in the data recording procedures, we opt for maximizing

$$r_{CL} = r_{CL,2} = \frac{\underline{\lambda}_{\Phi_{Z,k}}}{\overline{\lambda}_{\Phi_{Z,k}}} \tag{5.73}$$

so as to have a well-conditioned $\Phi_{Z,k}$.

Remark 5.2.3. Let $k_R \in \mathbb{N}_+$, with $k_R \geq k_0$, be the DT at which **Condition 3.1.1** is met. Also, let $k_Z \in \mathbb{N}_+$ represent any DT that is such that $k_Z \geq k_R$. As previously established, for any $k = k_Z \geq k_R$, $\Psi_k = \Psi_{k_Z} > 0$. Using (5.9) and (5.10),

$$P^{-1}(k_Z) = P_0^{-1} + \sum_{\tau=k_0}^{k_Z} \frac{1}{\alpha}\Psi_\tau.$$

Multiplying each side of the equality in the equation above by $P(k_Z)$ and rearranging, we get

$$P_J(k_Z) = I_{r_\psi} - P(k_Z)P_0^{-1} = P(k_Z)\sum_{\tau=k_0}^{k_Z}\frac{1}{\alpha}\Psi_\tau > 0$$

given that, for all k, $P(k) > 0$ (see (5.12)), $\alpha > 0$, and, for $k = k_Z \geq k_R$, $\Psi_{k_Z} > 0$. Now, considering (5.44) in general,

$$\tilde{\theta}(k_Z+1) = P(k_Z)P_0^{-1}\tilde{\theta}(k_0) + \Xi_{\epsilon,1}(k_Z+1),$$

which, due to (2.10), is also

$$\hat{\theta}(k_Z+1) - \theta = P(k_Z)P_0^{-1}\left[\hat{\theta}(k_0) - \theta\right] + \Xi_{\epsilon,1}(k_Z+1).$$

Recalling that $r_\psi = r_\phi = r_\theta$ and having established that $P_J(k_Z) > 0$ (implying $P_J^{-1}(k_Z)$ exists), the unknown, true parameter θ is identified as

$$\begin{aligned}\theta = {} & P_J^{-1}(k_Z)\left[\hat{\theta}(k_Z+1) - P(k_Z)P_0^{-1}\hat{\theta}(k_0)\right] \\ & - P_J^{-1}(k_Z)\Xi_{\epsilon,1}(k_Z+1).\end{aligned} \tag{5.74}$$

When it comes to the SUAC, because $w_\epsilon(k) = [0]^{r_{Q_\epsilon}\times c_{Q_\epsilon}}$ implies $\Xi_{\epsilon,1} = [0]^{r_\theta \times c_\theta}$ given (5.45), then (5.74) becomes

$$\theta = P_J^{-1}(k_Z)\left[\hat{\theta}(k_Z+1) - P(k_Z)P_0^{-1}\hat{\theta}(k_0)\right]. \tag{5.75}$$

We have shown that $\hat{\theta}(k) \to \theta$ ultimately (which **Theorem 9.4.2** also stipulates). Notice that, for the UUAC, the identified θ in (5.74) is subject to the representation error $w_\epsilon(k)$ (or $w\left(x(k)\right)$) present in the expression of $\Xi_{\epsilon,1}$ given by (5.45). It is therefore even more apparent that, for good learning, a good approximation structure that commands a small $\|w_\epsilon(k)\|_F$ has to be realized.

Remark 5.2.4. Similar expressions to (5.75) for the SUAC and (5.74) for the UUAC can be obtained when employing the DTNRLS algorithm of (5.1) instead and assuming that $\psi(k)$ in (5.1a) is exciting over a period of time. Based on **Definition B.1** in Appendix B and the preceding analysis, considering some DT $\tau \geq k_0$, scalar $\delta_\Psi \in \mathbb{N}_+$, and the time sequence $\Delta_\Psi = \{\tau, \tau+1, \ldots, \tau+\delta_\Psi\}$, the key to being able to do so would be to have

$$\sum_{k=\tau}^{\tau+\delta_\Psi} \Psi_k = \sum_{k=\tau}^{\tau+\delta_\Psi} \psi(k)\psi^\top(k) > 0. \tag{5.76}$$

It should be noted that (5.76) is only possible if the duration of Δ_Ψ is at least the number of rows (or columns) in Ψ_k. That is, $(\tau + \delta_\Psi) - (\tau) + 1 \geq r_\psi$ or $\delta_\Psi \geq r_\psi - 1$. Moreover, verifying (5.76) is quite similar to having $\Psi_k > 0$ if and when the rank condition on Z is met. Nonetheless, in the case of the DTNRLS based CL algorithm, verification of the rank condition can happen over the course of time, as opposed to necessarily having it happen over a long enough Δ_Ψ. Additionally, the designer not only has the freedom of selectively picking what goes into Z, but, doing so constructively can lead to faster convergence. For those reasons, necessitating that (5.76) be verified in the case of the DTNRLS algorithm could turn more restrictive than having the rank condition on Z being satisfied in the case of the DTNRLS based CL algorithm.

Table 5.2 is a summary of the DTNRLS based CL algorithm, its results and properties.

Next, provided are data recording procedures, similar to the CT framework data recording procedure proposed in [15, 16, 14, 17, 18].

Table 5.2: DTNRLS based CL algorithm, results, and properties.

DTNRLS based CL algorithm	
For $k \geq k_0$, $\phi\left(x(k)\right) \in \mathbb{R}^{r_\phi \times c_\phi}$ being the regressor, the history stack Z given by (3.1), $q_\epsilon(k)$ given by (2.14), $q_{Z,\epsilon}(k)$ given by (3.27), some positive definite $\xi_\phi \in \mathbb{R}^{r_\phi \times c_\phi}$, $\xi_Z \in \mathbb{R}^{r_Z \times c_Z}$, and $P(k_0-1) = P_0 \in \mathbb{R}^{r_\phi \times c_\psi}$, and some $\hat{\theta}(k_0) = \hat{\theta}_0 \in \mathbb{R}^{r_\theta \times c_\theta}$, update $\hat{\theta}$ by applying: $\psi(k) = \left[\phi\left(x(k)\right)\xi_\phi,\, Z\,\xi_Z\right] \in \mathbb{R}^{r_\psi \times c_\psi}$, $m^2(k) = \alpha I_{c_\psi} + \psi^\top(k)P(k-1)\psi(k)$, $K_{LS}(k) = P(k-1)\psi(k)\left(m^2(k)\right)^{-1}$, $Q_\epsilon(k) = \xi_\phi^\top q_\epsilon(k) + \xi_Z^\top q_{Z,\epsilon}(k)$, $\Delta\hat{\theta}_{LS}(k) = K_{LS}(k)Q_\epsilon(k))$, $\hat{\theta}(k+1) = \hat{\theta}(k) - \Delta\hat{\theta}_{LS}(k)$, and, for next iteration, $P(k) = P(k-1) - K_{LS}(k)\psi^\top(k)P(k-1)$.	
SUAC results and properties	**UUAC results and properties**
Same as DTNRLS algorithm in SUAC (see Table 5.1); If **Condition 3.1.1** is verified: $\tilde{\theta}(k) = [0]^{r_\theta \times c_\theta}$ is asymptotically stable; $P(k) \to [0]^{r_\psi \times c_\psi}$ as $k \to \infty$.	Same as DTNRLS algorithm in UUAC (see Table 5.1); If **Condition 3.1.1** is verified: $\tilde{\theta}(k) = [0]^{r_\theta \times c_\theta}$ is asymptotically stable; $P(k) \to [0]^{r_\psi \times c_\psi}$ as $k \to \infty$; $\left\lVert m^{-1}(k)q_\epsilon(k)\right\rVert_F$, and $\left\lVert \Delta\tilde{\theta}(k)\right\rVert_F$ belong to $\mathcal{L}^\infty$.
Convergence Properties	
$\tilde{\theta}(k)$ converges to the origin (meaning $\hat{\theta}(k)$ converges to θ) in the SUAC or $\tilde{\theta}(k)$ converges to a neighborhood of the origin (meaning $\hat{\theta}(k)$ converges to a neighborhood of θ) as long as **Condition 3.1.1** is verified. **Condition 3.1.1** is less demanding than the PE condition on $\psi_m\left(x(k)\right)$ in (5.25) and/or requiring that vec $\{w\} \in \mathcal{L}^2$).	

CHAPTER VI

DATA RECORDING PROCEDURE FOR CONCURRENT LEARNING IN DISCRETE-TIME FRAMEWORK

Verification of the rank condition is essential to realizing the CL algorithm properties. Given some recorded off-line data, it is possible, depending on the richness of the data, to begin parameter adaptation with a full row rank history stack Z (or Z_G). However, for on-line implementation (which is of importance in the present research), Z starts empty and, if the regressor is sufficiently rich, then the rank condition can be realized. The designer can nevertheless decide if CL adaptation is to be carried from the start or only after achieving the rank condition (i.e., **Condition 3.1.1**).

This chapter presents data recording procedures for CL implementation in DT settings. The recording procedures that follow are described in terms of both Z (defined in (3.1)) and Z_G (defined in (3.8)). Recall, as we have pointed out earlier, that Z_G may be much more suitable to use for numerical operations rather than Z (and $\bar{Z}_G$ in (3.9) for that matter) given that its contents, i.e., the columns of Z_G, are bounded. Also, as pointed to earlier, knowing Z and G, with the latter defined in (3.10), Z_G (and $\bar{Z}_G$) can be constructed, though, as we have seen in Chapter V, in the case of the DTNRLS based CL algorithm, keeping track of G, Z_G, and $\bar{Z}_G$ is not necessary. Lastly (but importantly), recall the definition of the regressor signal $\phi\left(x(k)\right) \in \mathbb{R}^{r_\phi \times c_\phi}$ in (2.5). That is

$$\phi\left(x(k)\right) = \left[\begin{array}{cccc} \zeta_1(x(k)), & \zeta_2(x(k)), & \ldots, & \zeta_{c_\phi}(x(k)) \end{array}\right],$$

where, for $i = 1, 2, \ldots, c_\phi$, $\zeta_i(x(k)) \in \mathbb{R}^{r_\phi}$.

6.1 Data Recording Procedure 1

Data Recording Procedure 1 (DRP1) can be implemented if CL is to be applied from the very beginning, i.e., $k = k_0$. Recall that **Condition 3.1.1**, which we need to verify, implies $c_Z \geq r_Z = r_\phi = r_\theta$. Let $n_Z = n_Z(k)$ be the number of filled slots (columns) in Z at time k. Without prior knowledge, we start out with an empty c_Z-slot (column wise) memory bank $Z \in \mathbb{R}^{r_Z \times c_Z}$, i.e., $n_Z(k_0 - 1) = 0$. As long as $n_Z < r_Z$, at any DT $k \geq k_0$,

$$n_Z^c = \min\left(c_Z - n_Z(k-1), c_\phi\right) \tag{6.1}$$

vectors $\zeta_i(x(k))$, $i = 1, 2, \ldots, c_\phi$, of $\phi\left(x(k)\right)$ that increase the rank of Z are selected and stored in Z. Essentially, for $k \geq k_0$,

$$n_Z(k) = n_Z(k-1) + n_Z^c.$$

A co-strategy (besides and less important than that of meeting the rank condition) for picking the ζ_i vectors can be devised. For instance, ζ_i's can be picked such that the ratio r_{CL} is maximized (given that this is imposed later). Regardless, once Z is full, for faster convergence of the learning algorithms, old data in Z is strategically replaced so as to increase the metric r_{CL}. Procedure 1 outlines DRP1.

6.2 Data Recording Procedure 2

With Data Recording Procedure 2 (DRP2), CL is only applied after sufficient collection of data is able to guarantee the rank condition. Let $H \in \mathbb{R}^{r_Z \times c_H}$, $c_H \in \mathbb{N}_+$, be a matrix in which data is stored during the transition period. That is, the time period before verification of the rank condition. If/when the rank condition is verified, we would have that $c_H \geq r_Z$. In the case when $c_H > r_Z$, we

Data Recording Procedure 1 DRP1 for maximizing r_{CL}

```
Require: Initialize $n_Z = 0$
  Consider one $\zeta_i(x(k))$, $i = 1, 2, \ldots, c_\phi$, of $\phi(x(k))$ (see (2.5)) at a time and do the following for all $\zeta_i$'s
  if $n_Z = 0$ then
    Update counter: $n_Z \leftarrow n_Z + 1$;
    Store $\zeta_i(x(k))$ in slot $n_Z$ of $Z$, corresponding $\bar{g}_{i,i}(k)$ in $G$, and corresponding row or column of $f(x(k))$ in $F_n$ (see (3.12) and (3.13)). Construct $Z_G$ and $\bar{Z}_G$ using $Z$ and $G$;
  else
    if $n_Z < r_Z$ then
      if rank $\left(\left[Z_G, \frac{\zeta_i(x(k))}{\sqrt{\bar{g}_{i,i}(k)}}\right]\right) >$ rank$(Z_G)$ then
        Update counter: $n_Z \leftarrow n_Z + 1$;
        Store $\zeta_i(x(k))$ in slot $n_Z$ of $Z$, corresponding $\bar{g}_{i,i}(k)$ in $G$, and corresponding row or column of $f(x(k))$ in $F_n$ (see (3.12) and (3.13)). Construct $Z_G$ and $\bar{Z}_G$ using $Z$ and $G$;
      end if
    else if $n_Z \geq r_Z$ and $n_Z < c_Z$ then
      Update counter: $n_Z \leftarrow n_Z + 1$;
      Store $\zeta_i(x(k))$ in slot $n_Z$ of $Z$, corresponding $\bar{g}_{i,i}(k)$ in $G$, and corresponding row or column of $f(x(k))$ in $F_n$ (see (3.12) and (3.13)). Construct $Z_G$ and $\bar{Z}_G$ using $Z$ and $G$;
    else
      Find metric $r_{old} = r_{CL}$ of current $Z_G$ (for DTNG based CL) or of current $Z$ (for DTNRLS based CL);
      Create an empty row vector $r_{new}$;
      for $j = 1$ to $c_Z$ do
        Set $\hat{Z}_G = Z_G$ (for DTNG based CL) or $\hat{Z} = Z$ (for DTNRLS based CL);
        Replace data in column $j$ of $\hat{Z}$ by $\zeta_i(x(k))$ and/or column $j$ of $\hat{Z}_G$ by $\frac{\zeta_i(x(k))}{\sqrt{\bar{g}_{i,i}(k)}}$;
        Find metric $r_{CL}$ of $\hat{Z}_G$ (for DTNG based CL) or of $\hat{Z}$ (for DTNRLS based CL) and save in column $j$ of $r_{new}$;
      end for
      Let $r_{max} = \max_{j=1,\ldots,c_Z} r_{new}$ found at column $j_{max}$ of $r_{new}$;
      if $r_{max} > r_{old}$ then
        Store $\zeta_i(x(k))$ in slot $j_{max}$ of $Z$, corresponding $\bar{g}_{i,i}(k)$ in $G$, and corresponding row or column of $f(x(k))$ in $F_n$ (see (3.12) and (3.13)). Construct $Z_G$ and $\bar{Z}_G$ using $Z$ and $G$;
      end if
    end if
  end if
```

extract the r_Z linearly independent columns of H. Let $T_Z(\cdot) \in \mathbb{R}^{r_Z \times c_Z}$ denote such an extraction transformation. We compute $Z = T_Z(H)$. However, in the case when the rank condition has been found to be verified and $c_H = r_Z$, $Z = T_Z(H) = H$. Next, we can proceed to apply DPR1 starting with a full $Z = T_Z(H) \in \mathbb{R}^{r_Z \times r_Z}$ matrix and, therefore, with $n_Z = r_Z$.

It is important to add, while data is being recorded in H, Z remains empty and therefore standard algorithm can be implemented instead. The DTNG based CL algorithm of (4.58) can for instance be implemented for $\xi_{CL} = [0]^{r_\phi \times r_\phi}$ and $\Delta\hat{\theta}_{CL} = [0]^{r_\phi \times c_{q_\epsilon}}$, with η picked according to (4.73). Essentially, this amounts to implementing a slightly modified version the standard DTNG algorithm. Similarly, the standard DTNRLS algorithm can be implemented in place of the DTNRLS based CL algorithm.

DRP2 is given by Procedure 2, where n_H denotes the number of filled slots in H. Moreover, associated with H are the row vector $G_H \in \mathbb{R}^{1 \times c_H}$ of $\bar{g}_{i,i}(\tau_j)$ values, $i \in \{1, 2, \ldots, c_\phi\}$ and $j \in \{1, 2, \ldots, c_Z\}$, and the matrix $F_{H,n} \in \mathbb{R}^{r_F \times c_F}$ containing entities in the uncertainties $f(x(\tau_j))$, in a similar fashion to G in (3.10) and F_n in (3.12) or (3.13).

Data Recording Procedure 2 DRP2 for Maximizing r_{CL}

```
Require: Initialize $n_H = 0$
  Consider one $\zeta_i(x(k))$, $i = 1, 2, \ldots, c_\phi$, of $\phi(x(k))$ (see (2.5)) at a time and do the following for all $\zeta_i$'s;
  if $Z$ is empty then
      Update counter: $n_H \leftarrow n_H + 1$;
      Store $\zeta_i(x(k))$ in slot $n_H$ of $H$, corresponding $\bar{g}_{i,i}(k)$ in $G_H$ (constructed similarly to $G$ in (3.10)), and corresponding row or column of $f(x(k))$ in $F_{H,n}$ (constructed similarly to $F_n$ in (3.12) and (3.13));
      if rank$(H) = r_Z$ then
          $Z = T_Z(H)$;
          Similarly, properly extract $G$ from $G_H$ and $F_n$ from $F_{H,n}$;
          Construct $Z_G$ and $\bar{Z}_G$ using $Z$ and $G$;
          Update counter: $n_Z = r_Z$;
      end if
  else
      Implement DRP1, i.e., Procedure 1;
  end if
```

6.3 Comparing DRP1 to DRP2

DRP1 is initiated at the start of parameter adaptation, i.e., at $k = k_0$, and, thus, subsequent rank check could be biased towards initially recorded data. Extracting and using the richest stored data in H before turning on the CL adaptation, DRP2 therefore prevents the bias issue that DRP1 might introduce. However, unlike DRP2, applying CL modification with DRP1 could be less susceptible to carry outdated and/or corrupted data.

CHAPTER VII

ILLUSTRATIONS

In this chapter, we apply and illustrate the previously investigated and developed algorithms (of Chapters IV and V), i.e., the DTNG and DTNRLS algorithms, as well as their CL modifications, to parameter estimation and uncertainty approximation problems.

For all simulations, we set $\alpha = 1$ and run the algorithms for $k \in [k_0 = 0, k_s = 500]$. Let $x_L, x_H \in \mathbb{R}$ be some scalars that are such that $x_H > x_L$. For $k \geq k_0$, measurements $x(k)$ span the set $D_x = [x_L, x_H] \subset \mathbb{R}$ uniformly, i.e., $x(k_0) = x_L$ and

$$x(k+1) = x(k) + \frac{x_H - x_L}{k_s - k_0}. \tag{7.1}$$

Regardless of the algorithm used, parameter estimates are initialized at the origin, i.e., $\hat{\theta}(k_0) = \hat{\theta}_0 = [0]^{r_\theta \times c_\theta}$.

To run the DTNG algorithm, we set $\Gamma = \eta I_{r_\theta}$, with $\eta = \frac{\overline{\eta}_{NG_s}}{2}$ when in SUAC to validate (4.33) and $\eta = \frac{\overline{\eta}_{NG}}{2}$ in UUAC to validate (4.24). For its CL modification, i.e., the DTNG based CL algorithm, we use $\Gamma = \eta I_{r_\theta}$, with $\eta = \eta_m = \frac{\overline{\eta}_{CL}}{2}$ so as to validate (4.60), $\xi_{NG} = \varepsilon_{NG} I_{r_\theta}$, and $\xi_{CL} = \varepsilon_{CL} I_{r_\theta}$, for some scalars $\varepsilon_{NG} > 0$ and $\varepsilon_{CL} > 0$, which would decide the weights of the contribution of the purely instantaneous term $\Delta\hat{\theta}_{NG}$ or that of the term based on recorded data $\Delta\hat{\theta}_{CL}$. When applying DRP2 and the rank condition is yet to be verified, as pointed out earlier, the CL adaption law is applied for $\xi_{CL} = [0]^{r_\theta \times r_\theta}$ and $\Delta\hat{\theta}_{CL}(k) = [0]^{r_\phi \times r_{q_\epsilon}}$, while setting $\eta = \frac{\overline{\overline{\eta}}_{NG}}{2}$

in order to validate (4.73), thus reducing it to a marginally modified version of the standard DTNG algorithm.

For simulation of all LS based algorithms, we initialize the gain matrix at $P(k_0 - 1) = P_0 = 100\, I_{r_\psi}$, unless otherwise explicitly mentioned. To run the DTNRLS based CL algorithm, we set $\xi_\phi = \varepsilon_\phi I_{r_{q_\epsilon}}$, $\xi_Z = \varepsilon_Z I_{c_Z}$, and pick scalars $\varepsilon_\phi > 0$ and $\varepsilon_Z > 0$. Recall that ξ_ϕ and ξ_Z (and, hence, ε_ϕ and ε_Z) are weights that can be used to accentuate the contribution of either the instantaneous approximation error $q_\epsilon(k)$ or the error $q_{Z,\epsilon}(k)$, based on recorded data, in the expression of the overall approximation error $Q_\epsilon(k)$, i.e., (5.63). It should be added that, when using DRP2, the standard DTRNLS algorithm is implemented till verification of the rank condition before mixing in the CL modification.

As a gauge of how good estimates $\hat{\theta}(k)$ become over time, we will, for the most part, compute metric

$$e(k) = \int_{D_x} \left\| \mathcal{F}\left(x, \hat{\theta}(k)\right) - f\left(x\right) \right\|_F d^{r_x} x \tag{7.2}$$

and plot it versus time. Speaking of plots, on the figures shown in this chapter, legends **‘NG’**, **‘NG-CL’**, **‘LS’**, and **‘LS-CL’** will be used to denote the DTNG, DTNG based CL, DTNRLS and the DTNRLS based CL algorithms respectively.

7.1 SUAC Simulations: Applying CL in DT

We start by approximating the structured uncertainty

$$\begin{aligned} y_s\left(x(k)\right) = f\left(x(k)\right) &= -0.25 + 10\left(e^{-\frac{(x(k)-\frac{\pi}{2})^2}{4}}\right) \\ &= \theta^\top \phi\left(x(k)\right), \end{aligned} \tag{7.3}$$

with

$$\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = \begin{bmatrix} -0.25 \\ 10 \end{bmatrix} \in \mathbb{R}^2$$

representing the true, unknown parameter vector and

$$\phi\left(x(k)\right) = \left[\begin{array}{cc} 1, & \exp\left(-\frac{\left(x(k)-\frac{\pi}{2}\right)^2}{4}\right) \end{array}\right]^\top \in \mathbb{R}^2$$

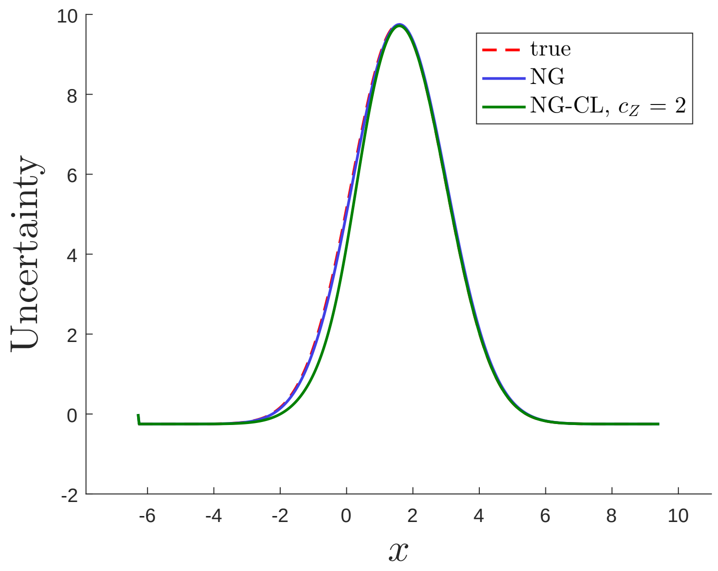
denoting the computable regressor vector. In this example, notice that (7.3) uses model (2.2a), $r_f = c_f = 1$, $r_\psi = r_Z = r_\phi = r_\theta = 2$, $c_\phi = c_\theta = 1$. Setting $c_Z = r_\theta = 2$, $\varepsilon_\phi = \varepsilon_Z = 1$, $x_L = -2\pi$, and $x_H = 3\pi$, DRP1 is used as the data recording procedure when running CL. We actually get very similar results when using DRP2 as the data recording procedure for CL approximation instead of DRP1. For the present numerical simulation example, i.e., (7.3), $r_\phi = r_\theta = 2$ and, therefore, waiting for the rank condition to be verified before applying CL might only require two samples of the x measurement. In Section 7.2, we will demonstrate how using DRP1 or DRP2 can influence the estimation results.

7.1.1 Applying Gradient Descent Based Algorithms

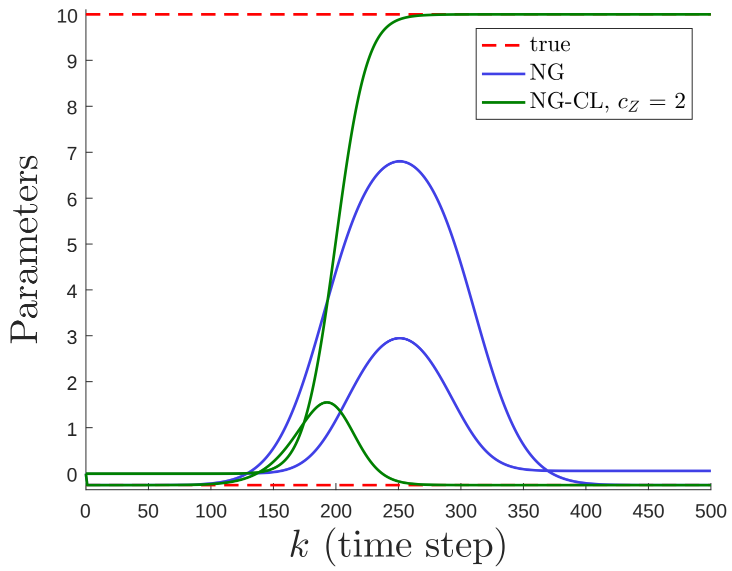
Experiment 1

Figure 7.1a shows the true uncertainty, its DTNG and DTNG based CL on-line approximations (plotted as functions of x) while Figure 7.1b shows the true parameters, their DTNG and DTNG based CL estimates. As anticipated the DTNG based CL parameter estimates converges to their ideal values unlike the DTNG parameter estimates. The on-line DTNG based CL estimation of y_s lags behind for just a while as y_s starts changing rapidly, but eventually catches and matches y_s, whereas the on-line DTNG estimation only matches y_s when in steady state. As the DTNG based CL parameter estimates get to their ideal values, it can be seen on Figure 7.2 that $e(k)$ goes to zero. The DTNG algorithm does relatively well at estimating the uncertainty, as Figure 7.1a shows. However, its parameters estimates are fundamentally adjusted to minimize the instantaneous approximation error $q(k)$ as opposed to being guided towards their ideal values. In the end, as Figure

7.2 consequently reveals, no set of the DTNG parameter estimates can be used to reconstruct the uncertainty y_s in this particular example.



(a) On-line uncertainty approximation.



(b) On-line parameter estimation.

Figure 7.1: Standalone on-line approximation results of the structured uncertainty described by (7.3) while using Gradient Descent based algorithms: DRP1 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.

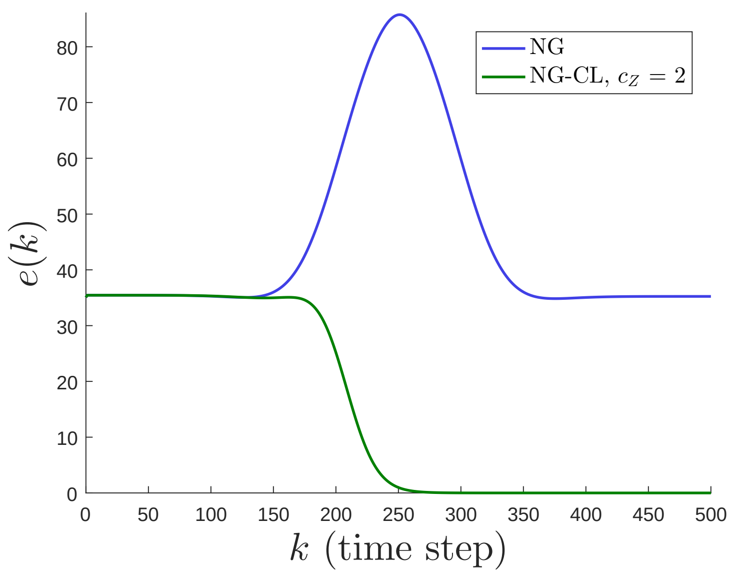


Figure 7.2: Metric $e(k)$ given on-line approximation of the structured uncertainty described by (7.3) while using Gradient Descent based algorithms: DRP1 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.

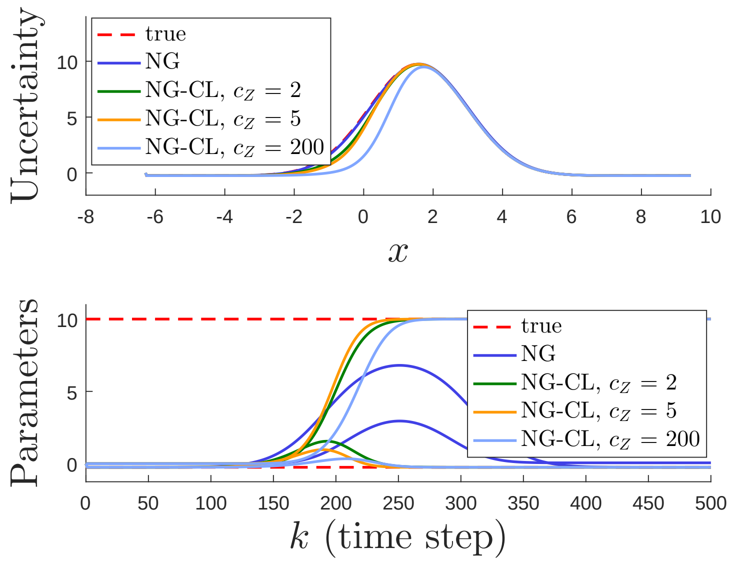
Experiment 2

Next, for $c_Z = 5, 200$, we look into how the DTNG based CL algorithm performs. The uncertainty and parameter estimations are shown on Figure 7.3a while Figure 7.3b shows the metric e and the evolution of the ratio r_{CL} (given by (4.86)) as time goes by. As shown, parameter estimates converge at different rates depending on the ratio r_{CL}. The higher r_{CL} is the faster the convergence. Moreover, as conjectured earlier in Remark 4.2.5, the rate of convergence seems to be higher for values of c_Z closer to $r_Z = r_\phi = r_\theta$.

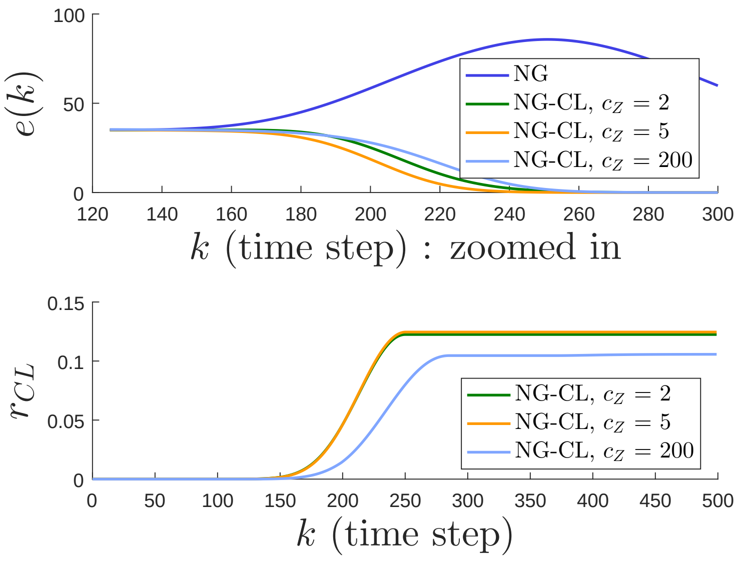
7.1.2 Applying Least Squares Based Algorithms

Experiment 3

The true uncertainty, its DTNRLS and DTNRLS based CL on-line approximations (plotted as functions of x) are shown on Figure 7.4a while Figure 7.4b shows a plot of the true parameters, their DTNRLS and DTNRLS based CL estimates. For this particular example, i.e., the approximation of



(a) On-line approximation results.



(b) Metric $e(k)$ and ratio r_{CL}.

Figure 7.3: Standalone on-line CL approximation results of the structured uncertainty described by (7.3) while using Gradient Descent based algorithms: DRP1 is used for CL, $c_Z = 2, 5, 200$, $\varepsilon_{NG} = \varepsilon_Z = 1$.

y_s given by (7.3), both methods lead to convergence of the parameter estimates to their true values and, as shown on Figure 7.5, $e(k)$ goes to zero in both cases. Convergence seems to be a little faster with the CL modification of the standard DTNRLS algorithm however. Shown on Figure 7.5 is also the evolution of the entries in $P(k)$, which achieve zero (or near to zero) steady states, therefore corroborating (5.32) as far as the DTNRLS method is concerned and (5.68) for its CL modification.

Experiment 4

Keeping the recorded data (obtained from running Experiment 3) in the history stack(s) and running the approximation schemes again, we get the results shown on Figure 7.6. Though the standard DTNRLS algorithm works very well, its CL modification, aided by the recorded history data, yields a faster convergence, therefore encouraging our research motivations.

Experiment 5

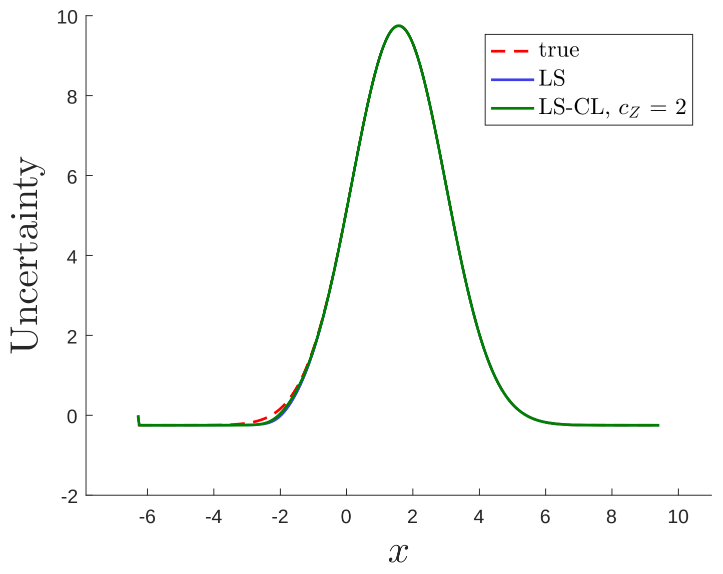
For further analysis, we run the DTNRLS based CL algorithm with $c_Z = 10, 200$ for comparison's sake. Figure 7.7a shows the uncertainty and parameter estimations while the metric e and the evolution of the ratio r_{CL}, which is given by (5.73) in this case, are shown on Figure 7.7b. All instances do well as far as function approximation is concerned (see Figure 7.7a). Parameter estimates converge at different rates (with the CL estimates converging faster than the standard DTNRLS parameter estimates), partly depending on r_{CL} (see Figure 7.7b).

7.2 UUAC Simulations: Applying CL in DT

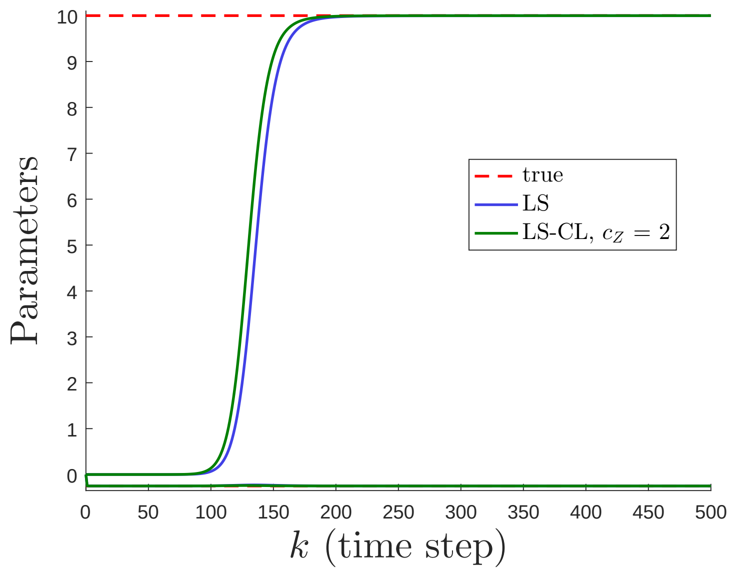
Now, say that we want to approximate the unstructured uncertainty

$$y_u\left(x(k)\right) = f\left(x(k)\right) = \sin\left(x(k)\right). \tag{7.4}$$

Notice from (7.4) that $r_f = c_f = 1$. For approximation purposes, we use model (2.2a), meaning $c_\phi = c_f = 1$, a RBNN with $r_\theta = 15$ basis functions and no bias term. See (2.5) (and (2.6)) for



(a) On-line uncertainty approximation.



(b) On-line parameter estimation.

Figure 7.4: Standalone on-line approximation results of the structured uncertainty described by (7.3) while using Least Squares based algorithms: DRP1 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.

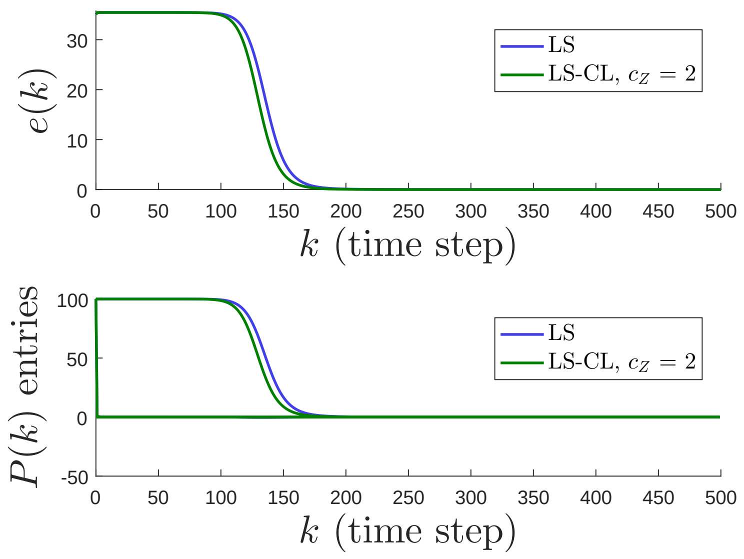
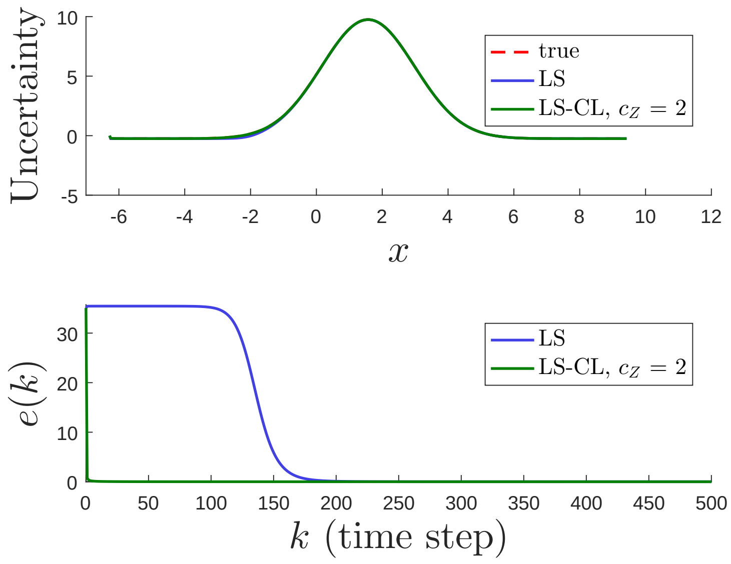


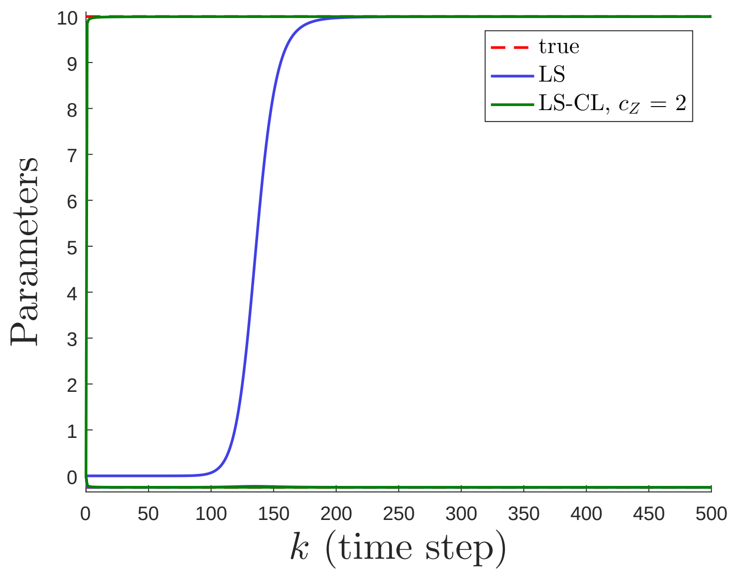
Figure 7.5: Metric $e(k)$ and evolution of the entries in $P(k)$, given on-line approximation of the structured uncertainty described by (7.3) while using Least Squares based algorithms: DRP1 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.

expression of the regressor $\phi\,(x(k))$. For any $i \in \{1, 2, \ldots, c_\phi\}$ and, because no bias term is used, $j \in \{1, 2, \ldots, r_\phi\}$, centroids $\mu_{\zeta_i,j}$'s are uniformly spaced on the interval $[x_L = -2\pi, x_H = 2\pi]$ and spreads $\sigma_{\zeta_i,j}$'s are all set equal to half of the absolute value of the distance between two consecutive centroids.

We first set out to perform off-line training using the RBNN structure. Our motives are two-fold. First, we want to find a good set of "true" parameters. Second, we want to make sure that we have a good RBNN structure for function approximation. We proceed by using both the standard DTNG and the DTNRLS algorithms. It is worth reminding again that, as opposed to on-line approximation, off-line training is run with all $x(k)$ measurements in $D_x = [x_L, x_H]$, obtained through (7.1), for values of $k \in [k_0, k_s]$. After 501 epochs of training with the DTNG algorithm, we plot the off-line approximation performance and the ℓ^2-norm (absolute value would be fine given problem (7.4)) of the representation error $w(x)$ in D_x on Figure 7.9a as well as the evolution of the identified "true"

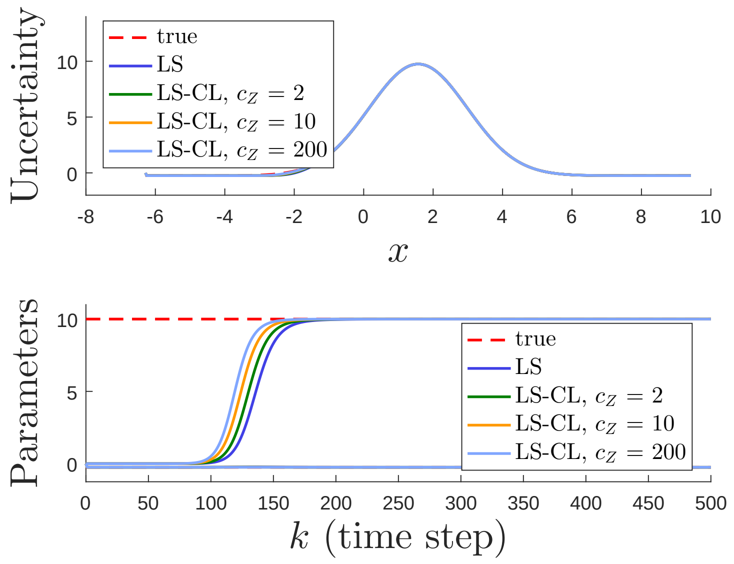


(a) On-line uncertainty approximation and metric $e(k)$.

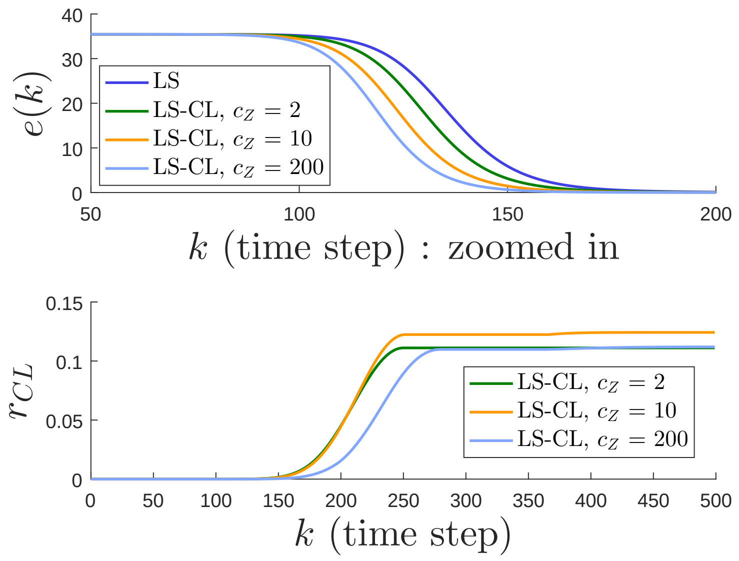


(b) On-line parameter approximation.

Figure 7.6: Standalone on-line approximation results of the structured uncertainty described by (7.3) starting with saved history stack Z obtained from Experiment 3 while using Least Squares based algorithms: DRP1 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.



(a) On-line approximation results.



(b) Metric $e(k)$ and ratio r_{CL}.

Figure 7.7: Standalone on-line CL approximation results of the structured uncertainty described by (7.3) while using Least Squares based algorithms: DRP1 is used for CL, $c_Z = 2, 10, 200$, $\varepsilon_\phi = \varepsilon_Z = 1$.

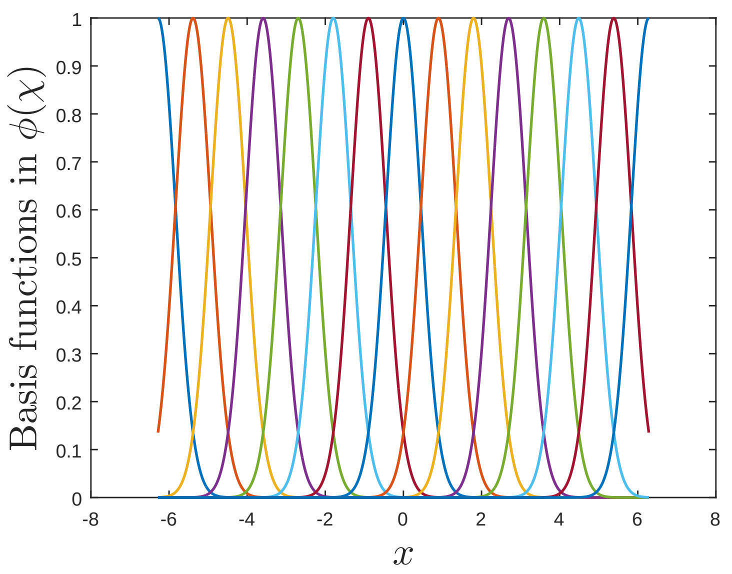
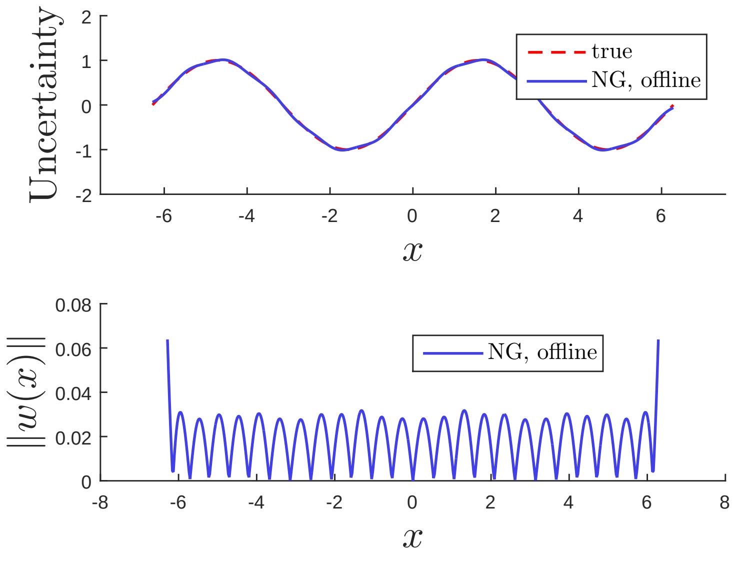


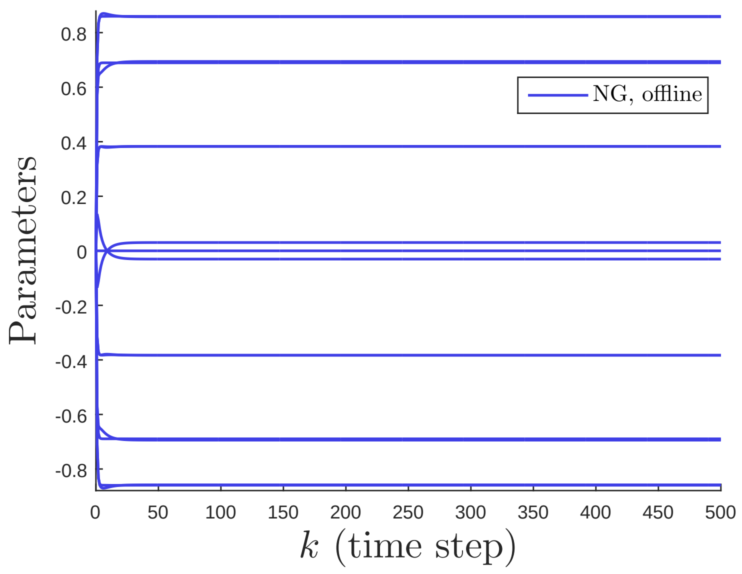
Figure 7.8: Basis functions used for approximating the unstructured uncertainty described by (7.4).

parameters on Figure 7.9b. Similalry, we have Figure 7.10a to show the ℓ^2-norm of $w(x)$ and Figure 7.10b to show the evolution of the "true" parameters when using the DTNRLS algorithm. Notice the subtle difference between Figure 7.9 and Figure 7.10. For instance, Figures 7.9b and 7.9b show that the parameters obtained in the case of the DTNG algorithm seem to converge slightly faster to their final values than the ones obtained in the case of the DTNRLS algorithm. In each case, the end parameter values are used to form the sough after true parameter set θ.

The off-line training done, from here on, we will treat its findings as the *gold standard*. That is, we will compare on-line parameter estimates to the "true" parameters found via off-line training. In the following, we run the the DTNG algorithm, the DTNRLS algorithm, and their respective CL modifications for on-line approximation of y_u.

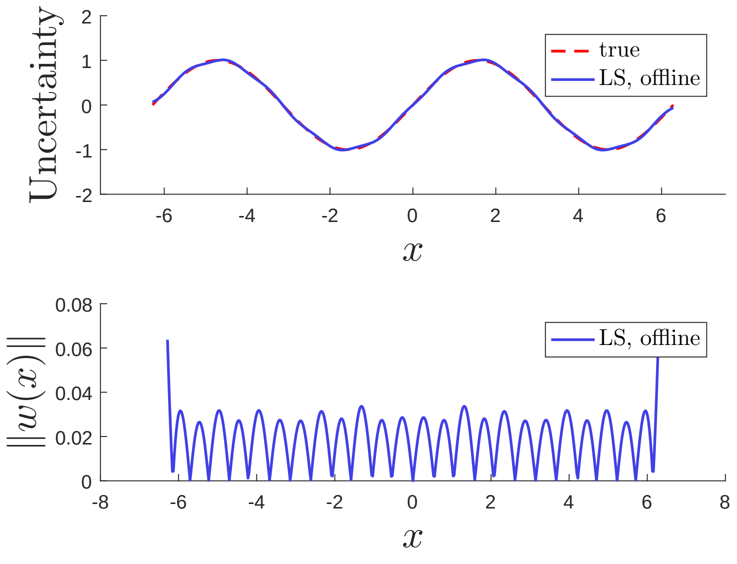


(a) Approximation performance.

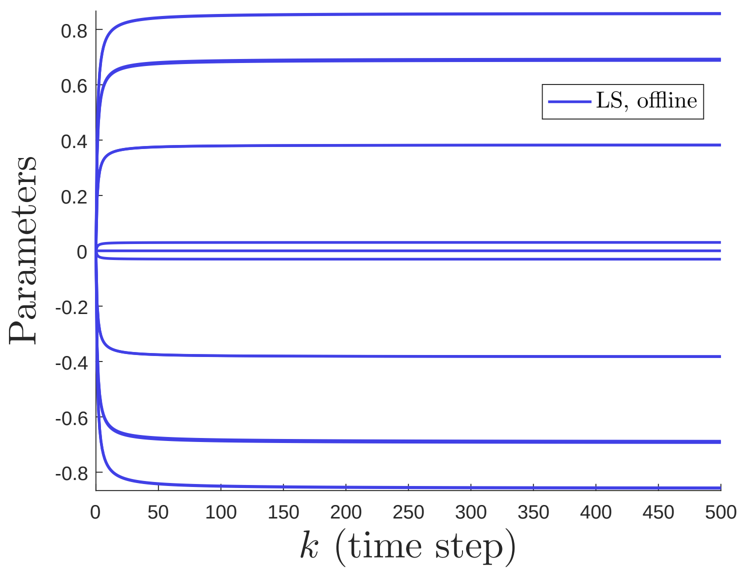


(b) Evolution of true parameters.

Figure 7.9: Off-line training results of the unstructured uncertainty described by (7.4) using the DTNG algorithm.

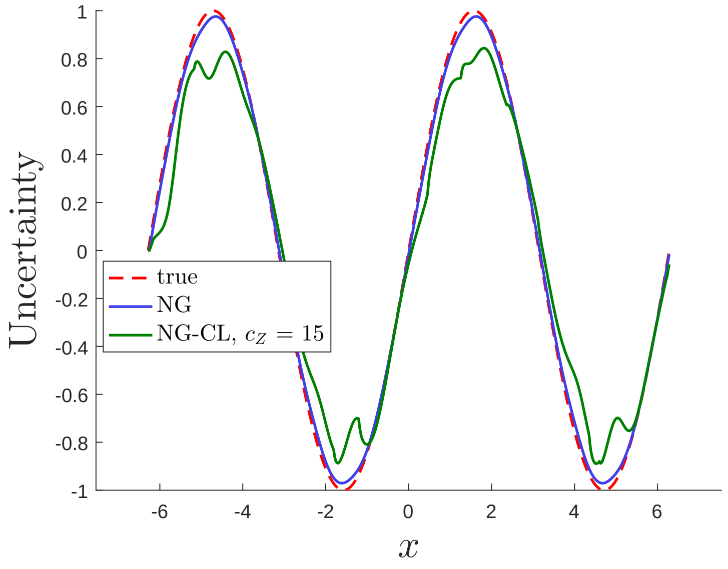


(a) Approximation performance.

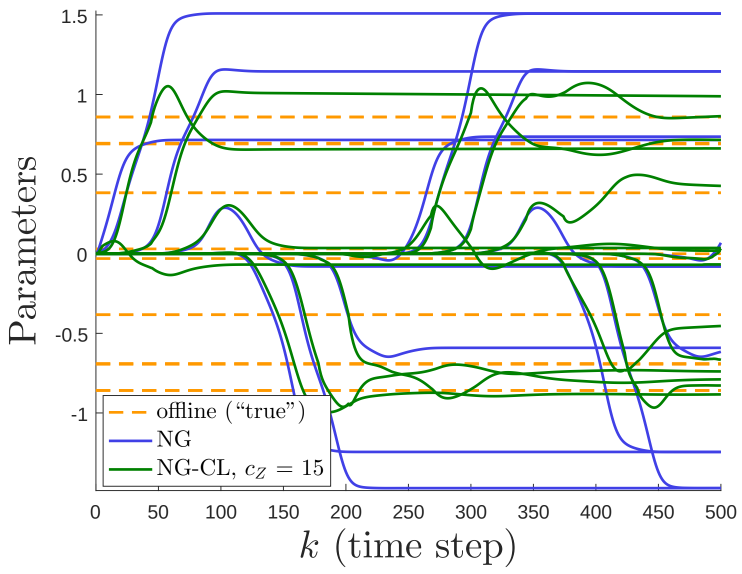


(b) Evolution of true parameters.

Figure 7.10: Off-line training results of the unstructured uncertainty described by (7.4) using the DTNRLS algorithm.

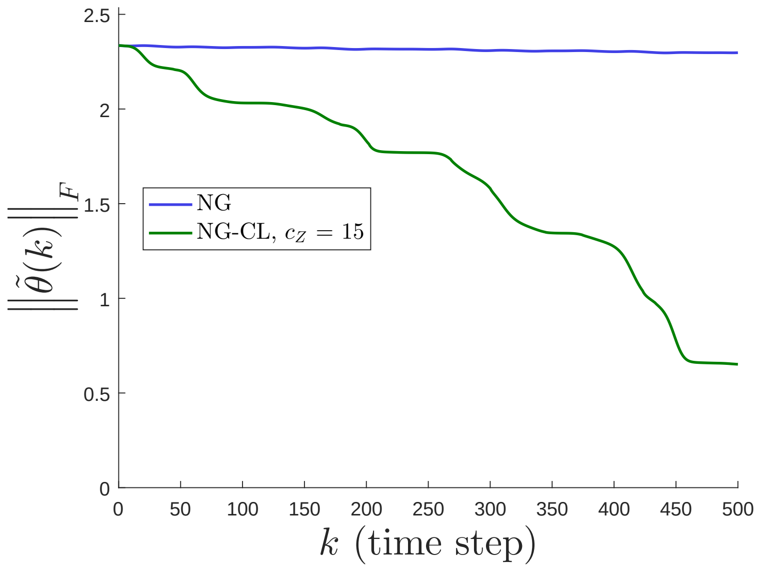


(a) On-line uncertainty approximation.

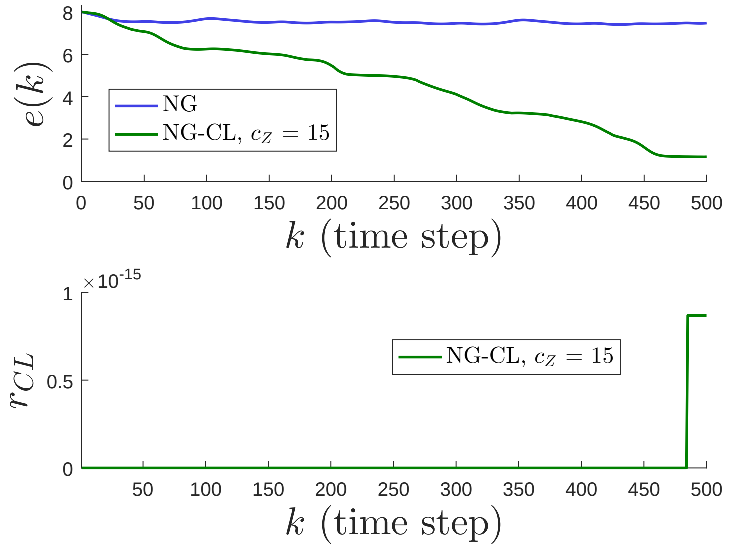


(b) On-line parameter estimation.

Figure 7.11: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Gradient Descent based algorithms: DRP1 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.

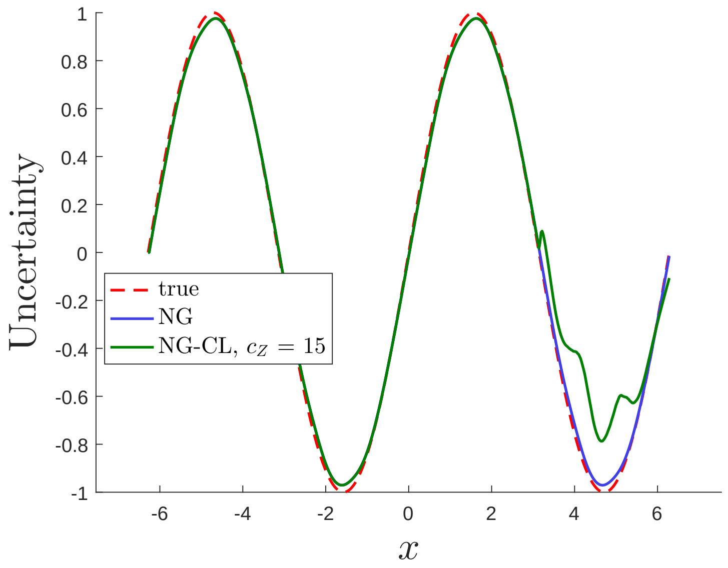


(a) Frobenius norm of parameter error.

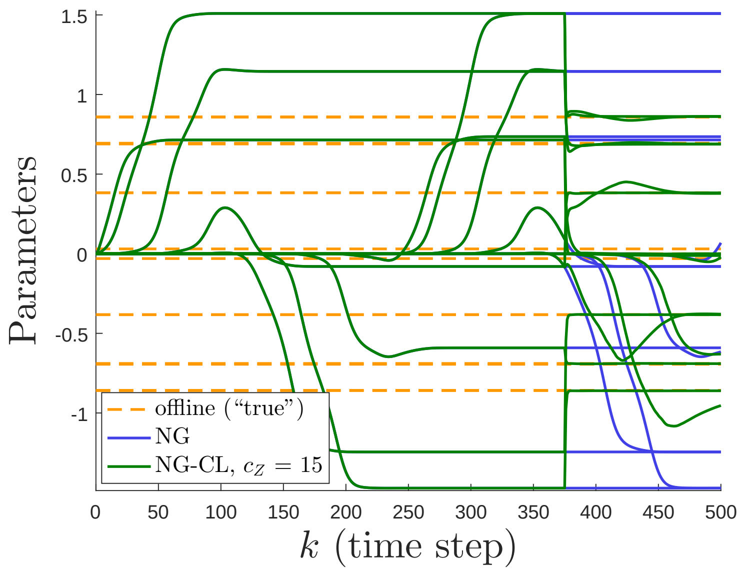


(b) Metric $e(k)$ and ratio r_{CL}.

Figure 7.12: Frobenius norm $\left\|\tilde{\theta}(k)\right\|_F$, metric $e(k)$, and ratio r_{CL} given on-line approximation of the unstructured uncertainty described by (7.4) while using Gradient Descent based algorithms: DRP1 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.

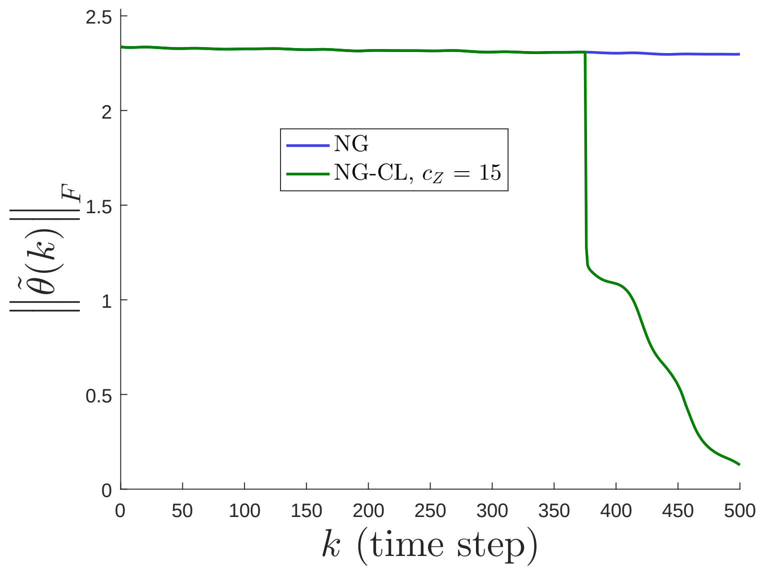


(a) On-line uncertainty approximation.

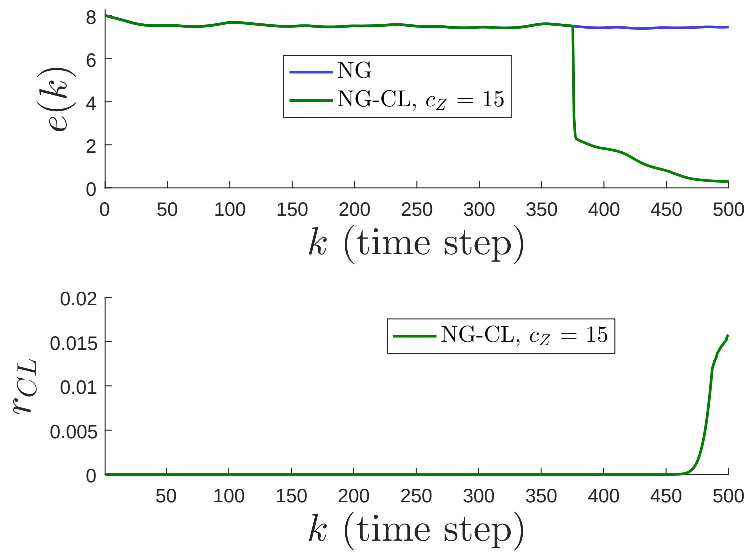


(b) On-line parameter estimation.

Figure 7.13: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Gradient Descent based algorithms: DRP2 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.



(a) Frobenius norm of parameter error.



(b) Metric $e(k)$ and ratio r_{CL}.

Figure 7.14: Frobenius norm $\left\|\tilde{\theta}(k)\right\|_F$, metric $e(k)$, and ratio r_{CL} given on-line approximation of the unstructured uncertainty described by (7.4) while using Gradient Descent based algorithms: DRP2 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.

7.2.1 Applying Gradient Descent Based Algorithms

Experiment 6

First, we set $\varepsilon_{NG} = \varepsilon_{CL} = 1$ for the DTNG based CL implementations, therefore granting the same contributing weight to both the instantaneous update term $\Delta\hat{\theta}_{NG}$ (see (4.5)) and the term based on recorded data $\Delta\hat{\theta}_{CL}$ (see (4.58)). We obtain the results shown on Figures 7.11 and 7.12 with DRP1 whereas Figures 7.13 and 7.14 are obtained when using DRP2.

It is clear in this case (Figure 7.11a versus Figure 7.13a) that DRP1 and DRP2 affect the approximation results differently. It is easy to see the difference here, as opposed to the corresponding SUAC example, because, for the present example, achieving the rank condition is more difficult due to the larger number of (row) dimensions in the RBNN regressor.

While, with DRP1, the rank condition tends to be biased towards or centered around initial recorded data, DRP2 allows for collection of data while waiting for the rank condition to be verified (here, around $k = 375$) to fully deploy CL. This is further demonstrated by the fact that, in the long run, as Figures 7.12b and 7.14b show, DRP2 achieves a much higher ratio r_{CL} comparatively to DRP1. It should be noted that, before the $k = 375$ mark, applying the modified DTNG algorithm in place of the full DTNG based CL law is the reason why the DTNG based CL function approximation on Figure 7.13a and parameter estimation on Figure 7.13b while using DRP2 follow closely that of the its DTNG counterparts. From that mark on, however, substantial changes in the evolution of the parameter estimates (see Figure 7.13b) lead to function approximation degradation (Figure 7.13a).

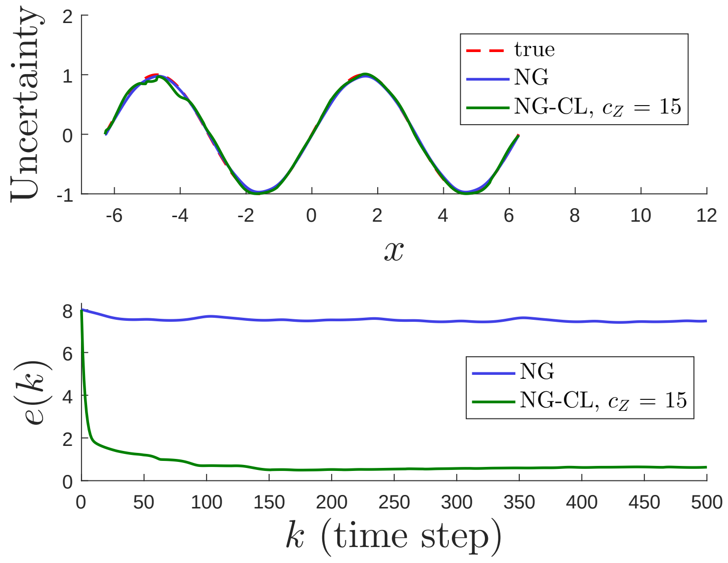
Moreover, regardless of the data recording technique used, as far as function approximation is concerned, the DTNG algorithm does see to do better than the DTNG based CL algorithm, at least, for the present simulation example (see Figures 7.11a and 7.13a). However, when it comes to parameter estimation, as shown on Figures 7.11b, 7.13b, 7.12a, and 7.14a, the DTNG based

CL algorithm fares better than the DTNG algorithm. The plots of the evolution of the metric $e(k)$ on Figures 7.12b and 7.14b reinforce our argument for CL having a better learning generalization property than pure NG. When using CL, as $\left\|\tilde{\theta}(k)\right\|_F$ decreases (Figures 7.12a and 7.14a) with k increasing, $e(k)$ also follows suit (Figures 7.12b and 7.14b). Given that DRP2 leads to a higher ratio r_{CL} than DRP1, as expected, we ultimately get much lower $\left\|\tilde{\theta}\right\|_F$ and e with DRP2. Moreover, it should be added that, by the end of the simulations, $r_{CL} \neq 0$ in both cases means, based on Procedures 1 and 2, that the rank condition has been satisfied.

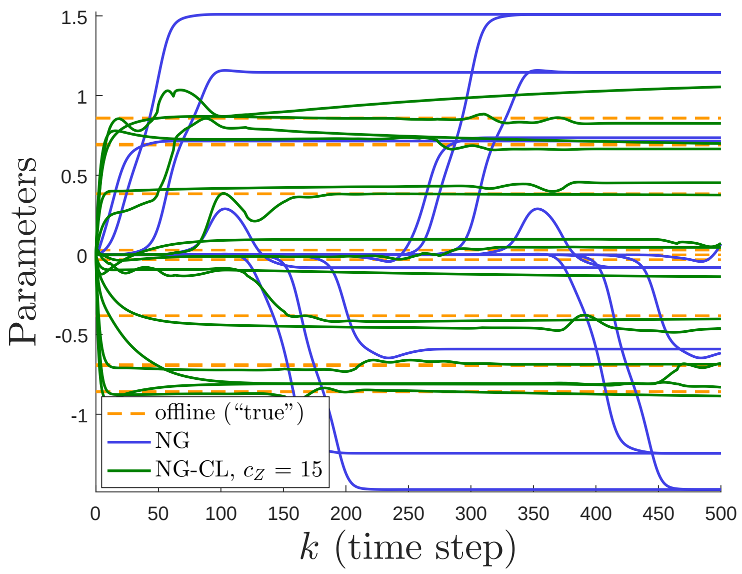
Experiment 7

We save the history stacks obtained at the end of Experiment 6. We then proceed to rerun the algorithms on-line for estimation of y_u while starting with the previously saved histories. In essence, we are guaranteed that the rank condition is verified even before the parameter updates take place. Rerunning the approximation algorithms, we get Figures 7.15 and 7.16 while respectively using DRP1 and DRP2. The DTNG based CL algorithm does much better than before (Experiment 6), both function approximation wise and parameter estimation wise. Its learning generalization is further improved. The reason for undertaking this experiment was to show how good an approximation the DTNG based CL algorithm could provide (Figures 7.15a and 7.16a compared to Figures 7.11a and 7.13a) if the rank condition is verified and the value of r_{CL} is significant. Recall that the rate of convergence of the DTNG based CL algorithm is dependent upon the spectral properties of the history stack Z_G.

Although rerunning the DTNG based CL algorithm using saved up histories yielded better approximation of the uncertainty, the fact remains that when implementing on-line approximation, we usually have to start with an empty Z or Z_G. Next, we look into how we can make the function approximation performance better even when Z starts empty.

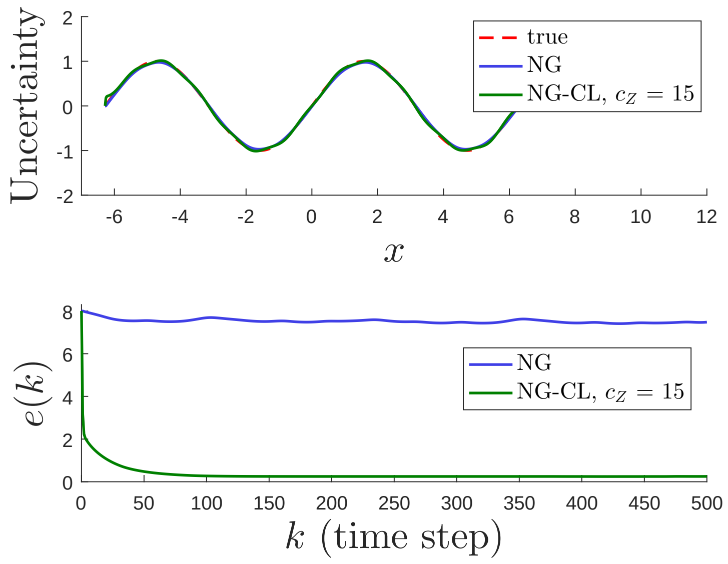


(a) On-line uncertainty approximation and metric $e(k)$.

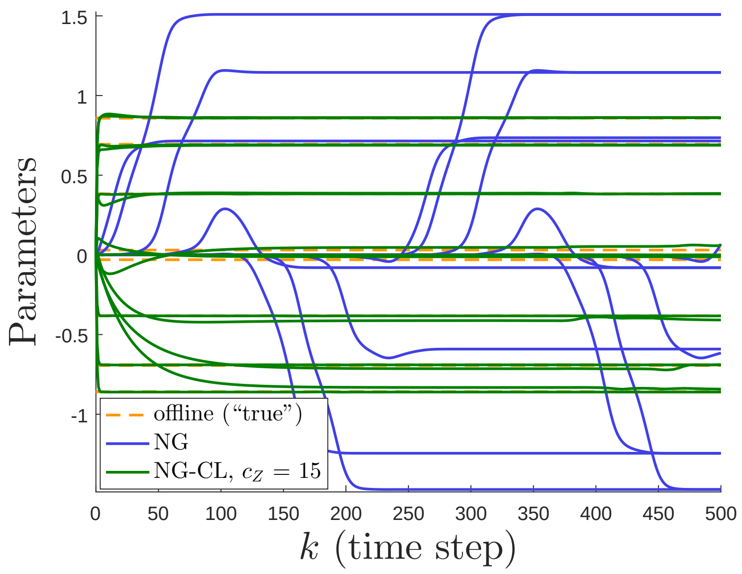


(b) On-line parameter approximation.

Figure 7.15: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) starting with saved history stack Z obtained from Experiment 6 while using Gradient Descent based algorithms: DRP1 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.

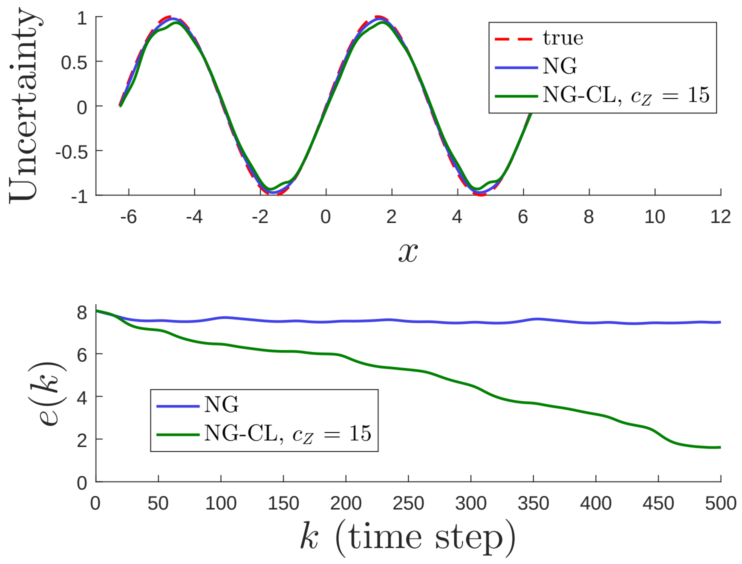


(a) On-line uncertainty approximation and metric $e(k)$.

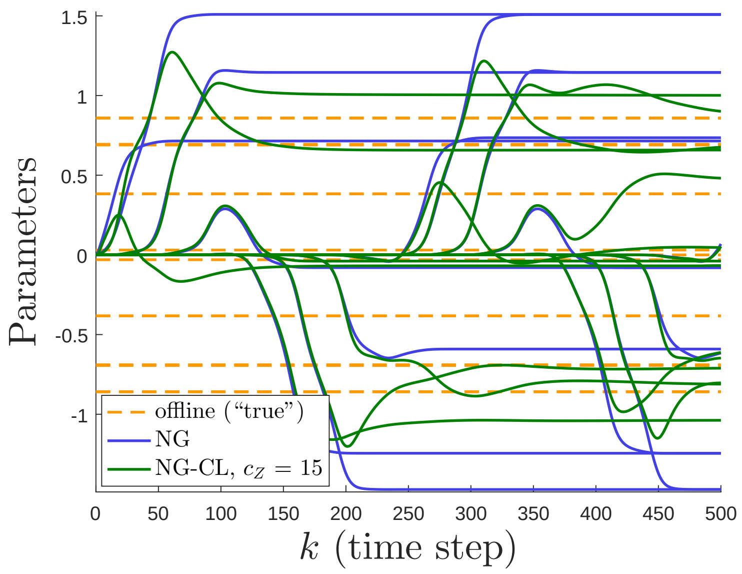


(b) On-line parameter approximation.

Figure 7.16: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) starting with saved history stack Z obtained from Experiment 6 while using Gradient Descent based algorithms: DRP2 is used for CL, $\varepsilon_{NG} = \varepsilon_{CL} = 1$.

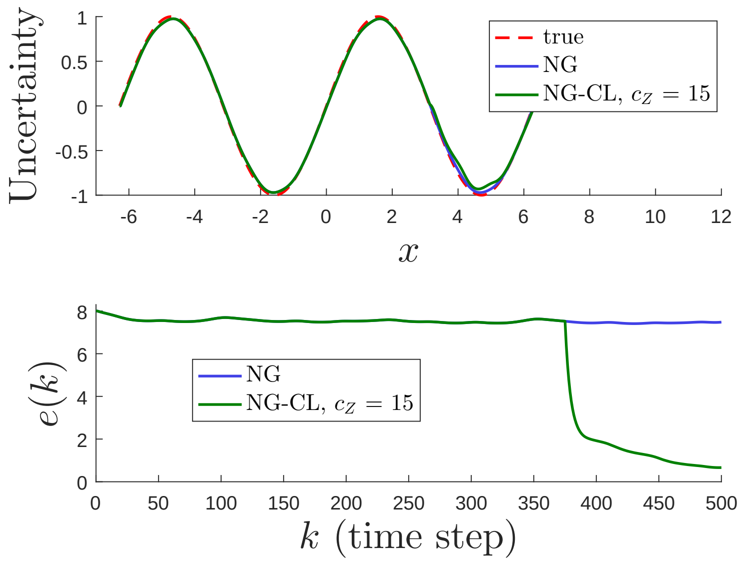


(a) On-line uncertainty approximation and metric $e(k)$.

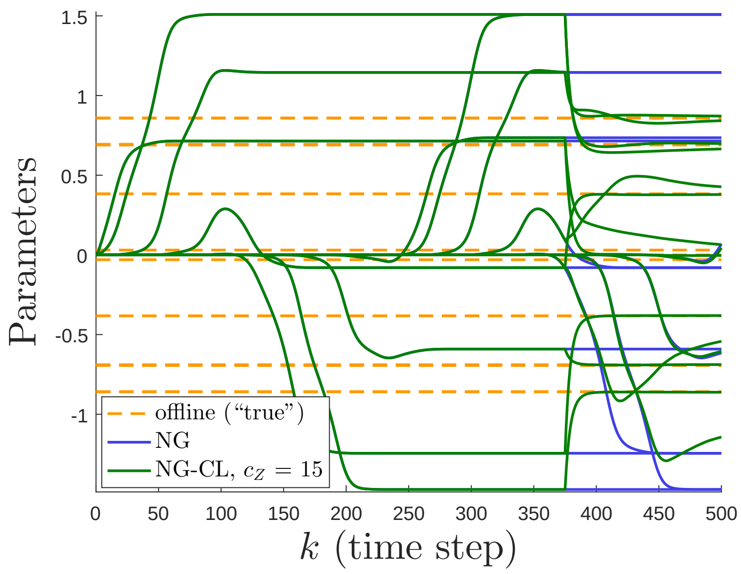


(b) On-line parameter approximation.

Figure 7.17: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Gradient Descent based algorithms: DRP1 is used for CL, $\varepsilon_{NG} = 1$, and $\varepsilon_{CL} = 0.1$.



(a) On-line uncertainty approximation and metric $e(k)$.



(b) On-line parameter approximation.

Figure 7.18: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Gradient Descent based algorithms: DRP2 is used for CL, $\varepsilon_{NG} = 1$, and $\varepsilon_{CL} = 0.01$.

Experiment 8

Recall that in the case of the DTNG based CL algorithm with DRP2 and in Exprement 6, the modified DTNG algorithm is first implemented in place of CL till the rank condition is guaranteed, and, only then, is the full DTNG based CL algorithm carried out. Doing so, the approximation result (Figure 7.13a) only degraded after the CL rank condition was verified and, equivalently, after the inclusion of the update term based on recorded data $\Delta\hat{\theta}_{CL}$. It therefore does seem that the adaptation based on recorded data, i.e., $\Delta\hat{\theta}_{CL}$, affects the one based on current data, i.e., $\Delta\hat{\theta}_{NG}$, at least as far as function approximation performance is concerned. While $\Delta\hat{\theta}_{NG}$ seeks to address instantaneous approximation, $\Delta\hat{\theta}_{CL}$ aims for better learning generalization. Hence, unless the parameter updates are being pulled in the same directions, $\Delta\hat{\theta}_{NG}$ and $\Delta\hat{\theta}_{CL}$ could be working destructively. To alleviate the effect of $\Delta\hat{\theta}_{CL}$ on $\Delta\hat{\theta}_{NG}$ (or vice versa), we can set ε_{NG} and ε_{CL} to different values.

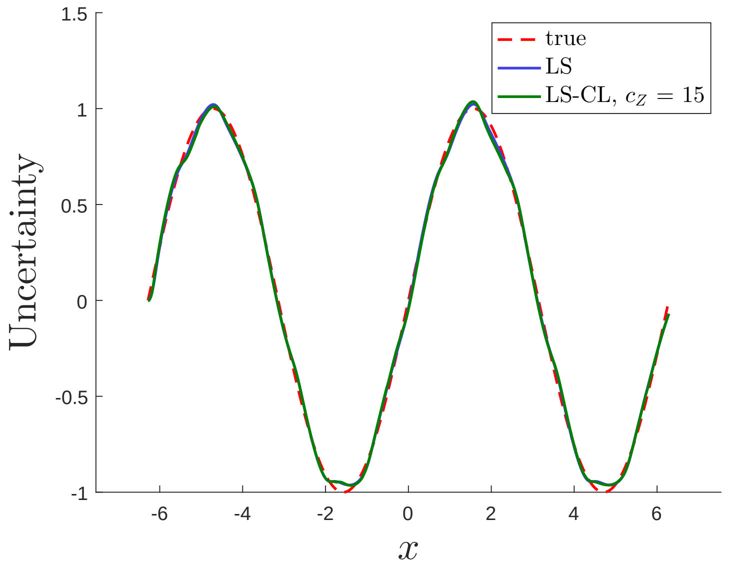
First, while using DRP1, we set $\varepsilon_{NG} = 1$ and $\varepsilon_{CL} = 10\%\,\varepsilon_{NG} = 0.1$. Simulation results are shown on Figure 7.17. Then, using DRP2 and setting $\varepsilon_{NG} = 1$ and $\varepsilon_{CL} = 1\%\,\varepsilon_{NG} = 0.01$ yields the results on Figure 7.18. As those figures demonstrate, good function approximation performance can still be achieved while also achieving good parameter estimation.

7.2.2 Applying Least Squares Based Algorithms

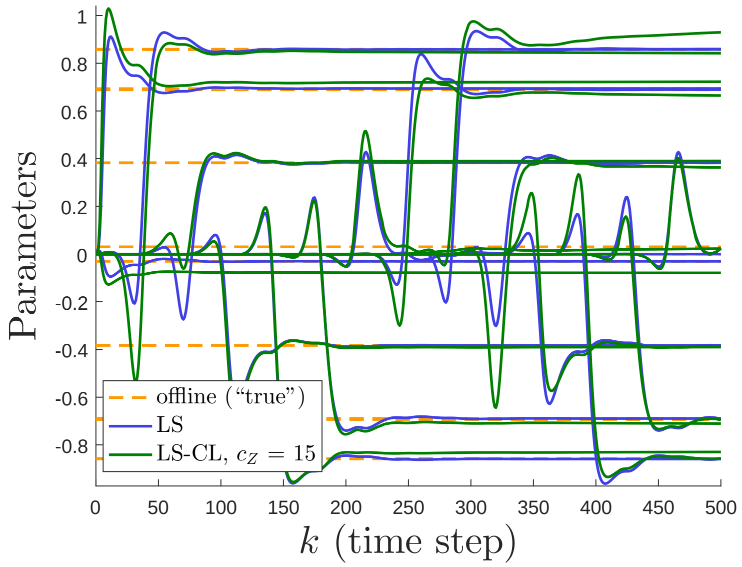
Experiment 9

Letting $\varepsilon_{\phi} = \varepsilon_{Z} = 1$ for DTNRLS based CL implementations, we obtain the results shown on Figures 7.19 and 7.20 with DRP1 whereas Figures 7.21 and 7.22 are obtained when using DRP2.

By comparing Figure 7.19 to Figure 7.21 (more so, Figure 7.19b compared to Figure 7.21b), it is clear in this case that DRP1 and DRP2 lead to different results. DRP2, for which the DTNRLS based CL algorithm is only applied if the rank condition is verified (here, around $k = 375$), achieves a much higher r_{CL} (see Figure 7.22b) than DRP1 (see Figure 7.20b), which is more inclined to be

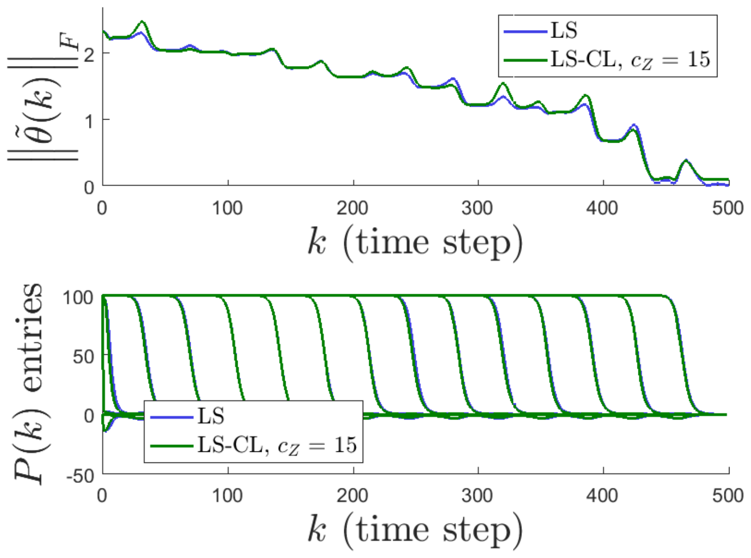


(a) On-line uncertainty approximation.

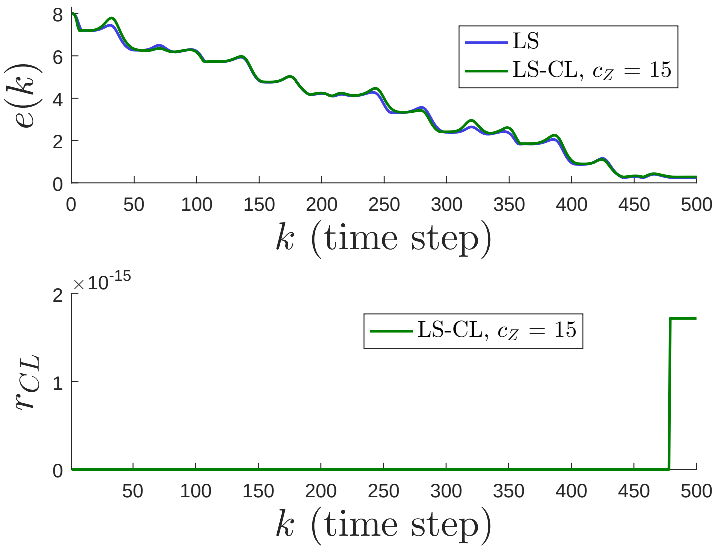


(b) On-line parameter estimation.

Figure 7.19: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP1 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.

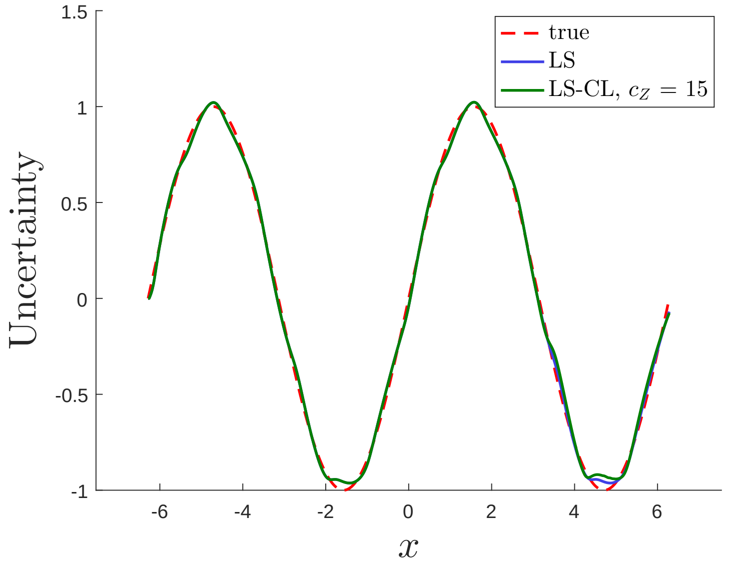


(a) Frobenius norm of parameter error and entries in $P(k)$.

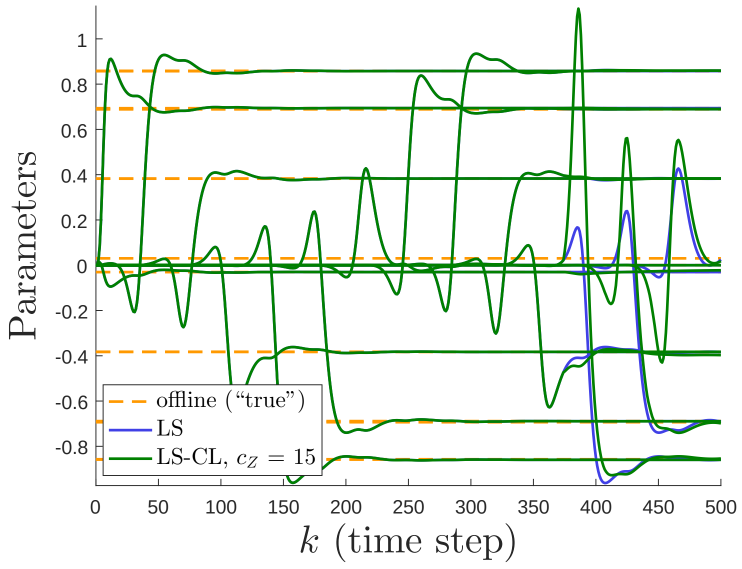


(b) Metric $e(k)$ and ratio r_{CL}.

Figure 7.20: Frobenius norm $\left\|\tilde{\theta}(k)\right\|_F$, entries in $P(k)$, metric $e(k)$, and ratio r_{CL} given on-line approximation of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP1 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.

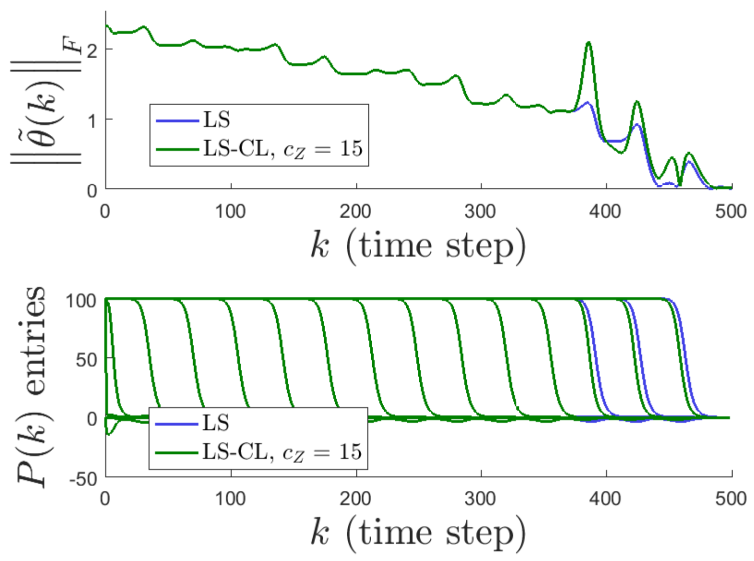


(a) On-line uncertainty approximation.

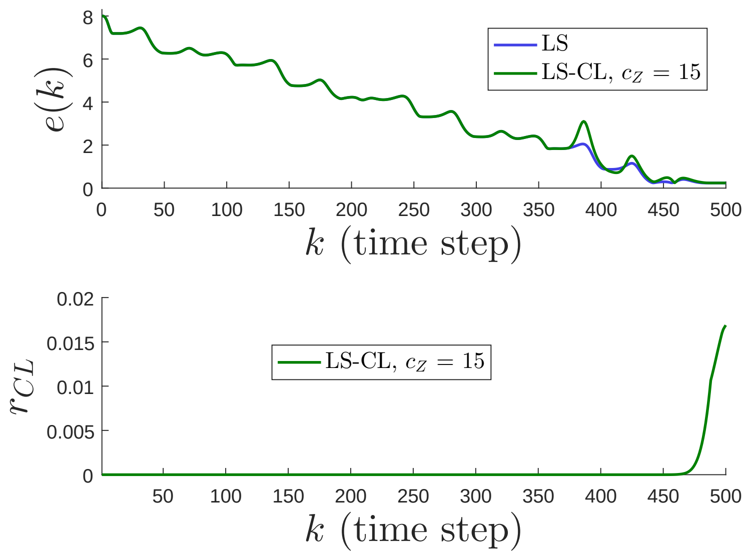


(b) On-line parameter estimation.

Figure 7.21: Standalone on-line training results of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP2 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.



(a) Frobenius norm of parameter error and entries in $P(k)$.



(b) Metric $e(k)$ and ratio r_{CL}.

Figure 7.22: Frobenius norm $\left\|\tilde{\theta}(k)\right\|_F$, entries in $P(k)$, metric $e(k)$, and ratio r_{CL} given on-line approximation of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP2 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.

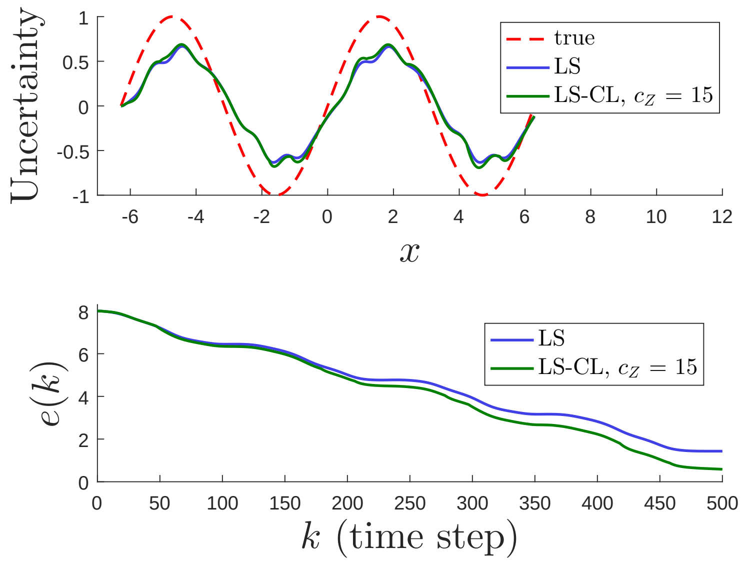
biased towards initially recorded data.

It should be noted that, regardless of the data recording technique used, as far as function approximation is concerned, both the DTNRLS algorithm and its CL modification do well (see Figures 7.19a and 7.21a). The entries in $P(k)$ achieve zero (or near to zero) steady states (see Figures 7.20a and 7.22a), thus, substantiating (5.32) and (5.68). Additionally, when it comes to parameter estimation, plots of the metric $e(k)$ on Figures 7.20b and 7.22b show that good learning is achieved for both algorithms.

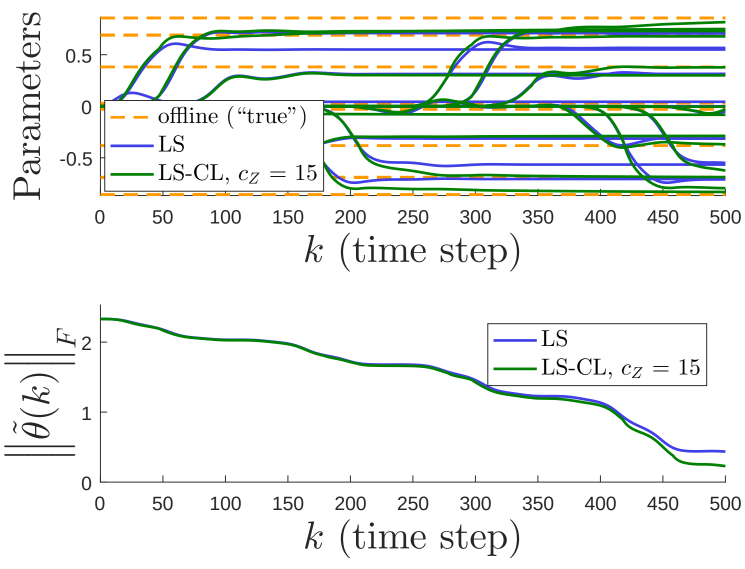
Hence, one may wonder if there is any benefit to applying the DTNRLS based CL algorithm. If so, remember first and foremost that we have theoretically proven that the CL modification of the standard DTNRLS algorithm does in fact lead to convergence of the parameter estimates under less restrictive conditions than the DTNRLS algorithm. Furthermore, to show how the DTNRLS based CL algorithm improves the standard DTNRLS algorithm we set the inital gain matrix to $P(k_0 - 1) = P_0 = 0.1\, I_{r_\psi}$ and rerun both technique for approximation of y_u. Figures 7.23 and 7.24 are obtained when using respectively DRP1 and DRP2. Here, because the maximum eigenvalue of P_0 is set low at the start (i.e., 0.1 versus 100, as we have done before), the approximation performance of both algorithms suffers. Nevertheless, the inclusion of selectively picked data in the parameter update law helps clearly separate the DTNRLS based CL algorithm from the standard DTNRLS algorithm, as Figures 7.23 and 7.24 show.

Experiment 10

Having saved the history stacks obtained from DTNRLS based CL implementations at the end of Experiment 9 (when using $P(k_0 - 1) = P_0 = 100\, I_{r_\psi}$), we rerun the learning algorithms on-line for approximation of y_u. That means we start with non-empty Z matrices and are guaranteed verification of the rank condition at $k = 0$ given that, from Figures 7.20b and 7.22b, $r_{CL} \neq 0$ at the end of the simulations. We get Figures 7.25 and 7.26 while respectively using DRP1 and DRP2. The

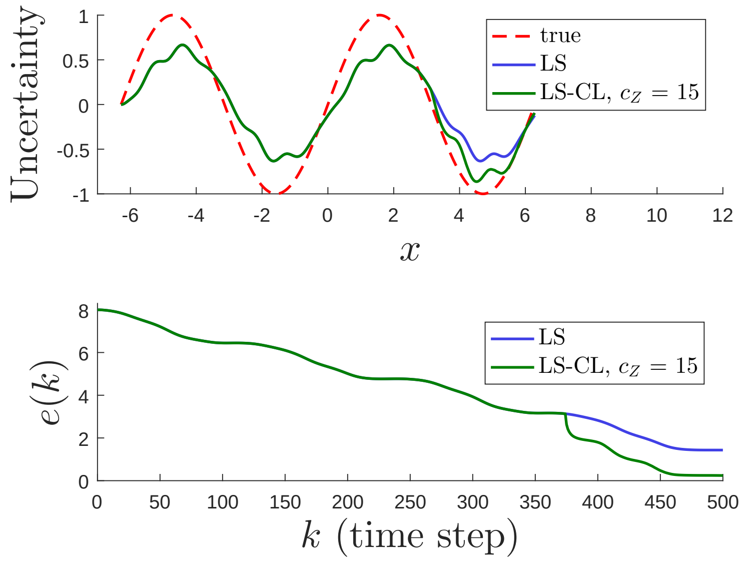


(a) On-line uncertainty approximation and metric $e(k)$.

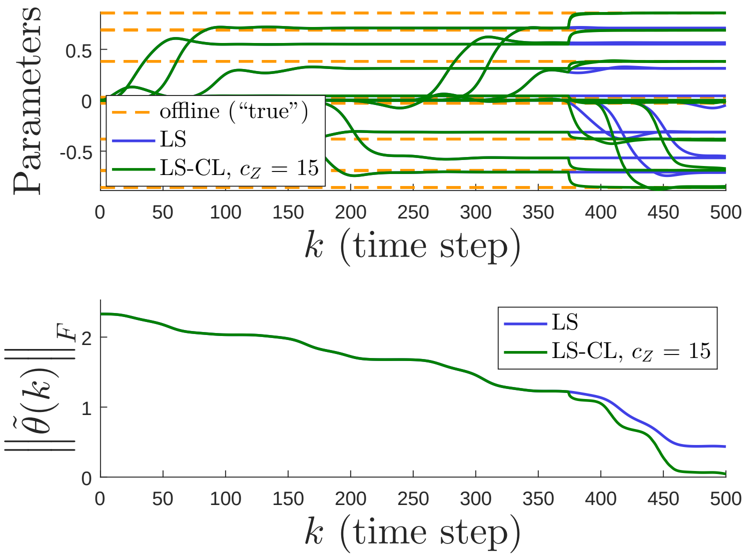


(b) On-line parameter approximation and Frobenius norm of parameter error.

Figure 7.23: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP1 is used for CL, $P_0 = 0.1\, I_{r\psi}$, $\varepsilon_\phi = \varepsilon_Z = 1$.

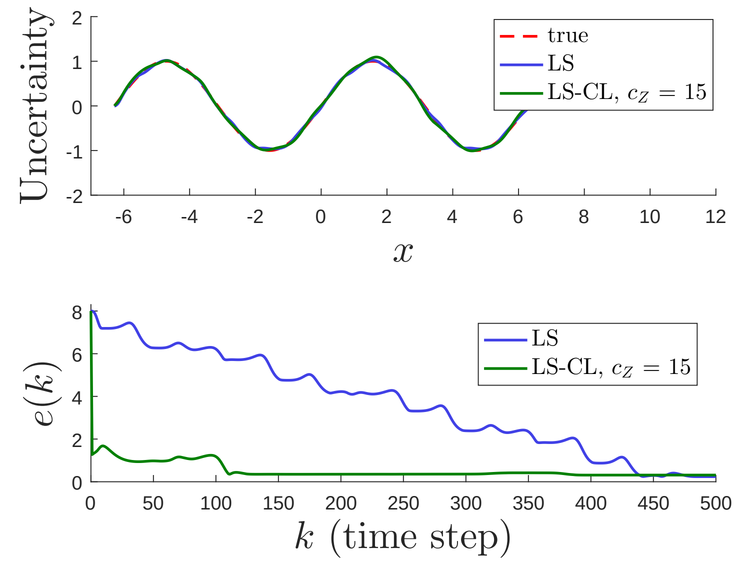


(a) On-line uncertainty approximation and metric $e(k)$.

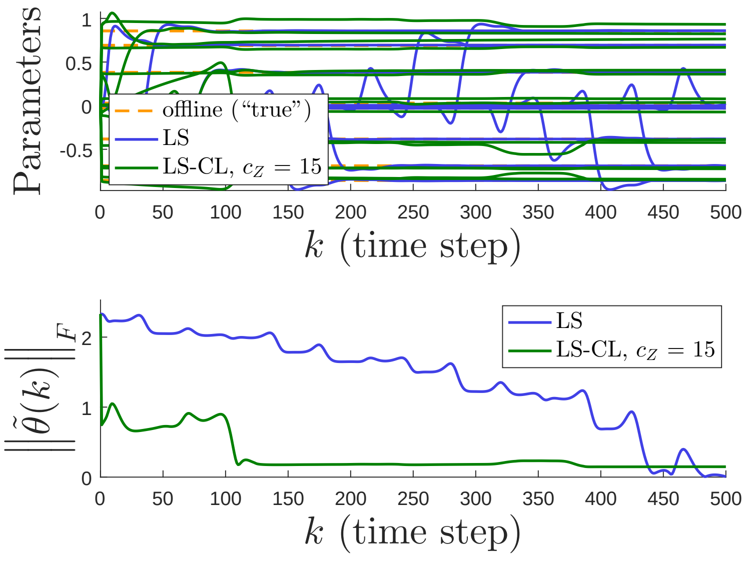


(b) On-line parameter approximation and Frobenius norm of parameter error.

Figure 7.24: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP2 is used for CL, $P_0 = 0.1\, I_{r\psi}$, $\varepsilon_\phi = \varepsilon_Z = 1$.

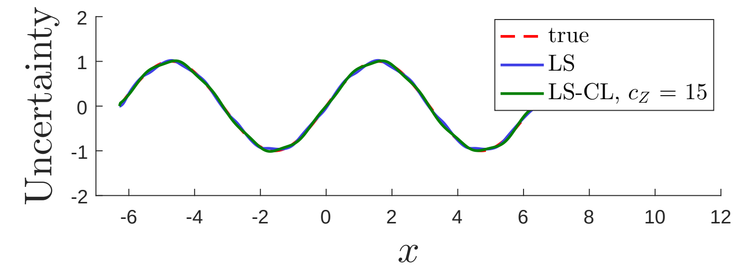


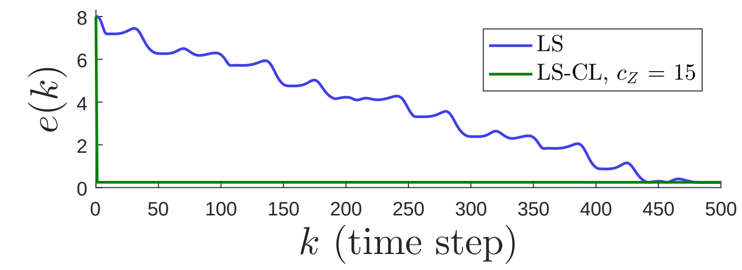
(a) On-line uncertainty approximation and metric $e(k)$.



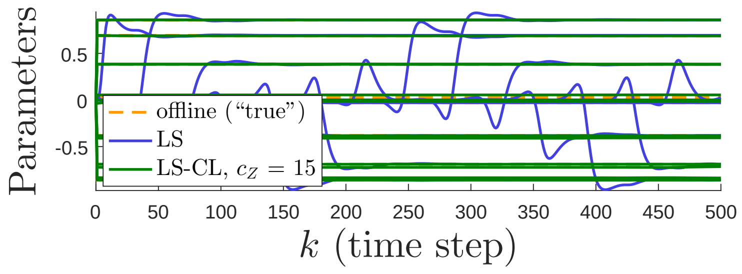
(b) On-line parameter approximation and Frobenius norm of parameter error.

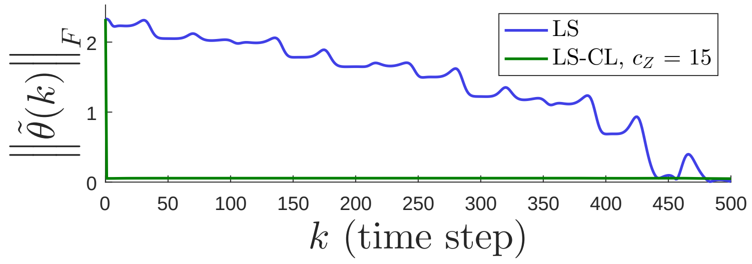
Figure 7.25: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) starting with saved history stack Z obtained from Experiment 9 while using Least Squares based algorithms: DRP1 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.





(a) On-line uncertainty approximation and metric $e(k)$.





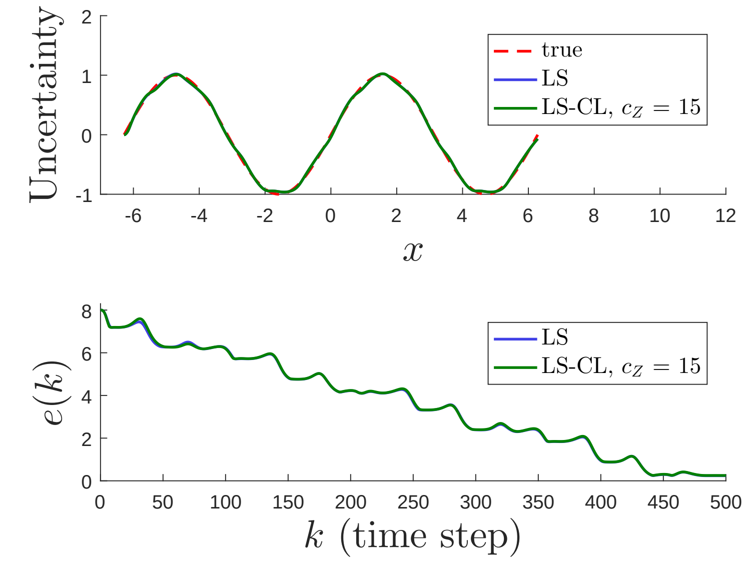
(b) On-line parameter approximation and Frobenius norm of parameter error.

Figure 7.26: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) starting with saved history stack Z obtained from Experiment 9 while using Least Squares based algorithms: DRP2 is used for CL, $\varepsilon_\phi = \varepsilon_Z = 1$.

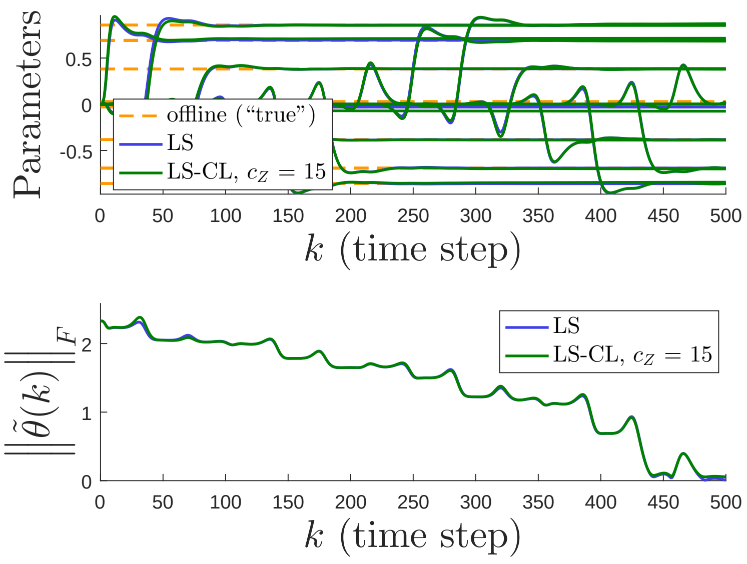
convergence of the DTNRLS based CL algorithm is much faster than in Experiment 9, especially when using DRP2 (see Figure 7.25b for DRP1 and Figure 7.26b for DRP2). However, it should be said that, though fast in the opening stages, convergence does stall towards as k increases.

Alike when running Gradient Descent based algorithms, the main goal behind running this experiment was to show that the DTNRLS based CL algorithm could provide good function approximation granted the rank condition is not only verified, but, also, the value of r_{CL} is significant enough. After all, it has been shown in **Remark 5.2.2** that the convergence of the DTNRLS based CL algorithm is a function of $r_{CL,1}$ in (5.71) or $r_{CL,2}$ in (5.72), hence, indirectly function of the spectral properties of the history stack Z. The present experiment also shows how off-line data (especially, if rich enough, i.e., verifies the rank condition) can be used when running the DTNRLS based CL algorithm on-line. Moreover, running the same algorithms as in Experiment 9, but, with the added saved data, and getting faster convergence results shows that the DTNRLS based CL could work very well when dealing with similar and/or repetitive tasks to be identified over and over.

Though using saved up data can be beneficial, when running on-line approximation with the DTNRLS based CL algorithm, there usually are no other choices than starting with an empty Z. From a parameter approximation point of view, we clearly see from the plots of $\left\|\tilde{\theta}(k)\right\|_F$ versus time on Figures 7.20a and 7.22a that, for the most part, CL methods might actually under perform a bit compared to the standard DTNRLS algorithm (at least for the present example and when the maximum eigenvalue of the initial guess of the gain matrix is set to be large). Therefore, next, we look into how those lapses in performance can be alleviated even when starting with an empty Z.

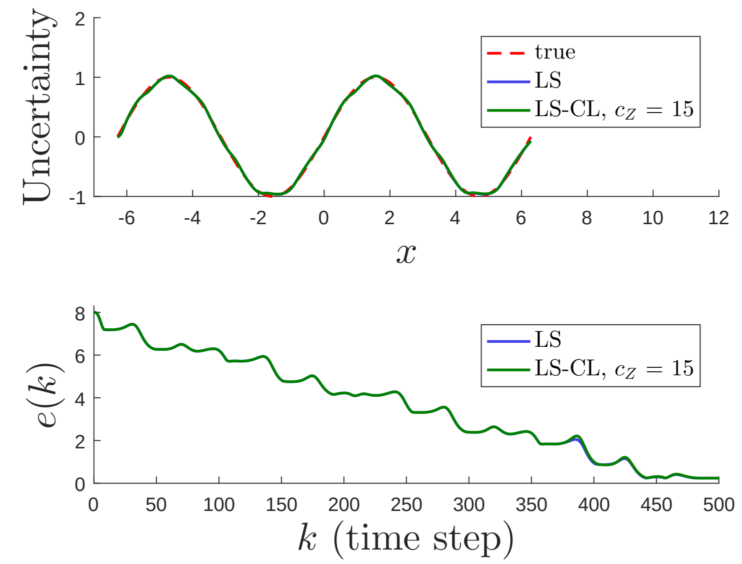


(a) On-line uncertainty approximation and metric $e(k)$.

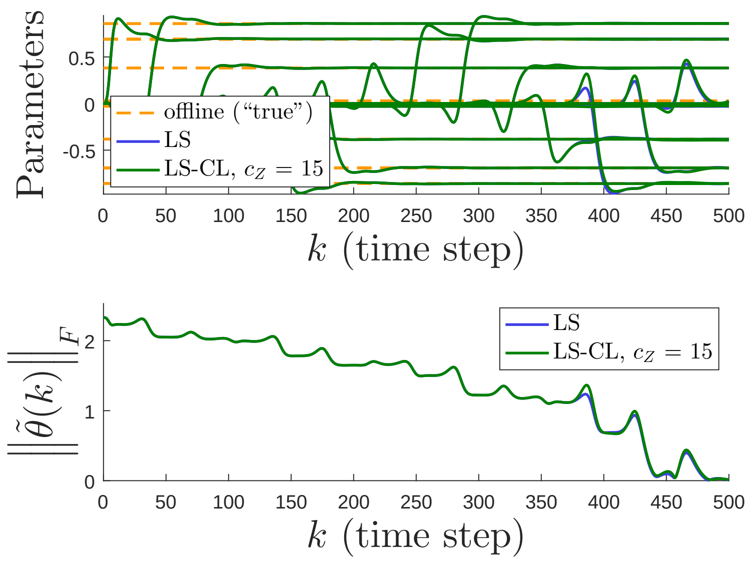


(b) On-line parameter approximation and Frobenius norm of parameter error.

Figure 7.27: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP1 is used for CL, $\varepsilon_\phi = 1$, and $\varepsilon_Z = 0.1$.



(a) On-line uncertainty approximation and metric $e(k)$.



(b) On-line parameter approximation and Frobenius norm of parameter error.

Figure 7.28: Standalone on-line approximation results of the unstructured uncertainty described by (7.4) while using Least Squares based algorithms: DRP2 is used for CL, $\varepsilon_\phi = 1$, and $\varepsilon_Z = 0.1$.

Experiment 11

When running DRP2, as mentioned before, the standard DTNRLS algorithm is applied first and it is when the rank condition comes to be verified (after collection of data) that the CL modification is implemented. Looking back at Experiment 9, we see when using the DTNRLS based CL algorithm with DRP2 that $\left\|\tilde{\theta}(k)\right\|_F$ and $e(k)$ only degraded around or after the $k = 375$ mark (see Figure 7.22), which coincides with the full application of the CL modification. Hence, it seems that incorporating the error based on recorded data $q_{Z,\epsilon}(k)$ into the overall approximation error $Q_\epsilon(k)$ does conflict with the instantaneous error $q_\epsilon(k)$. To create more balance or to choose the more influential approximation error, next, we experiment with the weighting matrices ξ_ϕ and ξ_Z.

While using both DRP1 and DRP2, we set $\varepsilon_\phi = 1$ and $\xi_Z = 10\%\varepsilon_\phi = 0.1$. Simulation results are shown on Figures 7.27 and 7.28 respectively. Again, for this particular example the standard DTNRLS algorithm does really well. However, as the previously mentioned figures show, besides the theoretical guarantees, we obtain good performances all around using the CL algorithm by appropriately setting the weights ξ_{NG} and ξ_Z.

7.3 Concluding Remarks

Overall, we have showed via illustrations in this chapter why Concurrent Learning and the use of memory can help achieve better learning, though, in the case of the Least Squares based Concurrent Learning scheme, it is a little bit difficult to show just that, especially when the spectral properties of P_0 are picked to be (very) large.

As we have pointed out in **Remark 5.2.1**, it could be argued that the standard DTNRLS algorithm is a memory based method. If so, the challenge is to show that the standard DTNRLS algorithm with an improved (or increased) external memory, which the DTNRLS based CL algorithm could (rightly) be described as, can actually bring about better results. That is what we have done in Experiment 9 when setting the minimum and maximum eigenvalues of the initial guess of the gain matrix low before running the learning algorithms. We would like also to emphasize the theoretical results obtained when using the DTNRLS based CL algorithm, i.e., the convergence of the parameter error to the origin asymptotically under less demanding condition.

CHAPTER VIII

APPLICATION 1: DISCRETE-TIME INDIRECT ADAPTIVE CONTROL OF A CLASS OF SINGLE STATE SYSTEMS USING GRADIENT BASED CONCURRENT LEARNING FOR PARAMETER ADAPTATION

We use the discrete-time Normalized Gradient based CL algorithm developed in Chapter IV for parameter estimates adaptation when designing a discrete-time adaptive controller for a single state discrete-time system bearing structured uncertainties. We only focus on the well studied indirect adaptive control (IAC) scheme [2, 3, 4, 5]. We show that if the Concurrent Learning condition, i.e., **Condition 3.1.1**, is verified, we can achieve both exponential error tracking and, simultaneously, exponential convergence of the parameter estimates to their true values.

Some of the previously used notations, which we advise against referring back to, have been changed in the present chapter.

8.1 Problem Statement

Consider the uncertain single state discrete-time system (or plant) given by

$$x(k+1) = f\left(\chi(k)\right) + g\left(\chi(k)\right) u(k), \tag{8.1}$$

where, at the discrete time-step $k \in \{k_0, k_0+1, k_0+2, \ldots\}$, with $k_0 \in \mathbb{N}$ being the initial starting time-step, $x(k) \in \mathbb{R}$ and $u(k) \in \mathbb{R}$ respectively are the state and the control input in (8.1). The causal vector $\chi(k) \in \mathbb{R}^{r_\chi}$, $r_\chi \in \mathbb{N}^+$, which appears in (8.1), can consist of known, computable and/or available for measurement past and present signals. The uncertain nature of (8.1) stems from the plant dynamics $f(\chi(k)) \in \mathbb{R}$ and $g(\chi(k)) \in \mathbb{R}$ being unknown functions of the signals in $\chi(k)$.

Assumption 8.1.1. *The functions $f(\chi(k))$ and $g(\chi(k))$ are structured uncertainties.*

Assumption 8.1.2. *The function $g(\chi(k))$ is bounded away from zero for all k. Further, there exists some known $\underline{g} > 0$, such that $|g(\chi(k))| \geq \underline{g}$.*

Per **Assumption 8.1.1**, there exist ideal, constant, but unknown parameter vectors $\theta_f \in \mathbb{R}^{r_{\theta_f}}$, $r_{\theta_f} \in \mathbb{N}^+$, and $\theta_g \in \mathbb{R}^{r_{\theta_g}}$, $r_{\theta_g} \in \mathbb{N}^+$, and known regressor vectors $\phi_f(\chi(k)) \in \mathbb{R}^{r_{\theta_f}}$ and $\phi_g(\chi(k)) \in \mathbb{R}^{r_{\theta_g}}$ such that, for all $\chi(k) \in \mathbb{R}^{r_\chi}$,

$$f(\chi(k)) = \theta_f^\top \phi_f(\chi(k)), \tag{8.2a}$$

$$g(\chi(k)) = \theta_g^\top \phi_g(\chi(k)). \tag{8.2b}$$

As for **Assumption 8.1.2**, we require that $g(\chi(k)) \neq 0$ for all k so that the control action $u(k)$ is not lost at any instance of time. We will later show how $\underline{g}$ can be made used of.

Our goal is to design a closed-loop discrete-time control action $u(k)$ that causes the state $x(k)$ to track a bounded reference $r(k) \in \mathbb{R}$, even in presence of the uncertainties f and g.

We assume knowledge of not only $r(k)$ but also $r(k+1)$. If $r(k+1)$ is not available, a stable reference model whose state $x_m(k)$ asymptotically tracks $r(k)$ can be constructed. That way, both $x_m(k)$ and $x_m(k+1)$ can be made available for use when computing the control action. We will later on show how such a reference model can be devised. Otherwise, if $r(k+1)$ is exactly known, then let $x_m(k) = r(k)$ and $x_m(k+1) = r(k+1)$.

We define the tracking error $e(k) \in \mathbb{R}$

$$e(k) = x(k) - x_m(k) \tag{8.3}$$

to quantify closed-loop performance of (8.1). If $e(k) \to 0$, then $x(k) \to x_m(k) \to r(k)$.

We approach the present design problem in a somewhat similar fashion to Lyapunov design and analysis procedures commonly undertaken for CT systems (see [2, 3, 4, 5]). First, we express $e(k+1)$ as

$$e(k+1) = x(k+1) - x_m(k+1) = f\left(\chi(k)\right) + g\left(\chi(k)\right)u(k) - x_m(k+1). \tag{8.4}$$

Assume for a moment that f and g are exactly known and consider the stabilizing controller

$$\overline{u}(k) = \frac{\gamma_e e(k) - f\left(\chi(k)\right) + x_m(k+1)}{g\left(\chi(k)\right)}, \; |\gamma_e| < 1. \tag{8.5}$$

If $u(k) = \overline{u}(k)$, from (8.4), we get that $e(k+1) = \gamma_e e(k)$; and, a Lyapunov proof can then be put forth to show that $e(k) \to 0$ or $x(k) \to x_m(k)$ exponentially.

However, with f and g being unknown, it is impossible to implement (8.5). We therefore resort to using adaptive control, and, more precisely, an IAC scheme.

8.2 Discrete-Time Indirect Adaptive Control

To implement the Indirect Adaptice Control (IAC) scheme, we first have to approximate the unknown plant dynamics f and g and use their approximations to define the adaptive controller. Let $\mathcal{F}$ and $\mathcal{G}$ be the approximations of f and g, respectively. Given the expressions f in (8.2a) and g in (8.2b), we let

$$\mathcal{F}(k) = \mathcal{F}\left(\phi_f\left(\chi(k)\right), \hat{\theta}_f(k)\right) = \hat{\theta}_f^\top(k)\phi_f\left(\chi(k)\right), \tag{8.6}$$

$$\mathcal{G}(k) = \mathcal{G}\left(\phi_g\left(\chi(k)\right), \hat{\theta}_g(k)\right) = \hat{\theta}_g^\top(k)\phi_g\left(\chi(k)\right), \tag{8.7}$$

where the time-varying parameter estimate vectors $\hat{\theta}_f(k) \in \mathbb{R}^{r_{\theta_f}}$ and $\hat{\theta}_g(k) \in \mathbb{R}^{r_{\theta_g}}$ are estimates of, respectively, θ_f and θ_g. Based on (8.5), for some γ_e such that $0 < |\gamma_e| < 1$, the indirect adaptive controller is defined as

$$u_i(k) = \frac{\gamma_e e(k) - \mathcal{F}(k) + x_m(k+1)}{\mathcal{G}(k)}. \tag{8.8}$$

We abstain from the choice of $\gamma_e = 0$ for reasons that will soon become clearer. The subscript i (in u_i) will be used throughout this chapter to designate *indirect*, as in IAC. Setting $u(k) = u_i(k)$ in (8.4), adding $\mathcal{G}(k)\left(u_i(k) - u_i(k)\right) = 0$ to the right hand side of (8.4), and using (8.8), we get

$$e(k+1) = \gamma_e e(k) - \hat{T}_i(k) + T_i(k), \tag{8.9}$$

where, from (8.2a), (8.2b), (8.6), and (8.7), the signal

$$\hat{T}_i(k) = \mathcal{F}(k) + \mathcal{G}(k) u_i(k) = \hat{\theta}_i^\top(k) \phi_i\left(\chi(k)\right) \tag{8.10}$$

can be considered an estimate of its true value

$$T_i(k) = f\left(\chi(k)\right) + g\left(\chi(k)\right) u_i(k) = \theta_i^\top \phi_i\left(\chi(k)\right), \tag{8.11}$$

with $\hat{\theta}_i(k) = \left[\hat{\theta}_f^\top(k), \hat{\theta}_g^\top(k)\right]^\top \in \mathbb{R}^{r_{\theta_i}}$, $r_{\theta_i} = r_{\theta_f} + r_{\theta_g}$, being the parameter estimate vector of the new, augmented unknown parameter vector $\theta_i = \left[\theta_f^\top, \theta_g^\top\right]^\top \in \mathbb{R}^{r_{\theta_i}}$ and

$$\phi_i^\top\left(\chi(k)\right) = \left[\phi_f^\top\left(\chi(k)\right), \phi_g^\top\left(\chi(k)\right) u_i(k)\right]^\top \in \mathbb{R}^{r_{\theta_i}}$$

being the new, augmented regressor vector. Let $\tilde{\theta}_i(k) = \hat{\theta}_i(k) - \theta_i \in \mathbb{R}^{r_{\theta_i}}$ denote the parameter error vector. We define the approximation error q_i as

$$q_i(k+1) = \hat{T}_i(k) - T_i(k) = \tilde{\theta}_i^\top(k) \phi_i\left(\chi(k)\right) \tag{8.12}$$

and rewrite (8.9) as

$$e(k+1) = \gamma_e e(k) - q_i(k+1). \tag{8.13}$$

Hence, from (8.13),

$$q_i(k+1) = \gamma_e e(k) - e(k+1) \tag{8.14}$$

gives us a way to numerically compute $q_i(k+1)$ for $k > k_0$. It should be added that, for convenience, many signals in this chapter (including q_i) are expressed at discrete-time $k+1$ even though actually computed at discrete time k, $k > k_0$, in simulation because they are causal.

Next, we investigate closed-loop behavior of error signals e and $\tilde{\theta}_i$, if the IAC scheme were to employ either one of the standard gradient descent algorithm or the gradient based CL algorithm (both, discussed in Chapter IV) for parameter adaptation.

Before going further, given our definitions in Chapter II, notice that (8.10) is expressed in the form of model (2.2a). Moreover, we are dealing here with a scalar approximation problem ($T_i \in \mathbb{R}$). The adaptation laws of Chapter IV will then be adjusted accordingly.

8.2.1 IAC with Normalized Gradient Adaptive Algorithm

In our current notation, for some gain $\eta_i > 0$ and given an initial parameter vector $\hat{\theta}_i\,(k_0)$, the discrete-time Normalized Gradient (NG) descent adaptive algorithm for $k > k_0$ is given by

$$\hat{\theta}_i(k+1) = \hat{\theta}_i(k) - \eta_i \Delta\hat{\theta}_{i_{NG}}(k), \tag{8.15}$$

where we define $\Delta\hat{\theta}_{i_{NG}}(k) \in \mathbb{R}^{r_{\theta_i}}$ as

$$\Delta\hat{\theta}_{i_{NG}}(k) = \frac{\phi_i\,(\chi(k))\, q_i(k+1)}{m_i^2(k)} \tag{8.16}$$

and, letting $\|\cdot\|$ denote the 2-norm operator, as we have done up to now, the normalization signal

$$m_i(k) = \sqrt{\alpha_i + \phi_i^\top(\chi(k))\,\phi_i\,(\chi(k))} = \sqrt{\alpha_i + \|\phi_i\,(\chi(k))\|^2},\, \alpha_i > 0,$$

defined after (4.1), ensures that $\phi_{m_i}\,(\chi(k)) \in \mathbb{R}^{r_{\theta_i}}$, expressed as

$$\phi_{m_i}\,(\chi(k)) = \frac{\phi_i\,(\chi(k))}{m_i(k)},$$

is bounded.

Let $\Phi_i(k) \in \mathbb{R}^{r_{\theta_i} \times r_{\theta_i}}$ be given by

$$\Phi_i(k) = \phi_i\,(\chi(k))\,\phi_i\,(\chi(k))^\top$$

and notice that $\Phi_i(k)$ is symmetric ($\Phi_i^\top(k) = \Phi_i(k)$) as well as positive semi-definite ($\Phi_i(k) \geq 0$) for all k. Also, let $\overline{\lambda}_{\Phi_i,k} = \lambda_{\max}\left(\Phi_i(k)\right)$, where, as defined before, $\lambda_{\max}(\cdot)$ denotes the maximum eigenvalue operator. Because $\Phi_i(k) \geq 0$, $\overline{\lambda}_{\Phi_i,k} > 0$ as long as $\phi_i\left(\chi(k)\right) \neq [0]^{r_{\theta_i}}$. From (8.12) and (8.16), the parameter error

$$\tilde{\theta}_i(k+1) = \hat{\theta}_i(k+1) - \theta_i$$

can also be expressed as

$$\tilde{\theta}_i(k+1) = \left[I_{r_{\theta_i}} - \frac{\eta_i \Phi_i(k)}{m_i^2(k)}\right] \tilde{\theta}_i(k), \tag{8.17}$$

where, as noted before, $I_{r_{\theta_i}} \in \mathbb{R}^{r_{\theta_i} \times r_{\theta_i}}$ represents the r_{θ_i}-by-r_{θ_i} identity matrix.

The positive definite Lyapunov function

$$V_{\tilde{\theta}_i}(k) = \tilde{\theta}_i^\top(k)\tilde{\theta}_i(k) = \left\|\tilde{\theta}_i(k)\right\|^2 \tag{8.18}$$

is used to study the parameter error $\tilde{\theta}_i(k)$ in (8.17). Denoting

$$\Delta\mathcal{V}(k+1) = \mathcal{V}(k+1) - \mathcal{V}(k)$$

for a given function $\mathcal{V}$ and letting $\beta_{i_1} = 2 - \dfrac{\eta_i \lambda_2(k)}{m_i^2(k)}$, we have shown in [28] as well as in Chapter IV (see (4.32), (4.35), and related expressions) that

$$\Delta V_{\tilde{\theta}_i}(k+1) \leq -\beta_{i_1} \frac{\eta_i q_i^2(k+1)}{m_i^2(k)} \leq 0 \tag{8.19}$$

so long as, for $\phi_i\left(\chi(k)\right) \neq [0]^{r_{\theta_i}}$, $\beta_{i_1} > 0$ or

$$0 < \eta_i < \frac{2m_i^2(k)}{\overline{\lambda}_{\Phi_i,k}} = \overline{\eta}_{i_{NG}}. \tag{8.20}$$

At any time step k, $\overline{\lambda}_{\Phi_i,k} = 0$ means $\phi_i\left(\chi(k)\right) = [0]^{r_{\theta_i}}$, which, given (8.16), implies $\hat{\theta}_i(k+1) = \hat{\theta}_i(k)$ and there is therefore no need to pick η_i. It can be shown, as done in Chapter IV, that $\overline{\eta}_{i_{NG}} < \infty$ and $0 < \beta_{i_1} < 2$ if η_i meets (8.20). In the end, based on (8.19), $V_{\tilde{\theta}_i}(k)$ is nonincreasing and, thus, bounded for all k.

Let us now study the tracking error $e(k)$. The ensuing analysis is adapted from [5] and is similar to what we have done in [9]. First, for some $\beta_{i_2} > 0$, consider the positive definite Lyapunov function

$$V_e(k) = \beta_{i_2} e^2(k), \tag{8.21}$$

For any a, $b \in \mathbb{R}$ and $\varepsilon > 0$, relationship (13.52) in [5] says

$$(a+b)^2 \le (1+\varepsilon)\, a^2 + \left(1 + \frac{1}{\varepsilon}\right) b^2. \tag{8.22}$$

Thus, for some $\varepsilon_i > 0$ and $\beta_{i_3} = 1 - \gamma_e^2(1+\varepsilon_i)$, we have from (8.13), (8.21), and using (8.22) that

$$\Delta V_e(k+1) \le -\beta_{i_3} V_e(k) + \beta_{i_2} \left(1 + \frac{1}{\varepsilon_i}\right) q_i^2(k+1). \tag{8.23}$$

We now put to use both (8.19) and (8.23). For some $\beta_{i_4} > 0$, we define the positive definite Lyapunov function

$$V_i(k) = V_e(k) + \beta_4 V_{\tilde{\theta}_i}(k). \tag{8.24}$$

Using (8.19), (8.23), and (8.24), we get that

$$\begin{aligned} \Delta V_i(k+1) &= \Delta V_e(k+1) + \beta_{i_4} \Delta V_{\tilde{\theta}_i}(k+1) \\ &\le -\beta_{i_3} V_e(k) - \frac{p_i(k) q_i^2(k+1)}{m_i^2(k)}, \end{aligned} \tag{8.25}$$

where

$$p_i(k) = \eta_i \beta_4 \beta_1 - \beta_{i_2} \left(1 + \frac{1}{\varepsilon_i}\right) m_i^2(k). \tag{8.26}$$

For stability result purposes, we require that $\beta_{i_3} > 0$, meaning

$$0 < \varepsilon_i < \frac{1-\gamma_e^2}{\gamma_e^2}, \tag{8.27}$$

which then leads to $0 < \beta_{i_3} < 1 - \gamma_e^2 < 1$ since $0 < |\gamma_e| < 1$, as well as $p_i(k) \geq 0$, which, from (8.26), means

$$\beta_{i_4} \geq \frac{\beta_{i_2}\left(1 + \frac{1}{\varepsilon_i}\right) m_i^2(k)}{\eta_i \beta_{i_1}}. \tag{8.28}$$

Now, assuming $\phi_i\left(\chi(k)\right)$ is bounded for all k, then, $m_i^2(k)$ is defined and, subsequently, so is β_{i_4}. By having $\beta_{i_3} > 0$ and $p_i(k) \geq 0$, we get from (8.25) that

$$\Delta V_i(k+1) = V_i(k+1) - V_i(k) \leq -\beta_{i_3} V_e(k) \leq 0. \tag{8.29}$$

Hence, both e and $\tilde{\theta}_i$ are bounded. Actually given (8.29) and the fact that V_i is positive definite, for all k, $V_i(k) \leq V_i(k_0)$. It is thus possible to uniformly bound both e and $\tilde{\theta}_i$. Applying **Theorem 13.5** in [5] here, we further have $\lim\limits_{k \to +\infty} V_e(k) = 0$, which, by (8.21), means e ultimately goes to zero. The same theorem allows us to write

$$V_e(k) \leq \max\left(V_e(k_0), \frac{\beta_{i_4}}{\beta_{i_3}} V_{\tilde{\theta}_i}(k_0)\right).$$

Thus, given (8.21) again, for all k,

$$|e(k)| \leq b_{iNG}^e = \sqrt{\max\left(e^2(k_0), \frac{\beta_{i_4}}{\beta_{i_2}\beta_{i_3}} \left\|\tilde{\theta}_i(k_0)\right\|^2\right)}. \tag{8.30}$$

For perspective, because $0 < \beta_{i_3} < 1$, $\varepsilon_i > 0$, $0 < \beta_{i_1} < 2$ and $0 < \eta_i \beta_1 < \dfrac{4m_i^2(k)}{\overline{\lambda}_{\Phi_i,k}}$ given (8.20), from (8.28),

$$\frac{\beta_{i_4}}{\beta_{i_2}\beta_{i_3}} > \max_k \left(\frac{m_i^2(k)}{\eta_i \beta_{i_1}}\right) > \frac{1}{4} \max_k \left(\overline{\lambda}_{\Phi_i,k}\right), \tag{8.31}$$

with $\overline{\lambda}_{\Phi_i,k} \leq \left\|\phi_i\left(\chi(k)\right)\right\|^2 < m_i^2(k)$ as shown in [28].

In all, when the IAC scheme of (8.8) is implemented with the NG algorithm of (8.16) used for parameter adaptation, both error signals e and $\tilde{\theta}_i$ are uniformly bounded. In addition, $e(k) \to 0$ as $k \to \infty$. Proving parameter identification, i.e., $\tilde{\theta}_i(k) \to [0]^{r_{\theta_i}}$, is however not possible unless ϕ_{m_i} is PE [3].

8.2.2 IAC with Normalized Gradient Based Concurrent Learning Adaptive Algorithm

To use CL for parameter adaptation, we first establish the uncertainty to be approximated. Iin this case, that uncertainty is T_i. Next, for $k > k_0$, we require knowledge of $T_i(k)$. This can be done in the two following fashions. For $k > k_0$, being able to compute $\hat{T}_i(k)$ using (8.10) and $q_i(k+1)$ using (8.14), from (8.12),

$$T_i(k) = \hat{T}_i(k) - q_i(k+1).$$

Also, referring back to (8.1) and (8.11), we have that

$$T_i(k) = f\left(\chi(k)\right) + g\left(\chi(k)\right) u_i(k) = x(k+1).$$

Now, let $Z_{G_i} \in \mathbb{R}^{r_{\theta_i} \times c_{Z_i}}$, for some $c_{Z_i} \in \mathbb{N}^+$, be a matrix whose columns are comprised of the normalized regressor vectors $\phi_{m_i}\left(\chi\left(\tau_{j_i}\right)\right)$, $G_i \in \mathbb{R}^{1 \times c_{Z_i}}$ be a row vector of $m_i\left(\tau_{j_i}\right)$ values and $F_i \in \mathbb{R}^{c_{Z_i}}$ be a vector containing the actual true values $T_i\left(\tau_{j_i}\right)$ recorded at some discrete time-step τ_{j_i} in the past, $k_0 \le \tau_{j_i} < k$, for $j_i = 1, 2, \ldots, c_{Z_i}$. That is

$$\begin{aligned}
Z_{G_i} &= \begin{bmatrix} \phi_{m_i}\left(\chi(\tau_1)\right), & \phi_{m_i}\left(\chi(\tau_2)\right), & \ldots, & \phi_{m_i}\left(\chi(\tau_{c_{Z_i}})\right) \end{bmatrix},\\
G_i &= \begin{bmatrix} m_i\left(\tau_1\right), & m_i\left(\tau_2\right), & \ldots, & m_i\left(\tau_{c_{Z_i}}\right) \end{bmatrix}, \text{ and}\\
F_i &= \begin{bmatrix} T_i\left(\tau_1\right), & T_i\left(\tau_2\right), & \ldots, & T_i\left(\tau_{c_{Z_i}}\right) \end{bmatrix}.
\end{aligned}$$

Notice that, because dealing with a vector regressor in this case, for any k, $m_i^2(k) = \bar{g}_{1,1}(k)$ for $\Xi_m = I_{r_{\theta_i}}$ and where $\bar{g}_{n,n}(k)$ (with $n = 1$ for the current problem) is defined in (3.3). Matrix Z_{G_i} will be our *history stack*. Moreover, notice that $\Phi_{Z_{G_i},k} = Z_{G_i} Z_{G_i}^\top \in \mathbb{R}^{r_{\theta_i} \times r_{\theta_i}}$ is symmetric. Additionally, much like (3.17)

$$\Phi_{Z_{G_i},k} = Z_{G_i} Z_{G_i}^\top = \sum_{j_i=1}^{c_{Z_i}} \phi_{m_i}\left(\chi\left(\tau_{j_i}\right)\right) \phi_{m_i}^\top\left(\chi\left(\tau_{j_i}\right)\right) = \sum_{j_i=1}^{c_{Z_i}} \frac{\phi_i\left(\chi(\tau_{j_i})\right) \phi_i^\top\left(\chi(\tau_{j_i})\right)}{m_i(k)}. \tag{8.32}$$

Given an initial parameter vector $\hat{\theta}_i(k_0)$, we pose for $k > k_0$ the gradient based Concurrent Learning algorithm adaptive algorithm

$$\hat{\theta}_i(k+1) = \hat{\theta}_i(k) - \eta_i \Delta\hat{\theta}_{i_{NG}}(k) - \eta_i \Delta\hat{\theta}_{i_{CL}}(k), \tag{8.33}$$

where $\Delta\hat{\theta}_{i_{NG}}(k)$ is defined in (8.16), the gain $\eta_i > 0$, and the adjustment term

$$\Delta\hat{\theta}_{i_{CL}}(k) = \sum_{j_i=1}^{c_{Z_i}} \frac{\phi_i\left(\chi\left(\tau_{j_i}\right)\right) q_{j_i}(k+1)}{m_i^2\left(\tau_{j_i}\right)} \tag{8.34}$$

features the approximation error based on recorded data q_{j_i}. Letting $E_{j_i}(k) = \hat{\theta}_i^\top(k)\phi_i\left(\chi\left(\tau_{j_i}\right)\right)$ take after (8.10), by using (8.11), and proceeding similarly to (8.12), we define

$$\begin{aligned} q_{j_i}(k+1) &= E_{j_i}(k) - T_i(\tau_{j_i}), \\ &= \tilde{\theta}_i^\top(k)\phi_i\left(\chi\left(\tau_{j_i}\right)\right) = \phi_i^\top\!\left(\chi\left(\tau_{j_i}\right)\right)\tilde{\theta}_i(k). \end{aligned} \tag{8.35}$$

For $k > k_0$, $q_{j_i}(k+1)$ can be numerically computed as

$$q_{j_i}(k+1) = \phi_i^\top\!\left(\chi\left(\tau_{j_i}\right)\right)\hat{\theta}_i(k) - T_i(\tau_{j_i}) \tag{8.36}$$

while using the stored data in Z_{G_i}, G_i, and F_i to do so. Given (8.32) and (8.35), $\Delta\hat{\theta}_{i_{CL}}(k)$ in (8.34) compactly reduces to

$$\begin{aligned} \Delta\hat{\theta}_{i_{CL}}(k) &= \sum_{j_i=1}^{c_{Z_i}} \frac{\phi_i\left(\chi\left(\tau_{j_i}\right)\right)\left[\phi_i^\top\left(\chi\left(\tau_{j_i}\right)\right)\tilde{\theta}_i(k)\right]}{m_i^2\left(\tau_{j_i}\right)} = \sum_{j_i=1}^{c_{Z_i}} \frac{\phi_i\left(\chi\left(\tau_{j_i}\right)\right)\phi_i^\top\!\left(\chi\left(\tau_{j_i}\right)\right)}{m_i^2\left(\tau_{j_i}\right)}\tilde{\theta}_i(k) \\ &= Z_{G_i} Z_{G_i}^\top \tilde{\theta}_i(k) = \Phi_{Z_{G_i},k}\,\tilde{\theta}_i(k). \end{aligned} \tag{8.37}$$

Hence, though we have expressed the adjustment term $\Delta\hat{\theta}_{i_{CL}}(k)$ differently here, i.e., (8.34) versus (4.59), both expressions are equivalent, as setting $\varepsilon_{w,k} = [0]^{r_\theta \times c_\theta}$, $r_\theta = r_{\theta_i}$ and $c_\theta = 1$ for the present problem, (4.61) and (8.37) are identical. Moving on, it then follows from (8.33) and (8.37) that the parameter error $\tilde{\theta}_i(k+1) = \hat{\theta}_i(k+1) - \theta_i$, in this case, can be written as

$$\tilde{\theta}_i(k+1) = \left[I_{r_{\theta_i}} - \frac{\eta_i \Phi_i(k)}{m_i^2(k)} - \eta_i Z_{G_i} Z_{G_i}^\top\right]\tilde{\theta}_i(k) = \left[I_{r_{\theta_i}} - \frac{\eta_i \Phi_i(k)}{m_i^2(k)} - \eta_i \Phi_{Z_{G_i},k}^\top\right]\tilde{\theta}_i(k). \tag{8.38}$$

We will impose the rank condition on Z_{G_i}, i.e., **Condition 3.1.1** for $Z_G = Z_{G_i}$. **Condition 3.1.1** requires that $c_{Z_i} \geq r_{\theta_i}$ and $\text{rank}(Z_{G_i}) = r_{\theta_i}$. As mentioned before, it is less restrictive than persistency of excitation because it only deals with a subset of past data, can be implemented and verified on-line, and can be achieved with ϕ_{m_i} being (sufficiently) exciting over a time sequence.

We state the following theorem, which we prove next.

Theorem 8.2.1. *Consider the single state plant defined by (8.1) bearing structured (or parametric) uncertainties. When implementing the IAC scheme of (8.8) while using the gradient based CL algorithm given by (4.58) for parameter adaptation,* $e(k) \to 0$ *and* $\hat{\theta}_i(k) \to \theta_i$*as* $k \to \infty$ *if the rank condition on* Z_{G_i}*, i.e.,* ***Condition 3.1.1*** *for* $Z_G = Z_{G_i}$*, is verified.*

Proof. We start the proof of **Theorem 8.2.1** by noting that, if **Condition 3.1.1** is met, then Z_{G_i} is full row rank and $\Phi_{Z_{G_i},k} = Z_{G_i} Z_{G_i}^\top$ is positive definite ($\Phi_{Z_{G_i},k} = Z_{G_i} Z_{G_i}^\top > 0$). Letting

$$\underline{\lambda}_{\Phi_{Z_{G_i}},k} = \lambda_{\min}\Big(\Phi_{Z_{G_i},k}\Big) = \lambda_{\min}\Big(Z_{G_i} Z_{G_i}^\top\Big)$$

and

$$\overline{\lambda}_{\Phi_{Z_{G_i}},k} = \lambda_{\max}\Big(\Phi_{Z_{G_i},k}\Big) = \lambda_{\max}\Big(Z_{G_i} Z_{G_i}^\top\Big),$$

with, this time around, $\lambda_{\min}(\cdot)$ denoting the minimum eigenvalue operator as initially defined, $\Phi_{Z_{G_i},k} = Z_{G_i} Z_{G_i}^\top > 0$ means $\overline{\lambda}_{\Phi_{Z_{G_i}},k} \geq \underline{\lambda}_{\Phi_{Z_{G_i}},k} > 0$. Making use of (8.18), we showed in [28] that, along (8.38),

$$V_{\tilde{\theta}_i}(k+1) \leq \beta_{i_5} V_{\tilde{\theta}_i}(k) - \beta_{i_1} \frac{\eta_i q_i^2(k+1)}{m_i^2(k)}, \tag{8.39}$$

where

$$0 < \beta_{i_5} = 1 - \eta_i \underline{\lambda}_{\Phi_{Z_{G_i}},k} \left(2 - \eta_i \left(\frac{2\overline{\lambda}_{\Phi_i,k}}{m_i^2(k)} + \overline{\lambda}_{\Phi_{Z_{G_i}},k}\right)\right) < 1 \tag{8.40}$$

if η_i not only satisfies (8.20) but is also picked such that

$$0 < \eta_i < \frac{2m_i^2(k)}{2\overline{\lambda}_{\Phi_i,k} + \overline{\lambda}_{\Phi_{Z_{G_i}},k}\, m_i^2(k)} = \overline{\eta}_{i_{CL}}, \tag{8.41}$$

expressed similarly to (4.83), but with $\varepsilon_{NG} = \varepsilon_{CL} = 1$. Notice that $\overline{n}_{i_{CL}} < \overline{n}_{i_{NG}}$. Therefore, picking η_i that guarantees (8.41) also guarantees (8.20). Next, considering the functions $V_{\tilde{\theta}_i}$ in (8.18), V_e in (8.21), and V_i in (8.24),

$$V_i(k) = V_e(k) + \beta_4 V_{\tilde{\theta}_i}(k) = \beta_{i_2} e^2(k) + \beta_{i_4} \left\|\tilde{\theta}_i(k)\right\|^2.$$

For all k, we can thus write

$$\min\left(\beta_{i_2}, \beta_{i_4}\right) \|\vartheta_i(k)\|^2 \leq V_i(k) \leq \max\left(\beta_{i_2}, \beta_{i_4}\right) \|\vartheta_i(k)\|^2, \tag{8.42}$$

where $\min(\cdot)$ and $\max(\cdot)$ respectively denote the minimum and maximum operators, and $\vartheta_i(k) = \left[e(k), \tilde{\theta}_i^\top(k)\right]^\top \in \mathbb{R}^{r_{\theta_i}+1}$. Since CL is only used for approximation purposes, then (8.23) remains unchanged here. Thus, reworking (8.23) to get

$$V_e(k+1) \leq \beta_{i_6} V_e(k) + \beta_{i_2}\left(1 + \frac{1}{\varepsilon_i}\right) q_i^2(k+1), \tag{8.43}$$

where $\beta_{i_6} = 1 - \beta_{i_3} = \gamma_e^2\left(1 + \varepsilon_i\right)$, and, using (8.39), (8.43), we have from (8.24) that

$$V_i(k+1) \leq \beta_{i_6} V_e(k) + \beta_{i_4}\beta_{i_5} V_{\tilde{\theta}_i}(k) - \frac{p_i(k) q_i^2(k+1)}{m_i^2(k)}, \tag{8.44}$$

where $p_i(k)$ is given by (8.26). Notice that $0 < |\gamma_e| < 1$ and ε_i satisfying (8.27) means $0 < \beta_{i_6} < 1$. We avoid having $\gamma_e = 0$ as a choice so that $\beta_{i_6} V_e(k) = 0$ only when $V_e(k) = 0$. Moving on, here also, by setting β_{i_4} such that (8.28) holds, $p_i(k) \geq 0$. If so, i.e., $p_i(k) \geq 0$, continuing along from (8.44),

$$V_i(k+1) \leq \beta_{i_6} V_e(k) + \beta_{i_4}\beta_{i_5} V_{\tilde{\theta}_i}(k) \leq \beta_{i_7} V_i(k), \tag{8.45}$$

where $\beta_{i_7} = \max\left(\beta_{i_5}, \beta_{i_6}\right)$. Picking η_i to satisfy (8.41) implies $0 < \beta_{i_5} < 1$. Coupling that with $0 < \beta_{i_6} < 1$, then $0 < \beta_{i_7} < 1$. As a result, (8.45) implies that $\lim\limits_{k\to+\infty} V_i(k) = 0$ exponentially, and, given (8.42), $\lim_{k\to+\infty} \|\vartheta_i(k)\| = 0$ exponentially as well. Hence, $e(k) \to 0$, and $\tilde{\theta}_i(k) \to [0]^{r_{\theta_i}}$ or $\hat{\theta}_i(k) \to \theta_i$ exponentially as $k \to \infty$, therefore completing the proof of **Theorem 8.2.1**. ■

Further, with $V_i(k)$ being positive definite and considering (8.45), $V_i(k) \leq \beta_{i_7} V_i(k_0)$ for all k. Hence, from (8.42) and denoting $\beta_{i_8} = \dfrac{\max(\beta_{i_2}, \beta_{i_4})}{\min(\beta_{i_2}, \beta_{i_4})} \geq 1$,

$$\|\vartheta_i(k)\|^2 \leq \beta_{i_7}\beta_{i_8} \|\vartheta_i(k_0)\|^2 .$$

For all k, given that $\|\vartheta_i(k)\|^2 = e^2(k) + \left\|\tilde{\theta}_i(k)\right\|^2$, then both $|e(k)| \leq b_{iCL}$ and $\left\|\tilde{\theta}_i(k)\right\| \leq b_{iCL}$, with

$$b_{iCL} \leq \sqrt{\beta_{i_7}\beta_{i_8}\left(e^2(k_0) + \left\|\tilde{\theta}_i(k_0)\right\|^2\right)}. \tag{8.46}$$

8.3 Reference Model

For $k > k_0$, a stable reference model system that tracks a desired bounded reference $r(k)$ can be given by

$$x_m(k+1) = -a_m x_m(k) + b_m r(k), \tag{8.47}$$

where $|a_m| < 1$ [3].

If $r(k) = r$ is constant (at least for a long period of time), it can be shown for $k > k_0$ that, given an initial state $x_m(k_0)$,

$$x_m(k) = \left[x_m(k_0) - \frac{b_m r}{1+a_m}\right](-a_m)^{k-k_0} + \frac{b_m r}{1+a_m}.$$

Hence, by picking $b_m = 1 + a_m$, as $k \to \infty$, $x_m(k) \to r$ since $|a_m| < 1$.

Let the reference be such that $|r(k)| \leq M_r$ for all k. If $b_m = 1 + a_m$ then, for all k, $|x_m(k)| \leq M_{x_m}$, where $M_{x_m} > 0$ can be chosen as $M_{x_m} = |x_m(k_0)| + 2M_r$.

8.4 Design Fine-Tuning

Let us say that we want x to be such that, for some $M_x > M_{x_m} > 0$, $|x(k)| \leq M_x$. From (8.3),

$$|x(k)| \leq |e(k)| + |x_m(k)|.$$

We can therefore find some $M_e > 0$ such that $|e(k)| \leq M_e$ means $|x(k)| \leq M_x$ for all k. Such an M_e can simply be $M_e = M_x - M_{x_m}$. Then, given (8.30) and (8.46), we will have to ensure $b^e_{iNG} \leq M_e$ and $b_{iCL} \leq M_e$. For both designs, given (8.3), notice that $e(k_0) = 0$ if $x_m(k_0) = x(k_0)$.

For the NG-based IAC design, if $e(k_0) = 0$, given (8.30) and (8.31), $b^e_{iNG} \leq M_e$ can be achieved if, for some $M_{\phi_i} > 0$ such that $\|\phi_i(\chi(k))\| \leq M_{\phi_i}$,

$$\left\|\tilde{\theta}_i(k_0)\right\| \leq 2\frac{M_e}{M_{\phi_i}}.$$

Given our definition of ϕ_i and that of u_i in (8.8), we can write

$$M_{\phi_i} = \sqrt{M^2_{\phi_f} + M^2_{\phi_g}\left(\frac{\gamma_e M_e + M_{\mathcal{F}} + M_{x_m}}{\underline{g}}\right)^2},$$

where for some $M_{\phi_f}, M_{\phi_g}, M_{\mathcal{F}} > 0$, $\|\phi_f(\chi(k))\| \leq M_{\phi_f}$, $\|\phi_g(\chi(k))\| \leq M_{\phi_g}$, and $|\mathcal{F}(k)| \leq M_{\mathcal{F}}$.

As for the CL-based IAC design, if $e(k_0) = 0$, given (8.46), $b_{iCL} \leq M_e$ if

$$\left\|\tilde{\theta}_i(k_0)\right\| \leq \frac{M_e}{\beta_{i_7}\beta_{i_8}}.$$

8.5 Numerical Simulations

Before describing our simulation example, given $\underline{g}$ defined in **Assumption 8.1.2**, we make sure that the approximation $\mathcal{G}$ of g, as given by (8.7), verifies $|\mathcal{G}(k)| \geq \underline{g}$ for all k. For that, we redefine

$$\mathcal{G}(k) = \begin{cases} \mathcal{G}(k) & \text{, if } |\mathcal{G}(k)| \geq \underline{g}, \\ \mu(\mathcal{G}(k))\,\underline{g} & \text{, if } |\mathcal{G}(k)| < \underline{g}, \end{cases}$$

where, for all $k > k_0$,

$$\mu\left(\mathcal{G}(k)\right) = \begin{cases} 1 & \text{, if } \mathcal{G}(k) > 0, \\ 1 & \text{, if } \mathcal{G}(k) = 0 \text{ and } \mathcal{G}(k-1) > 0, \\ -1 & \text{, if } \mathcal{G}(k) = 0 \text{ and } \mathcal{G}(k-1) < 0, \\ -1 & \text{, if } \mathcal{G}(k) < 0. \end{cases}$$

However, this necessitates that $\hat{\theta}_g(k_0)$ be initialized such that $\mathcal{G}(k_0) \neq 0$ since $\mathcal{G}(k_0 - 1)$ is not defined.

Consider the single state system in (8.1), where the structured uncertainties f and g, given by (8.2a) and (8.2b) respectively, are such that

$$\begin{aligned} &\theta_f = [-0.25,\, 10]^\top,\, \phi_f\left(\chi(k)\right) = \begin{bmatrix} 1, & \exp\left(-\frac{\left(x(k)-\frac{\pi}{2}\right)^2}{4}\right) \end{bmatrix}^\top, \\ &\theta_g = 5, \text{ and } \phi_g\left(\chi(k)\right) = 1. \end{aligned} \tag{8.48}$$

We desire to track train steps reference input of various heights, but for which $M_r = 10$. In the legends of the figures below, **NG** and **CL** are used to respectively indicate the use of the standard DTNG and DTNG based CL algorithms.

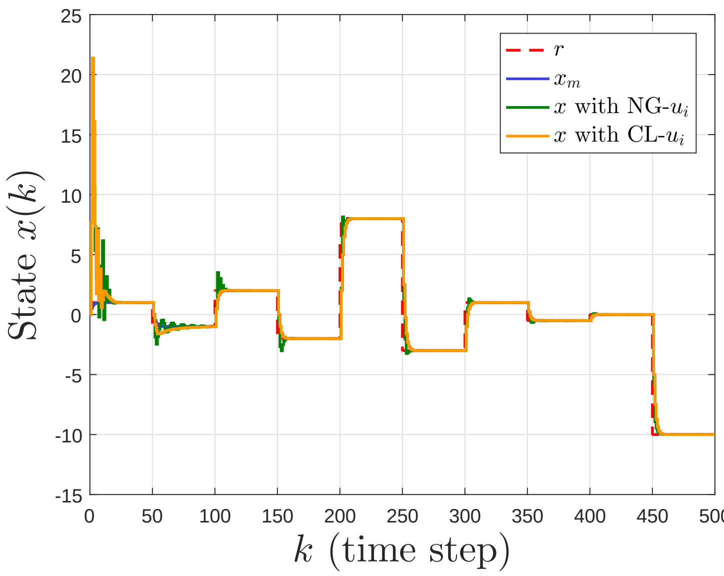
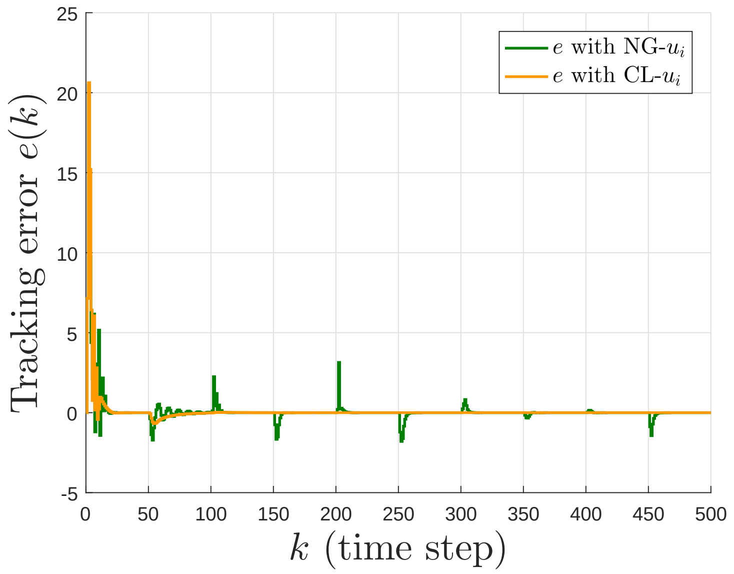
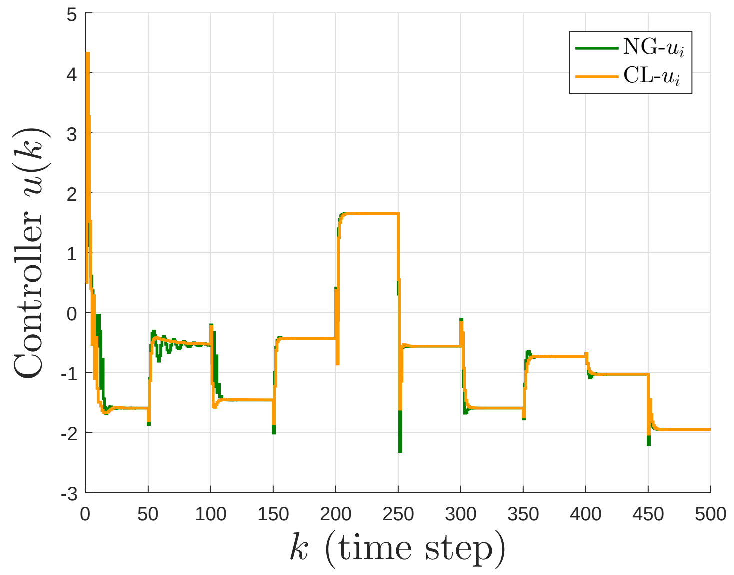


Figure 8.1: Evolution of the state x when implementing both IAC designs for tracking of the train steps.



(a) Tracking error $e(k)$.



(b) IAC scheme controller.

Figure 8.2: Evolution of the tracking error e and IAC scheme control action u_i when tracking the train steps.

For all simulations, we set $\alpha_i = 1$. We pick $\eta_i = 0.5\overline{\eta}_{i_{NG}}$ when using the NG algorithm and $\eta_i = 0.5\overline{\eta}_{i_{CL}}$ for the gradient based CL algorithm. Additionally, we set $c_{Z_i} = r_{\theta_i}$ and use DRP1 (described in Chapter VI) as the data recording procedure when implementing CL.

Let $k \in [k_0, k_f] \cap \mathbb{N}$, where k_f denotes a final simulation time-step. We set $k_0 = 0$ and $k_f = 500$ here. Furthermore, since $\phi_g(\chi(k)) = 1$ in our example, choosing $\hat{\theta}_g(k_0) = 1$ for instance ensures $\mathcal{G}(k_0) \neq 0$. The other parameter estimates are set to the origin. We assume knowing $\underline{g} = 1$. We start at $x(k_0) = x_m(k_0) = 0$ and set $\gamma_e = 0.5$, $a_m = -0.5$.

Figure 8.1 and Figure 8.2a compare the performance of the NG-based and CL-based IAC designs when tracking the train steps reference. As shown, tracking is achieved with both designs. Only when using CL however are we able to recover the true parameters in the plant dynamics f and g as Figure 8.3 and Figure 8.4 show. With the CL-based IAC design, judging by Figure 8.2a (or even Figure 8.1), tracking is significantly better during transients after convergence of the parameter estimates to their true values. Figure 8.2b shows the control signals.

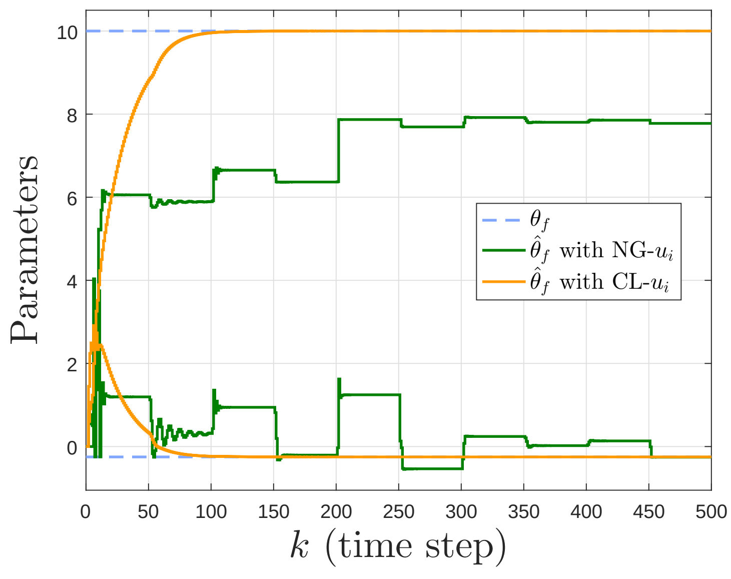


Figure 8.3: Evolution of the parameter estimates in $\mathcal{F}$ when implementing both IAC designs for tracking of the train steps.

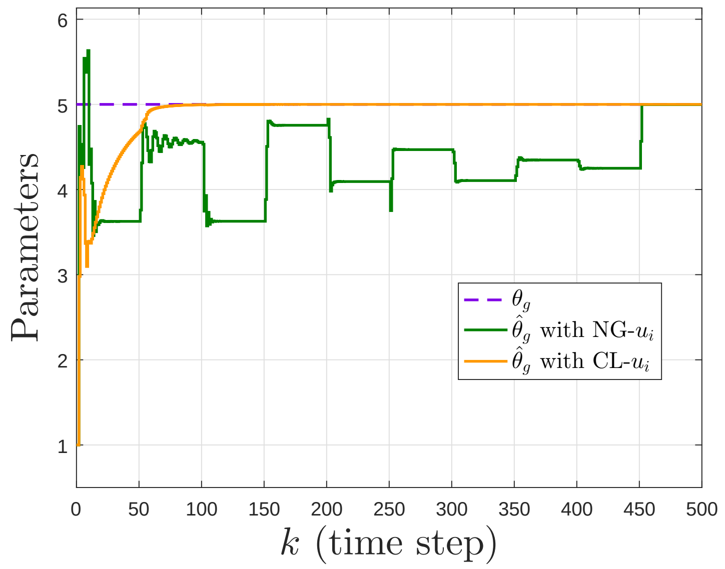


Figure 8.4: Evolution of the parameter estimates in $\mathcal{G}$ when implementing both IAC designs for tracking of the train steps.

8.6 Concluding Remarks about IAC Design

The present chapter studies a single state DT system with structured uncertainties. We develop an IAC scheme to cope with the inherent parametric uncertainties in the system. The behavior of closed-loop error signals when the designed IAC scheme utilizes either the DTNG algorithm or the DTNG based CL algorithm developed in Chapter IV for parameter adaptation is analyzed.

With the NG-based IAC design, we can ultimately drive the tracking error to zero but cannot formally prove parameter identification without requiring persistency of excitation.

We have previously showed that the gradient based CL algorithm for stand-alone approximation can help achieve parameter identification if a condition less restrictive than the persistency of excitation condition is met. CL is implemented by using in its adaptation law data saved in the form of past normalized regressor vectors inside a history stack. Provided that there are as many linearly independent normalized regressor vectors in the history stack as the number of dimensions in the same normalized regressors, we show exponential convergence of the tracking error and parameter error to zero when applying the CL-based IAC design. Numerical simulations are provided to corroborate our results.

CHAPTER IX

APPLICATION 2: FRICTION PARAMETER IDENTIFICATION OF COMAU RACER ROBOT USING LEAST SQUARES BASED CONCURRENT LEARNING

We perform system identification of the Comau Racer robot (described in Section 9.1.2) using the discrete-time Normalized Recursive Least Squares based algorithm developed in Chapter V.

We would like to add that the work described in this chapter has been done collaboratively with a team of researchers. Everything pertaining to modeling, design of experiments and off-line system identification has previously been done by the aforementioned research group. We have worked to use our developed CL method for on-line system identification.

Moreover, it is advised that the reader does not refer to previously used notations as some have been reused and redefined in the present chapter.

9.1 Background and Problem Formulation

9.1.1 Robot Dynamics

Throughout this chapter, we consider an r_q-link robotic manipulator, $r_q \in \mathbb{N}_+$, with $q = \left[q_1, q_2, \ldots, q_{r_q}\right]^\top \in \mathbb{R}^{r_q}$ denoting a vector of r_q positions in *joint* coordinate *space*. Several publications (see [33, 34, 35, 36, 37]) outline standard methods often made use of to describe the dynamics of a rigid body manipulator. Here, using Euler-Lagrange equations, the dynamics of an

r_q-link rigid robotic manipulator can be expressed as

$$D(q)\ddot{q} + C\left(q,\, \dot{q}\right)\dot{q} + g(q) + \tau_c + \tau_v = \tau, \tag{9.1}$$

where $\dot{q} \in \mathbb{R}^{r_q}$ and $\ddot{q} \in \mathbb{R}^{r_q}$ represent joint velocity and acceleration vectors respectively, $D(q) \in \mathbb{R}^{r_q \times r_q}$ is a positive definite inertia matrix, $C\left(q,\, \dot{q}\right)\dot{q} \in \mathbb{R}^{r_q}$ represents centrifugal and Coriolis forces, $g(q) \in \mathbb{R}^{r_q}$ is a vector of gravitational torques, and the torques $\tau_c \in \mathbb{R}^{r_q}$, $\tau_v \in \mathbb{R}^{r_q}$, and $\tau = \left[\tau_1,\, \tau_2,\, \ldots,\, \tau_{r_q}\right]^\top \in \mathbb{R}^{r_q}$ respectively represent a vector of Coulomb frictions, a vector of viscous frictions, and a vector of joint torques for control of the manipulator. To viaually illustrate a robot manipulator, on Figure 9.1, we show a sketch of a three degree of freedom (DoF) robot with joint positions q_i, $i = 1,\, 2,\, 3$, and lengths L_i of the manipulator's links.

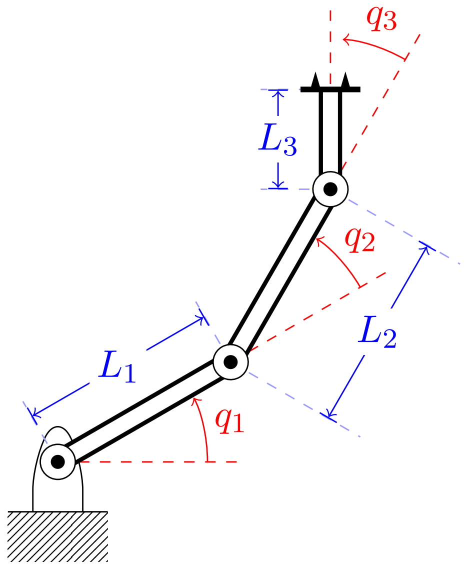


Figure 9.1: Sketch of a 3-DoF robot manipulator.

Consider the diagonal matrices $F_c \in \mathbb{R}^{r_q \times r_q}$ and $F_v \in \mathbb{R}^{r_q \times r_q}$, which we define as

$$F_c = \begin{bmatrix} f_{c,1} & 0 & \ldots & 0 \\ 0 & f_{c,2} & \ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & 0 & f_{c,r_q} \end{bmatrix} \text{ and } F_v = \begin{bmatrix} f_{v,1} & 0 & \ldots & 0 \\ 0 & f_{v,2} & \ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & 0 & f_{v,r_q} \end{bmatrix},$$

with scalars $f_{c,i}$ and $f_{v,i}$ respectively denoting Coulomb and viscous friction parameters for each of the joints $i \in \{1, 2, \ldots, r_q\}$. The Coulomb friction torque τ_c and viscous friction torque τ_v are modeled as

$$\tau_c = F_c \, \mathrm{sign}(\dot{q}), \tag{9.2}$$

where $\mathrm{sign}\,(\cdot)$ denotes the signum function and is such that, for some arbitrary vector $\bar{x} = [\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{r_{\bar{x}}}]^\top \in \mathbb{R}^{r_{\bar{x}}}$, $r_{\bar{x}} \in \mathbb{N}_+$,

$$\mathrm{sign}(x) = [\mathrm{sign}(\bar{x}_1),\ \mathrm{sign}(\bar{x}_2),\ \ldots,\ \mathrm{sign}(\bar{x}_{r_{\bar{x}}})]^\top \in \mathbb{R}^{r_{\bar{x}}},$$

and

$$\tau_v = F_v \dot{q}. \tag{9.3}$$

As [35, 36, 37] show, the dynamical equation (9.1) of the manipulator can also be expressed as

$$Y_b(\chi(t))\, p_b = \tau(t), \tag{9.4}$$

where, for $t \in \mathbb{R}$ representing continuous time or process time,

$$\chi(t) = [\ddot{q}(t), \dot{q}(t), q(t)]^\top \in \mathbb{R}^{r_\chi},$$

$r_\chi = 3r_q$, the constant vector $p_b \in \mathbb{R}^{r_{p_b}}$, $r_{p_b} \in \mathbb{N}_+$, contains r_{p_b} parameters that are functions of inertial and/or friction parameters, and the matrix $Y_b(\chi(t)) \in \mathbb{R}^{r_q \times r_{p_b}}$ depends on the geometry of the robot (which is assumed to be known). We note that the dynamic model (9.4), which is said to be linear in the parameter vector p_b or a Linear Parametric Model (LPM), certainly looks like a regression or function approximation model, with $Y_b(\chi)$ playing the role of the regression matrix while p_b consists of the unknown, but ideal parameters to be identified.

9.1.2 Comau Racer Robot

We experiment with an $r_q = 6$ rotational joints *Comau Racer* industrial robot. The Comau Racer is made of an anthropomorphic arm and a spherical wrist, where the rotation axes of the last three joints intersect at a single point.

The parameters in p_b are independent and fully define the dynamic model of the manipulator, i.e., (9.1) and/or (9.4). Hence, they form what is known as the base parameter set [34, 37, 38, 39]. Though p_b is not unique, its dimensions can be well defined. Given the method described in [38], a $r_{p_b} = 52$ base parameter set for the Comau Racer can be found.

9.1.3 Model Identification

Several schemes (see [35, 36, 40, 41, 42, 43, 44] and references within) for control of robotic manipulators can be found in literature. In particular, model-based control techniques make use of dynamical equations, which, in turn, depend on specific inertial and/or friction parameters. Therefore, good knowledge of these parameters is crucial when performing model-based control. In contrast, when interested in model-free control, knowing the parameters in p_b may not be as necessary, but would help design a good base controller, which could enhance the performance of the overall control scheme.

Though robot manufacturers often provide kinematic and some inertial parameters, in many instances, not all parameters are given. Friction parameters, for example, are usually not known. Computer-aided design (CAD) drawings can be used to determine many parameters. However, a physically built robot is usually slightly different from its CAD drawings after addition of the screws, cabling, and, possibly, other add-ons. Hence, researchers have been conducting parameter identification through experiments (or by running experiments and using experimentally collected data) in order to determine appropriate parameter values (see [34, 45, 46]).

Considering the LPM given by (9.4), the following off-line parameter identification procedure is usually employed.

1. Step 1: Defining excitation tracking trajectory for the robot that are suitable for identification;

2. Step 2: Running experiments and sampling data points;

3. Step 3: Signal filtering for calculation of velocity $\dot{q}$ and acceleration $\ddot{q}$ using joint positions q, which can then be used for computation of Y_b;

4. Step 4: Constructing an overdetermined linear system based on (9.4);

5. Step 5: Applying a system identification learning technique of choice for parameter identification.

First, as we have done previously, we define off-line parameter identification or off-line training as running a learning technique for identification purposes once it is deemed that a sufficient number of data points have been collected and the simulations of and/or experiments on the system have been completed. Now, say $N \in \mathbb{N}_+$ number data points are collected at times $t = t_0, t_1, \ldots, t_{N-1}$ with sampling time $T_s = t_i - t_{i-1}$, $i \in \{1, 2, \ldots, N-1\}$. The overdetermined system mentioned above in Step 4 is given by

$$Sp_b = b_\tau, \tag{9.5}$$

where the information matrix $S \in \mathbb{R}^{(Nr_q)\times r_{p_b}}$ and the vector $b_\tau \in \mathbb{R}^{Nr_q}$ are such that

$$S = \begin{bmatrix} Y_b(\chi(t_0)) \\ Y_b(\chi(t_1)) \\ \vdots \\ Y_b(\chi(t_{N-1})) \end{bmatrix} \text{ and } b_\tau = \begin{bmatrix} \tau(t_0) \\ \tau(t_1) \\ \vdots \\ \tau(t_{N-1}) \end{bmatrix},$$

with $\chi(t_i) = [\ddot{q}(t_i), \dot{q}(t_i), q(t_i)]^\top$ and $\tau(t_i)$, $i \in \{1, 2, \ldots, N\}$, assumed to be available at sample times t_i.

For collection of necessary data for off-line training using (9.5), as Step 1 stipulates, an excitation trajectory for robot tracking has to be defined first. The aforesaid excitation trajectory has to be not only practical, i.e., meet joint angles, velocity, and acceleration constraints, it also has to be rich enough to yield good system identification. Richness of the excitation trajectory, which is closely related to the concept of persistency of excitation (see Appendix B), mathematically translates into matrix S being at least full row rank if $Nr_q \leq r_{p_b}$ or full column rank $Nr_q \geq r_{p_b}$.

Several methods for defining excitation trajectories exist, some of which can be found in [47, 48, 49] and the references within those papers. The present research uses the method defined in [49], where desired joint positions (and, as a result, velocities and accelerations) to track are functions of harmonic sine and cosine waveforms, hence finite Fourier series. That is, the desired joints position vector $q^d = \left[q_1^d,\, q_2^d,\, \ldots,\, q_{r_q}^d\right]^\top \in \mathbb{R}^{r_q}$ to track is such that, for joints $i = 1, 2, \ldots, r_q$,

$$q_i^d(t) = q_{i,0}^d + \sum_{k=1}^{M} \left(a_{i,k} \sin(k\omega_f t) + b_{i,k} \cos(k\omega_f t)\right), \tag{9.6}$$

with $q_{i,0}^d$ being a constant (with respect to time) desired joint position offset, $a_{i,k}$ and $b_{i,k}$ representing Fourier coefficients, M being the number of harmonics to use, and ω_f being a fundamental frequency. Desired velocity $\dot{q}_i^d(t)$ and acceleration $\ddot{q}_i^d(t)$ can be analytically derived from (9.6). It is worth mentioning that a large M, i.e., a large number of harmonics, leads to higher accelerations, which could prove important for good identification. Frequency ω_f needs to be appropriately picked proportionally to the sampling frequency $\omega_s = \dfrac{2\pi}{T_s}$ while making sure not to excite the robot at its resonance frequency. Moving on, as [49] advocates, parameters $q_{i,0}^d$, $a_{i,k}$, and $b_{i,k}$ for $i = 1, 2, \ldots, r_q$ and $k = 1, 2, \ldots, M$, are found by maximizing the determinant of $S^\top S$ evaluated at $q_i(t_j) = q_i^d(t_j)$, $\dot{q}_i(t_j) = \dot{q}_i^d(t_j)$, and $\ddot{q}_i(t_j) = \ddot{q}_i^d(t_j)$, $i = 1, 2, \ldots, r_q$ and $j = 1, 2, \ldots N$, while making sure to take in consideration the robot joint and workspace constraints.

Position $q(t)$ and torque $\tau(t)$ vectors are measured at every sample time. Also, as Step 3 stipulates, velocity $\dot{q}(t)$ and acceleration $\ddot{q}(t)$ are obtained via signal filtering of $q(t)$. In that sense, letting $q_m(t) = \left[q_{m,1}, q_{m,2}, \ldots, q_{m,r_q}\right]^\top \in \mathbb{R}^{r_q}$, $\dot{q}_m(t) \in \mathbb{R}^{r_q}$, $\ddot{q}_m(t) \in \mathbb{R}^{r_q}$, and $\tau_m(t) = \left[\tau_{m,1}(t), \tau_{m,2}(t), \ldots, \tau_{m,r_q}(t)\right]^\top \in \mathbb{R}^{r_q}$ be the measured and/or filtered position, velocity, acceleration, and torque vectors respectively, it is not always the case that the previously listed measured and/or filtered entities are exactly equal to their true (analytic) selves. Actually, how good q_m, $\dot{q}_m$, $\ddot{q}_m$, and τ_m are in comparison to their correspondingly true q, $\dot{q}$, $\ddot{q}$, and τ has to do with the quality of the measuring devices and filtering process. Moreover, in essence, this also means that it is not possible (at least, not always) to compute $Y_b(\chi)$. Rather, $Y_b(\chi_m)$, where $\chi_m = [\ddot{q}_m, \dot{q}_m, q_m]^\top \in \mathbb{R}^{r_\chi}$, is what we would be computing. Let $S_m \in \mathbb{R}^{(Nr_q)\times r_{p_b}}$ and $b_{\tau_m} \in \mathbb{R}^{Nr_q}$ be such that

$$S_m = \begin{bmatrix} Y_b(\chi_m(t_0)) \\ Y_b(\chi_m(t_1)) \\ \vdots \\ Y_b(\chi_m(t_{N-1})) \end{bmatrix} \text{ and } b_{\tau_m} = \begin{bmatrix} \tau_m(t_0) \\ \tau_m(t_1) \\ \vdots \\ \tau_m(t_{N-1}) \end{bmatrix}.$$

Now, defining the errors $e_Y \in \mathbb{R}^{r_q}$, $e_\tau \in \mathbb{R}^{r_q}$, $e \in \mathbb{R}^{r_q}$, $e_S \in \mathbb{R}^{Nr_q}$, $e_b \in \mathbb{R}^{Nr_q}$, $\bar{e} \in \mathbb{R}^{Nr_q}$ as

$$\begin{aligned} e_Y(t) &= Y_b(\chi_m(t))\, p_b - Y_b(\chi(t))\, p_b, \\ e_\tau(t) &= \tau_m(t) - \tau(t), \\ e(t) &= e_\tau(t) - e_Y(t), \\ e_S(t) &= S_m p_b - S p_b, \\ e_b(t) &= b_{\tau_m} - b_\tau, \\ \bar{e}(t) &= e_b(t) - e_S(t), \end{aligned}$$

from (9.4) and (9.5) respectively, we get that

$$\tau_m(t) = Y_b(\chi_m(t))\, p_b + e(t) \tag{9.7}$$

and

$$b_{\tau_m} = S_m p_b + \bar{e}(t). \tag{9.8}$$

Using the Weighted Least Squares estimation (WLSE) method, for some positive definite covariance matrix $C \in \mathbb{R}^{(Nr_q)\times(Nr_q)}$, i.e., $C > 0$, while ignoring $\bar{e}(t)$ in (9.8) or, rather, assuming no error exists, i.e., $\bar{e}(t) = [0]^{Nr_q}$ and, therefore, $b_{\tau_m} = S_m p_b$, the estimate $\hat{p}_{b,WLSE} \in \mathbb{R}^{r_{p_b}}$ of p_b is computed as

$$\hat{p}_{b,WLSE} = \left(S_m^\top C^{-1} S_m\right)^{-1} S_m^\top C^{-1} b_{\tau_m}. \tag{9.9}$$

Expression (9.9) can also be described as a *weighted batch* Least Squares computation, where all existing recorded data for approximation is used for calculation of the parameter estimates. It should be noted that, with $C > 0$, $\left(S_m^\top C^{-1} S_m\right)^{-1}$ exists if $S_m^\top S_m$ is invertible, which, in turns, is possible to achieve granted the recorded data is rich enough. This is why the robot has to be commanded to track an excitation trajectory.

In light of all the points made previously and in the case of off-line training of the Comau Racer robot, first, given all joints and workspace restrictions, an optimal desired excitation trajectory is designed. Then, after running the robot through the trajectory and collecting enough data point, the WLSE method is used to solve for $\hat{p}_{b,WLSE}$ given in (9.9). The estimate $\hat{p}_{b,WLSE}$ can then be used to compute the predicted torque vector $\tau_p(t, \hat{p}_{b,WLSE})$ that is such that, for some estimation $\hat{p}_b \in \mathbb{R}^{r_{p_b}}$ of p_b, we define $\tau_p = \left[\tau_{p,1}, \tau_{p,2}, \ldots, \tau_{p,r_q}\right]^\top \in \mathbb{R}^{r_q}$ as

$$\tau_p(t, \hat{p}_b) = Y_b(\chi_m(t))\, \hat{p}_b. \tag{9.10}$$

Hence, using (9.10),

$$\tau_p(t, \hat{p}_{b,WLSE}) = Y_b(\chi_m(t))\, \hat{p}_{b,WLSE}.$$

Notice that τ_p in (9.10) is actually modeled after (9.4). From the experiments run on the Comau Racer robot, the prediction $\tau_p(t, \hat{p}_{b,WLSE})$ satisfactorily matched the measured torque $\tau_m(t)$.

9.1.4 Problem Formulation

In the present research, we focus on on-line approximation as opposed to its off-line counterpart. That is, the parameter identification algorithm is run as the robot is commanded to move. Identification needs not wait for collection of all data points to be implemented, as is the case for off-line training. Rather, with on-line approximation, the unknown parameters in p_b are being identified as each robot data point is being collected. On-line identification is of particular interest in the context of adaptive control, where approximation results (be it function approximation or identification of parameters) are made use of in the definition of the control law. We will use the off-line parameter estimation results as the "measuring stick" to which we will compare what we obtain when we perform on-line approximation instead.

9.2 Discrete-Time Representation

We consider model (9.7) as opposed to (9.4) for its more generic structure. Though (9.7) describes a CT system, because data points, collected and used for parameter estimation, are being sampled at times t_i, $i = 0, 1, \ldots, N-1$, therefore creating a process that is discrete in nature, we will use approximation schemes suited for the discrete-time framework.

We start by first redefining the approximation problem in the DT domain. As mentioned above, data is collected a times $t = t_0, t_1, \ldots, t_{N-1}$ with sampling time T_s between consecutive samples. Written differently, we have $t = t_0 + kT_s$, $k \in \{k_0, k_1, \ldots, k_{N-1}\} \in \mathbb{N}$, with $k_0 = 0$ and $k_i = k_{i-1} + 1$ for $i \in \{1, 2, \ldots, N-1\}$. Dropping T_s for conciseness, (9.7) translates to the DT model

$$\tau_m(k) = Y_b(\chi_m(k))\, p_b + e(k). \tag{9.11}$$

We have nevertheless essentially assumed that the system is being sampled fast enough that (9.11) is able to closely mimic (9.7).

9.3 Partial Approximation

It may be the case that certain parameters are well known or, at least, known well enough and, consequently, the complexity of the approximation problem could be reduced by only identifying unknown parameters, as opposed to all r_{p_b} parameters. To do so, the contribution of known parameters (if any) to τ_m (given by (9.11)) must be subtracted from τ_m and the approximation problem should be redefined.

Let $\tau_m^{kn} \in \mathbb{R}^{r_q}$ be the known portion of τ_m and $\tau_m^{ukn} \in \mathbb{R}^{r_q}$ be the unknown portion of τ_m. Known parameters can be made use of to reconstruct (to certain degree) τ_m^{kn} by extracting and using the known portions $p_b^{kn} \in \mathbb{R}^{r_{kn}}$, $r_{kn} \in \mathbb{N}$ and $r_{kn} < r_{p_b}$, of p_b and, correspondingly, $Y_b^{kn}(\chi_m) \in \mathbb{R}^{r_q \times r_{kn}}$ of $Y_b(\chi_m)$. While neglecting measurement errors that may have occurred, τ_m^{kn} can be calculated or, rather, approximated (alike τ_p) as

$$\tau_m^{kn}(t) \simeq Y_b^{kn}(\chi_m(k))\, p_b^{kn}. \tag{9.12}$$

Thus, if

$$\tau_m(k) = \tau_m^{kn}(k) + \tau_m^{ukn}(k),$$

for all k, using (9.12), τ_m^{ukn} can be numerically computed or, again, approximated (given that (9.12) is an approximation of $\tau_m^{kn}(k)$) as

$$\tau_m^{ukn}(k) = \tau_m(k) - \tau_m^{kn}(k) \simeq \tau_m(k) - Y_b^{kn}(\chi_m(k))\, p_b^{kn}. \tag{9.13}$$

We could then express τ_m^{ukn} much like τ_m in (9.11). That is, we could find unknown portions $p_b^{ukn} \in \mathbb{R}^{r_\theta}$, with $r_\theta = r_{p_b} - r_{kn}$, of p_b and, correspondingly, $Y_b^{ukn}(\chi_m) \in \mathbb{R}^{r_q \times r_\theta}$ of $Y_b(\chi_m)$ such that

$$\tau_m^{ukn}(k) = Y_b^{ukn}(\chi_m(k))\, p_b^{ukn} + e^{unk}(k), \tag{9.14}$$

where we define $e^{ukn}(k) \in \mathbb{R}^{r_q}$ as

$$e^{ukn}(k) = \tau_m(k) - Y_b^{kn}(\chi_m(k))\, p_b^{kn}.$$

In that sense, e^{ukn} is the error incurred by numerically computing τ_m^{ukn} using (**??**). Letting

$$\tau_m^{ukn} = \left[\begin{array}{cccc} \tau_{m,1}^{ukn}, & \tau_{m,2}^{ukn}, & \ldots, & \tau_{m,r_q}^{ukn} \end{array}\right]^\top,$$

$$Y_b^{ukn} = \left[\begin{array}{cccc} Y_{b,1}^{ukn\,\top}, & Y_{b,2}^{ukn\,\top}, & \ldots, & Y_{b,r_q}^{ukn\,\top} \end{array}\right]^\top,$$

and

$$e^{unk} = \left[\begin{array}{cccc} e_1^{unk}, & e_2^{unk}, & \ldots, & e_{r_q}^{unk} \end{array}\right]^\top,$$

where, for $i = 1, 2, \ldots, r_q$, $\tau_{m,i}^{ukn} \in \mathbb{R}$, $Y_{b,i}^{ukn} \in \mathbb{R}^{r_\theta}$, and $e_i^{unk} \in \mathbb{R}$ are, respectively, the row elements of τ_m^{ukn}, $Y_b^{ukn}(\chi_m)$, and e^{unk}, elementally, (9.14) means

$$\tau_{m,i}^{ukn}(k) = Y_{b,i}^{ukn}(\chi_m(k))\, p_b^{ukn} + e_i^{unk}(k). \tag{9.15}$$

9.4 On-Line Approximation and Parameter Identification

Analyses and derivations regarding function approximation and parameter identification will be done considering τ_m^{ukn} in (9.14), instead of τ_m. Notice however that, for $r_{kn} = 0$, $r_\theta = r_{p_b} - r_{kn} = r_{p_b}$, $\tau_m^{ukn} = \tau_m$, and we therefore return to the original problem of approximating τ_m. Also, any subsequent result we arrive at by studying function approximation while considering τ_m^{ukn} would also be applicable to function approximation when only considering any of the scalars $\tau_{m,i}^{ukn}$, $i = 1, 2, \ldots, r_q$, defined in (9.15) instead, given that $\tau_{m,i}^{ukn}$ is essentially a special case of τ_m^{ukn}.

Now, drawing parallels to Chapter V, while considering (9.14), let $f(k) \in \mathbb{R}^{r_q}$, $\phi(\chi_m(k)) \in$

$\mathbb{R}^{r_\theta \times r_q}$, $\theta \in \mathbb{R}^{r_\theta}$, and $e_f(k) \in \mathbb{R}^{r_q}$ given by

$$f(k) = \tau_m^{ukn}(k), \tag{9.16a}$$

$$\phi\left(\chi_m(k)\right) = Y_b^\top\left(\chi_m(k)\right), \tag{9.16b}$$

$$\theta = p_b^{ukn}, \tag{9.16c}$$

$$e_f(k) = e^{ukn}(k) \tag{9.16d}$$

be, respectively, the uncertainty to approximate, the regressor, the unknown but constant and ideal parameter vector, and the representation error due to approximation imperfections. It should be added that, though labeled as uncertain and as mentioned above, $f(k) = \tau_m^{ukn}$ is nevertheless computable via (9.13). Continuing, given the expressions in (9.16), in the newly adopted notations used for conciseness, (9.14) is rewritten as

$$f(k) = \phi^\top(\chi_m(k))\,\theta + e_f(k), \tag{9.17}$$

The LPM $\mathcal{F}\left(\chi_m(k), \theta\right) \in \mathbb{R}^{r_q}$, such that

$$\mathcal{F}\left(\chi_m(k), \theta\right) = \phi^\top(\chi_m(k))\,\theta, \tag{9.18}$$

approximates $\tau_m(k)$ for $\chi_m \in D_{\chi_m} \subset \mathbb{R}^{r_\chi}$ and $\theta \in D_\theta \subset \mathbb{R}^{r_\theta}$, with D_{χ_m} and D_θ being compact sets. It should be noted that (9.18) is in model (2.2b) form. Assuming that $e_f(k)$ is bounded, there exists a scalar $E \geq 0$ such that, for all k, $\chi_m \in D_{\chi_m}$, and $\theta \in D_\theta$,

$$\left\|e_f(k)\right\|_F \leq E. \tag{9.19}$$

This is a fair assumption to make given that the dynamical equation (9.1) can be written as (9.4), featuring the same regression matrix Y_b as in (9.11), (9.14), and (9.17), and measurements $\chi_m(k)$ are in practice supposed to be bounded. Furthermore, going back to (9.17), it is worthwhile noting that, we are, in essence, dealing here with an unstructured uncertainty approximation case as opposed to its structured uncertainty counterpart, given that, as (9.17) stipulates, it is not always the case that $\mathcal{F}\left(\chi(k), p_b\right)$ can be used to exactly and fully reconstruct $f(k)$ even if we are provided with the best θ that can be found on D_θ.

Chapter V (or reference [50]) investigates using the Least Squares method for function approximation and system identification. More specifically, it studies and compares the standard DT Normalized Recursive Least Squares (DTNRLS) algorithm and its CL modification. CL consists of selectively picking and recording past data so as to use it together with current data for parameter identification. Unlike standard learning techniques, CL does not require persistently exciting (PE) inputs but rather a more relaxed condition, involving the past, stored data being rich enough for results listed above to be achieved.

For on-line approximation purposes, we define $\hat{\theta}(k) \in \mathbb{R}^{r p_b}$ as the estimate of θ. Notice that $\hat{\theta}(k)$ is function of DT time k, as it is to be adjusted as k evolves via an adaptation law. In contrast, the ideal, true parameter vector θ is, as mentioned above, assumed to be constant. Also, consider the scheme $\mathcal{F}\left(\chi_m(k), \hat{\theta}(k)\right) \in \mathbb{R}^{r_q}$ modeled as

$$\mathcal{F}\left(\chi_m(k), \hat{\theta}(k)\right) = \phi^\top(\chi_m(k))\,\hat{\theta}(k) \tag{9.20}$$

after $\mathcal{F}\left(\chi_m(k), \theta\right)$ in (9.18). Let $\tilde{\theta}(k) \in \mathbb{R}^{r_\theta}$ and $\tilde{f}(k) \in \mathbb{R}^{r_q}$, given by

$$\tilde{\theta}(k) = \hat{\theta}(k) - \theta \tag{9.21}$$

and

$$\tilde{f}(k) = \mathcal{F}\left(\chi_m(k), \hat{\theta}(k)\right) - f(k), \tag{9.22}$$

be the parameter error and approximation error respectively. Notice that, numerically speaking, (9.22) gives us a way to compute $\tilde{f}$. Now, using (9.18), (9.20), and (9.21), $\tilde{f}(k)$ in (9.22) is also

$$\tilde{f}(k) = \mathcal{F}\left(\chi_m(k), \tilde{\theta}(k)\right) - \left[f(k) - \mathcal{F}\left(\chi_m(k), \theta\right)\right], \tag{9.23}$$

where, given the linearity in the parameter approximator of $\mathcal{F}$,

$$\begin{aligned}\mathcal{F}\left(\chi_m(k), \tilde{\theta}(k)\right) &= \mathcal{F}\left(\chi_m(k), \hat{\theta}(k)\right) - \mathcal{F}\left(\chi_m(k), \theta\right) \\ &= Y_b(\chi_m(k))\,\hat{\theta}(k) - Y_b(\chi_m(k))\,p_b \\ &= Y_b(\chi_m(k))\,\tilde{\theta}(k).\end{aligned}$$

From (9.17), (9.23) also means

$$\tilde{f}(k) = \mathcal{F}\left(\chi_m(k), \tilde{\theta}(k)\right) - e_f(k). \tag{9.24}$$

Consider the following definitions.

- Let $\zeta_i(\chi_m(k)) \in \mathbb{R}^{r_\theta}$, $i = 1, 2, \ldots, r_\theta$, represent the column vectors of $\phi\left(\chi_m(k)\right) \in \mathbb{R}^{r_\theta \times r_q}$. That is

$$\phi\left(\chi_m(k)\right) = \left[\begin{array}{cccc} \zeta_1(\chi_m(k)), & \zeta_2(\chi_m(k)), & \ldots, & \zeta_{r_{p_b}}(\chi_m(k)) \end{array}\right]. \tag{9.25}$$

- Let $Z \in \mathbb{R}^{r_Z \times c_Z}$, with $r_Z = r_\theta$ and $c_Z \in \mathbb{N}_+$, be a matrix that is filled column-wise with vectors $\zeta_i(\chi_m(\sigma_j))$, $i \in \{1, 2, \ldots, r_\theta\}$, obtained or found at discrete time σ_j, $k_0 \leq \sigma_j < k$, with $j = 1, 2, \ldots, c_Z$. Essentially,

$$Z = \left[\begin{array}{cccc} \zeta_i(\chi_m(\sigma_1)), & \zeta_i(\chi_m(\sigma_2)), & \ldots, & \zeta_i(\chi_m(\sigma_{c_Z})) \end{array}\right]. \tag{9.26}$$

  Moreover, for any k, say that the uncertainty $f(k) \in \mathbb{R}^{r_q}$ is such that

$$f(k) = \tau_m^{ukn}(k) = \left[\begin{array}{cccc} f_1(k), & f_2(k), & \ldots, & f_{r_q}(k) \end{array}\right]^\top,$$

  where the scalars $f_i(k)$, $i = 1, 2, \ldots, r_q$, are the entries of vector $f(k)$. Now, let $F \in \mathbb{R}^{c_Z}$ be a vector containing the uncertainty entities in $f(k)$. That is,

$$F = \left[\begin{array}{cccc} f_i(\sigma_1), & f_i(\sigma_2), & \ldots, & f_i(\sigma_{c_Z}) \end{array}\right]^\top, \tag{9.27}$$

for some $i \in \{1, 2, \cdots, r_q\}$, denoting the columns indexes of $f(\sigma_j)$, $j \in \{1, 2, \ldots, c_Z\}$. It should be noted that matrices Z and F could be time dependent, i.e., $Z = Z(k)$ and $F = F(k)$. Finally, we define $\Phi_{Z,k} \in \mathbb{R}^{r_Z \times r_Z}$ as

$$\Phi_{Z,k} = ZZ^\top. \tag{9.28}$$

Notice that $\Phi_{Z,k}$ is symmetric, i.e., $\Phi_{Z,k}^\top = \Phi_{Z,k}$, and (at least) positive semidefinite, i.e., $\Phi_{Z,k} \geq 0$.

Next, we describe and state the properties of the learning algorithms used here for on-line function approximation and parameter identification. As aforesaid, these learning algorithms are studied more in Chapter V.

9.4.1 Applying the Discrete-Time Normalized Recursive Least Squares Algorithm

Let $\hat{\theta}(k_0) \in \mathbb{R}^{r_\theta}$ be an initial vector parameter estimates. For $k \geq k_0$, we have from Chapter V the DTNRLS algorithm for updating the parameter estimate $\hat{\theta}$ as

$$\psi(k) = \phi\left(\chi_m(k)\right), \tag{9.29a}$$

$$Q_\epsilon(k) = \tilde{f}(k), \tag{9.29b}$$

$$K_{LS}(k) = P(k-1)\psi(k)\left(m^2(k)\right)^{-1}, \tag{9.29c}$$

$$\Delta\hat{\theta}_{LS}(k) = K_{LS}(k)Q_\epsilon(k), \tag{9.29d}$$

$$\hat{\theta}(k+1) = \hat{\theta}(k) - \Delta\hat{\theta}_{LS}(k), \tag{9.29e}$$

where $\phi\left(\chi_m(k)\right)$ is given by (9.16b), by letting $\psi(k) \in \mathbb{R}^{r_\psi \times c_\psi}$ and $Q_\epsilon(k) \in \mathbb{R}^{r_{Q_\epsilon}}$ then, for the present algorithm, $r_\psi = r_\theta$ and $c_\psi = r_q$ given (9.29a) while $r_{Q_\epsilon} = r_q$ given (9.29b), $K_{LS}(k) \in \mathbb{R}^{r_\psi \times c_\psi}$, $\Delta\hat{\theta}_{LS}(k) \in \mathbb{R}^{r_\theta}$, we define the gain matrix $P(k) \in \mathbb{R}^{r_\psi \times r_\psi}$ as

$$P(k) = P(k-1) - K_{LS}(k)\psi^\top(k)P(k-1), \tag{9.30}$$

with $P(k_0-1) = P_0 \in \mathbb{R}^{r_\psi \times r_\psi}$ chosen from the start as a symmetric, positive-definite matrix (i.e., $P_0^\top = P_0$ and $P_0 > 0$), and the normalization matrix signal $m(k) \in \mathbb{R}^{c_\psi \times c_\psi}$ is such that, for some scalar $\alpha > 0$,

$$m^2(k) = \alpha I_{c_\psi} + \psi^\top(k) P(k-1) \psi(k). \tag{9.31}$$

Referring to Chapter Chapter V, when dealing with an unstructured uncertainty, it can be formally shown that the DTNRLS algorithm in (9.29) guarantees that:

- The parameter error $\tilde{\theta}(k)$ is uniformly ultimately bounded (UUB) or, said differently, $\tilde{\theta}(k)$ (and, thus, $\hat{\theta}(k)$) ultimately ends up in a neighborhood around the origin (in a neighborhood around θ) provided $e_f(k) \in \mathcal{L}^2$ (see Appendix C for the definitions of $\mathcal{L}^p$, $p \in [1, \infty)$, and $\mathcal{L}^\infty$ signal spaces);

- For all $k > k_0$, $P(k) = P^\top(k)$ is positive definite and bounded. There also exists a constant matrix $P_{ss} \in \mathbb{R}^{r_\psi \times c_\psi}$ such that $\lim\limits_{k \to \infty} P(k) = P_{ss}$;

- For some constant matrix $\hat{\theta}_{ss} \in \mathbb{R}^{r p_b}$, $\lim\limits_{k \to \infty} \hat{\theta}(k) = \hat{\theta}_{ss}$ if $e_f(k) \in \mathcal{L}^p$, $p \in [1, \infty)$.

Theoretically speaking, stability of $\tilde{\theta}(k)$ could also be shown if we are guaranteed a PE $\psi_m(k) = \psi(k) m^{-1}(k) \in \mathbb{R}^{r_\psi \times c_\psi}$ [3, 2, 4], and that is without any other requirement on e_f. Nevertheless, requiring PE condition and/or $\mathcal{L}^p$, $p \in [1, \infty)$, properties of e_f is quite demanding. Concurrent Learning, which we employ and apply next, requires a milder condition than both previously mentioned conditions for stability guarantees.

9.4.2 Applying the Discrete-Time Normalized Recursive Least Squares Based Concurrent Learning Algorithm

CL makes use of memory bank, dubbed as *history stack*, where carefully chosen past data is saved. That history stack is what we have denoted here as Z above in (9.26).

Consider the following notations and definitions. Let $\xi_\phi \in \mathbb{R}^{r_q \times r_q}$ and $\xi_Z \in \mathbb{R}^{c_Z \times c_Z}$ be positive definite matrices, i.e., $\xi_\phi > 0$ and $\xi_Z > 0$. Similarly to $\mathcal{F}$, consider the approximation based on recorded data $\bar{\mathcal{F}}(Z, \theta) \in \mathbb{R}^{c_Z}$ and its corresponding on-line scheme $\bar{\mathcal{F}}\left(Z, \hat{\theta}(k)\right) \in \mathbb{R}^{c_Z}$ given by

$$\bar{\mathcal{F}}(Z, \theta) = Z^\top \theta \tag{9.32}$$

and

$$\bar{\mathcal{F}}\left(Z, \hat{\theta}(k)\right) = Z^\top \hat{\theta}(k), \tag{9.33}$$

respectively. Much like $\tilde{f}$, we compute the approximation error based on recorded data $\tilde{f}_Z(k) \in \mathbb{R}^{c_Z}$ as

$$\tilde{f}_Z(k) = \bar{\mathcal{F}}\left(Z, \hat{\theta}(k)\right) - F, \tag{9.34}$$

with F given by (9.27). This time around, let $\psi(k) \in \mathbb{R}^{r_\psi \times c_\psi}$ be defined as

$$\psi(k) = \left[\phi\left(\chi_m(k)\right) \xi_\phi, \, Z \, \xi_Z\right]. \tag{9.35}$$

That is $r_\psi = r_\theta$ and $c_\psi = r_q + c_Z$. We also define the row vector $\bar{F}(k) \in \mathbb{R}^{1 \times (r_q + c_Z)}$ as

$$\bar{F}(k) = \left[f^\top(k) \, \xi_\phi, \, F^\top \xi_Z\right]. \tag{9.36}$$

In light of the expressions of $\psi(k)$ in (9.35) and $\bar{F}(k)$ in (9.36), let the overall approximation error $Q_\epsilon(k) \in \mathbb{R}^{r_{Q_\epsilon}}$ be given by

$$Q_\epsilon(k) = \psi^\top(k) \hat{\theta}(k) - \bar{F}^\top(k), \tag{9.37}$$

which consequently means that $r_{Q_\epsilon} = r_q + c_Z$ in this case. Notice that $\tilde{f}(k)$ in (9.22), $\tilde{f}_Z(k)$ in (9.34), along with expressions (9.20), (9.33), (9.35), and (9.36) make it possible to re-express (9.37) as

$$Q_\epsilon(k) = \xi_\phi^\top \tilde{f}(k) + \xi_Z^\top \tilde{f}_Z(k). \tag{9.38}$$

As (9.38) shows, matrices ξ_ϕ and ξ_Z are weighting matrices that could be used to emphasize either the contribution of the approximation error based on current data, calculated as $\tilde{f}(k)$, or the contribution of the approximation error based on recorded, past data, calculated as $\tilde{f}_Z(k)$.

The DTNRLS based CL algorithm, which we described below in (9.39), very much resembles the standard DTNRLS algorithm in (9.29). For $k \geq k_0$, for some positive definite matrices $\xi_\phi \in \mathbb{R}^{r_q \times r_q}$ and $\xi_Z \in \mathbb{R}^{c_Z \times c_Z}$, given an initial parameter estimate $\hat{\theta}(k_0) \in \mathbb{R}^{r_\theta}$, and for a chosen positive definite gain matrix $P(k_0 - 1) = P_0$, with $P_0^\top = P_0$, updating the parameter estimate θ is done by recursively carrying out

$$\begin{aligned} \psi(k) &= \left[\phi\left(\chi_m(k)\right)\xi_\phi, Z\,\xi_Z\right], \\ Q_\epsilon(k) &= \xi_\phi^\top \tilde{f}(k) + \xi_Z^\top \tilde{f}_Z(k), \\ K_{CL}(k) &= P(k-1)\psi(k)\left(m^2(k)\right)^{-1}, \\ \Delta\hat{\theta}_{CL}(k) &= K_{CL}(k) Q_\epsilon(k), \\ \hat{\theta}(k+1) &= \hat{\theta}(k) - \Delta\hat{\theta}_{CL}(k), \end{aligned} \tag{9.39}$$

with $P(k)$ and $m^2(k)$ respectively given by (9.30) and (9.31). Comparing the DTNRLS algorithm in (9.29) and its CL modification in (9.39) reveals that the actual calculations of $\psi(k)$ and $Q_\epsilon(k)$ is where their differences lie. Otherwise, form-wise at least, both algorithms are very similar.

CL needs not PE inputs for realization of convergence of the parameter error $\tilde{\theta}$. Nevertheless, it requires linear independence of the data in the history stack Z. This condition, also known as the rank condition, is given by **Condition 3.1.1**.

Condition 3.1.1 means $c_Z \geq r_Z$ and, denoting rank$(\cdot)$ as the rank operator, rank$(Z) = r_Z = r_\theta$. It is less demanding than persistency of excitation as it only deals with a subgroup of past data (in the history stack). Additionally, unlike the PE condition, it is possible to implement and supervise the rank condition on-line.

If **Condition 3.1.1** is verified, we have from **Theorem** that, for the unstructured uncertainty case, the DTNRLS based CL algorithm guarantees that:

- The parameter error $\tilde{\theta}(k) \rightarrow [0]^{r_{p_b}}$ asymptotically, or, equivalently, $\hat{\theta}(k) \rightarrow p_b$ asymptotically as times evolves;

- Here also, $P(k) = P^{\top}(k)$ is positive definite and bounded for all $k > k_0$. Moreover, $\lim\limits_{k\rightarrow\infty} P(k) = P_{ss} = [0]^{r_\psi \times r_\psi}$ (which actually allows to show convergence of $\tilde{\theta}$ to the origin).

Remark 5.2.4 in Chapter V emphasizes that necessitating $\mathcal{L}^2$ and/or $\mathcal{L}^p$, $p \in [1, \infty)$, properties for e_f to realize some of the theoretical guarantees of the standard DTNRLS algorithm could be, just like the PE condition, more restrictive than requiring that the rank condition on Z be met in the case of the CL modification.

We have mentioned throughout this work that the data that goes into the history stack Z (or the data that makes up Z) undergoes a selective process. Chapter VI provides data recording procedures for constructing Z. Two procedures are presented in [50] therein, namely the Data Recording Procedure 1 (DRP1) and the Data Recording Procedure 2 (DPR2). Both procedures seek to maximize the minimum eigenvalue of $\Phi_{Z,k}$ (defined by (9.28)) given that, as shown in [50], such a maximization would speed up the convergence of $\tilde{\theta}(k)$ towards the origin. However, while DRP1 calls for application of the CL modification starting at $k = k_0$, i.e., actively looking to start mixing current and past data from the on-start, with DRP2, CL is only applied after sufficient data guaranteeing verification of the rank condition has been collected. Both procedures have their advantages as well as disadvantages, as [50] points.

9.5 Numerical Simulations and Experimental Results

We now put both the standard DTNRLS algorithm and its CL modification to task on the $r_q = 6$ rotational joints Comau Racer industrial robot. Both algorithms are used for on-line parameter identification.

We are however only interested in identifying the 12 friction parameters associated with robot, which have been labeled parameters 41 through 52 (out of the 52 parameters in p_b). It is not too far-fetched to assume that kinematic and inertial parameters are known well enough based on the information provided by robot manufacturers and/or CAD drawings. Friction parameters, as we have pointed to before, are however not necessarily (readily) known. Identifying them on-line, as we have set out to do, could, for instance, help better a friction torque compensation scheme when controlling the robot.

Numerical simulations of Section 9.5.1 are obtained by simulating a model of the robot in Matlab/Simulink. Real-life experiments are also carried out and reported in Section 9.5.2.

For implementation of the algorithms, we start with

$$P(k_0 - 1) = P_0 = 100\, I_{r_\psi} \text{ and } \hat{\theta}(k_0) = [0]^{r_\theta},$$

while setting $\alpha = 1$, unless otherwise explicitly stated. For CL only, DRP1 is chosen as the data recording procedure, we set $c_Z = r_Z = r_\theta$. Lastly, on the figures shown in Sections 9.5.1 and 9.5.2, legends **'LS'** and **'LS-CL'** are used to denote the DTNRLS and the DTNRLS-based CL algorithms respectively.

9.5.1 Numerical Simulations

Let $\hat{\theta}_{WLSE} \in \mathbb{R}^{r_\theta}$ be the unknown portion of the estimate $\hat{p}_{b,WLSE}$ found via off-line training using the WLSE method. We define and compute metric

$$\varepsilon_{\hat{\theta}}(k) = \left\| \hat{\theta}(k) - \hat{\theta}_{WLSE} \right\| \tag{9.40}$$

as a measure of the difference (in norm) between the estimates obtained on-line and off-line as time evolves.

Seen on Figure 9.2 is the evolution of the parameter estimates, the evolution of metric $\varepsilon_{\hat{\theta}}$, and the final values of the parameter estimates after 20 seconds of simulation. It should be added that, for CL implementation, we set $\xi_\phi = I_{r_q}$, and $\xi_Z = I_{c_Z}$. What Figure 9.2 shows is that, as far as recovering $\hat{\theta}_{WLSE}$, though both algorithms perform well, the CL modification seemingly does better than the standard DTNRLS algorithm.

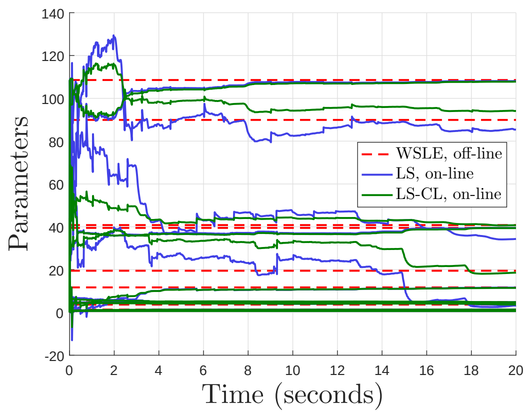
9.5.2 Experimental Results

Here, parameter identification is performed on the actual Comau Racer robot in real-time. We have set $\xi_\phi = I_{r_q}$ and $\xi_Z = I_{c_Z}$ for use of the DTNRLS based CL technique. From the data collected, as Figure 9.3 shows, we are able to plot the evolution of the parameters estimates and their final (steady-state) values. Notice on Figure 9.3a that we experience large transients between $t = 7$ seconds and $t = 10$ seconds marks when using the DTNRLS based CL method. This could be due to setting the minimum (and, thus, maximum) eigenvalue of P_0 to a large number. Hence, to see with better clarity what happens in steady state, we have added Figure 9.3b as a zoomed-in version of Figure 9.3a.

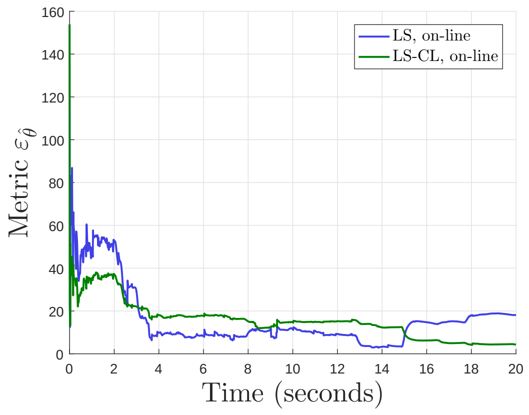
Additionally, we set $P_0 = 0.001\, I_{r_\psi}$ (hence, with much smaller spectral properties than what they were for the previous run) for use of both LS methods and repeat the real-time parameter identification experiment. It should also be said that we set $\xi_\phi = 50\, I_{r_q}$ and $\xi_Z = 0.1\, I_{c_Z}$ this time around. Our results are shown Figure 9.4. Unlike before, we do not experinece large transients in the evolution of the DTNRLS based CL parameter estimates.

Notice on both Figures 9.3c and 9.4b that, unlike in Section 9.5.1, there are more discrepancies between the WLSE off-line estimates and both the final values of the DTRNLS and DTNRLS based CL on-line estimates. We have up to now considered the WLSE estimates as the "golden standard," with respect to which, in Section 9.5.1, we have computed metric $\varepsilon_{\hat{\theta}}$. However, an ultimately better measure of how good the parameters estimates are would be to compute the predicted torque τ_p defined in (9.10).

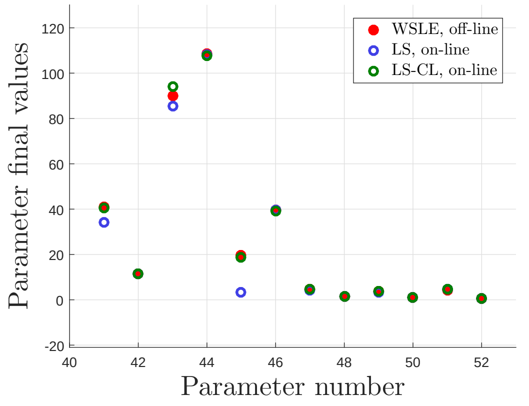
Let $\hat{p}_{b,LS}$ and $\hat{p}_{b,CL}$ be the final values of the DTNRLS and DTNRLS based CL parameter



(a) *Evolution of parameter estimates.*



(b) *Evolution of* $\|\varepsilon_{p_b}\|$.



(c) *Parameter estimate final values.*

Figure 9.2: Numerical simulations: parameter identification results of the Comau Racer robot using the DTNRLS algorithm and the DTNRLS based CL (with DRP1, $\xi_\phi = I_{r_q}$, *and* $\xi_Z = I_{c_Z}$*) algorithm.*

estimates of p_b. That means $\hat{p}_{b,LS}$ and $\hat{p}_{b,CL}$ contains the known parameters and the correspondingly final values of the estimated unknown friction parameters using either methods. For some estimate $\hat{p}_b \in \mathbb{R}^{r_p}$, let

$$\tau_p(k, \hat{p}_b) = Y_b(\chi_m(k))\, \hat{p}_b, \tag{9.41}$$

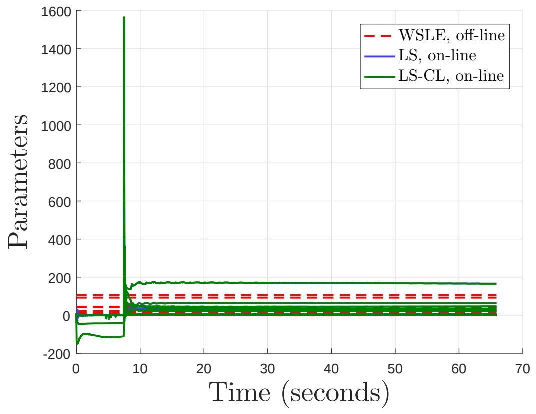
be the DT version of CT framework $\tau_p(t, \hat{p}_b)$ in (9.10). Aside from $\tau_p(k, \hat{p}_{b,WLSE})$, we can thus also compute the predicted vector torques $\tau_p(k, \hat{p}_{b,LS})$ and $\tau_p(k, \hat{p}_{b,CL})$ using (9.41). For illustration purposes, on Figure 9.5, for when $P_0 = 100\, I_{r_\psi}$, we plot measurements $\tau_{m,i}$, $i = 1, 2, \ldots, 6$, and their predictions $\tau_{p,i}(k, \hat{p}_{b,WLSE})$, $\tau_{p,i}(k, \hat{p}_{b,LS})$, and $\tau_{p,i}(k, \hat{p}_{b,CL})$. Moreover, from a numerical point of view, we compare measured and predicted torques. In fact, for N being the number of sample points collected and $i = 1, 2, \ldots, 6$, we compute metrics

$$\varepsilon_{\tau_i}(\hat{p}_b) = \frac{1}{N} \sum_{\sigma=k_0}^{k_{N-1}} \left|\tau_{p,i}(\sigma, \hat{p}_b) - \tau_{m,i}(\sigma)\right|^2, \tag{9.42}$$

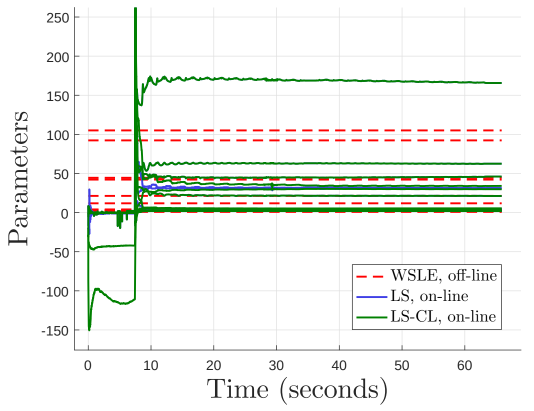
which, essentially, are the mean square error (MSE) between the predicted and measured torques for each of the 6 joints of the Comau Racer robot, and tabulate them in Table 9.1 for when $P_0 = 100\, I_{r_\psi}$ and Table 9.2 for when $P_0 = 0.001\, I_{r_\psi}$. All units are in SI. We see that the obtained on-line parameter estimates provide better torque predictions than the WLSE estimates. Moreover, as previously remarked in Chapter VII, when the spectral properties of P_0 are large, as Table 9.1 shows, from an application standpoint, it is difficult to clearly point to the benefit (aside from the theoretical guarantees) of using the DTNRLS based CL algorithm. Instead, when choosing P_0 with smaller spectral properties, we can now, as Table 9.2 reveals, say that the DTNRLS based CL method improves the standard DTNRLS algorithm.

9.6 Concluding Remarks

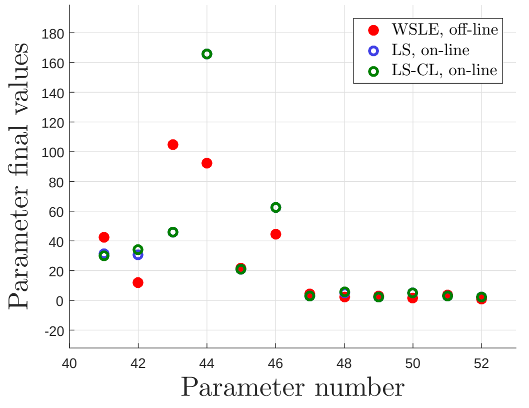
We use the Normalized Recursive Least Squares algorithms of Chapter V for system identification of the Comau Racer robot. In particular, we are interested in estimating friction parameters.



(a) Evolution of parameter estimates (zoomed out).

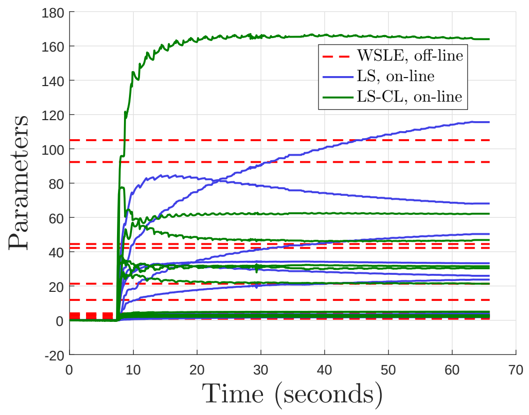


(b) Evolution of parameter estimates (zoomed in).

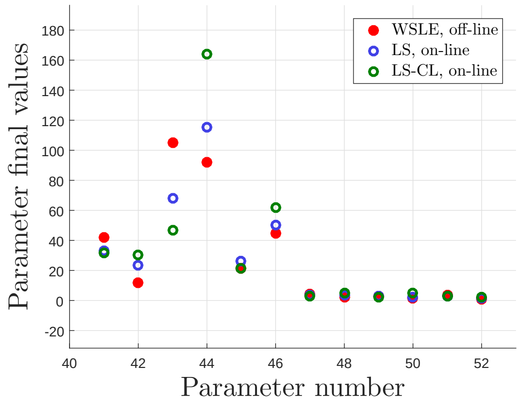


(c) Parameter estimate final values.

Figure 9.3: Experimental results: parameter identification results of the Comau Racer robot using the DTNRLS algorithm and the DTNRLS based CL (with DRP1, $\xi_\phi = I_{r_q}$, and $\xi_Z = I_{c_Z}$) algorithm: $P_0 = 100\, I_{r_\psi}$.



(a) Evolution of parameter estimates (zoomed out).

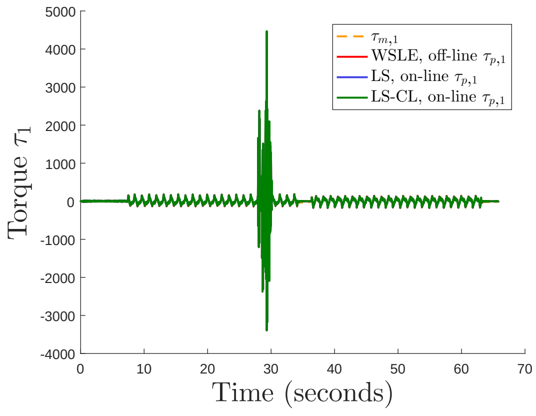


(b) Parameter estimate final values.

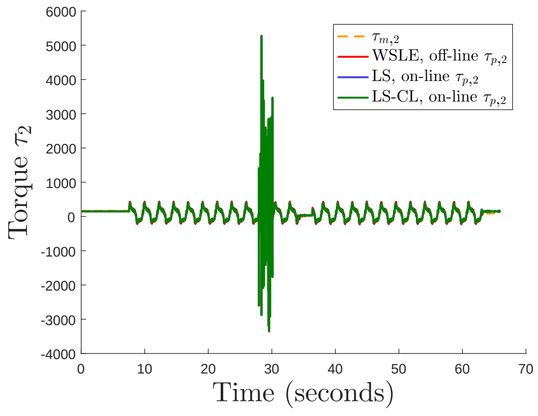
Figure 9.4: Experimental results: parameter identification results of the Comau Racer robot using the DTNRLS algorithm and the DTNRLS based CL (with DRP1, $\xi_\phi = 50\, I_{r_q}$, and $\xi_Z = 0.1\, I_{c_Z}$) algorithm: $P_0 = 0.001\, I_{r_\psi}$.

Table 9.1: Mean square torque prediction errors: $P_0 = 100\, I_{r_\psi}$, DTNRLS based CL is implemented with DRP1, $\xi_\phi = I_{r_q}$, and $\xi_Z = I_{c_Z}$.

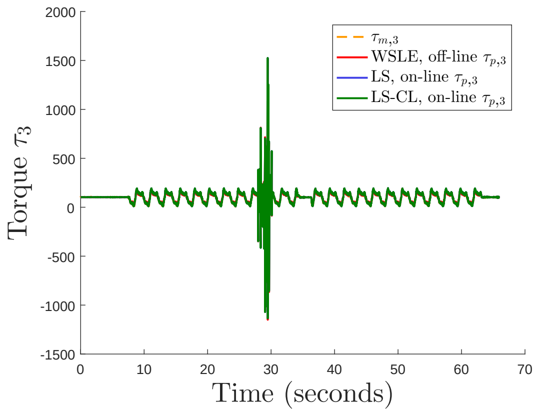
	WLSE	DTNRLS	DTNRLS based CL
$\varepsilon_{\tau_1} \times 1000$	432.24	**431.84**	431.85
$\varepsilon_{\tau_2} \times 1000$	533.56	**524.43**	**524.43**
$\varepsilon_{\tau_3} \times 1000$	157.43	**154.41**	**154.41**
$\varepsilon_{\tau_4} \times 1000$	6.65	**6.27**	6.28
$\varepsilon_{\tau_5} \times 1000$	5.38	**5.05**	**5.05**
$\varepsilon_{\tau_6} \times 1000$	3.35	**3.14**	**3.14**



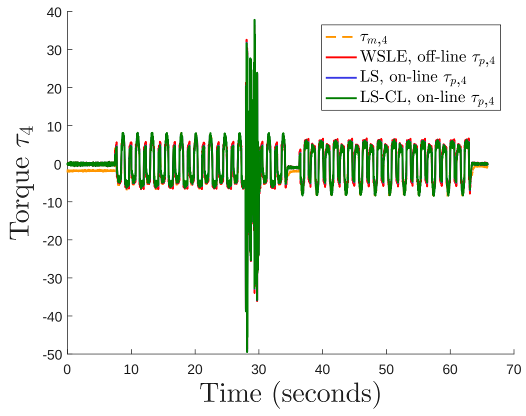
(a) $\tau_{m,1}$ and its predictions.



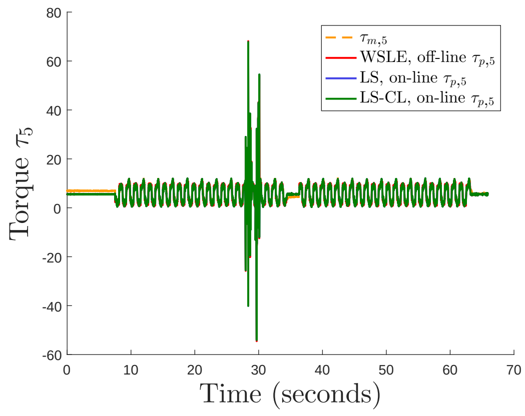
(b) $\tau_{m,2}$ and its predictions.



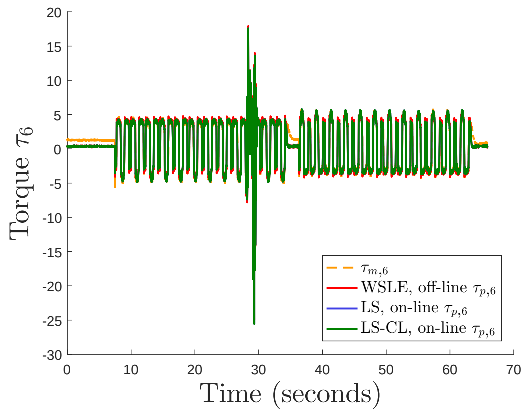
(c) $\tau_{m,3}$ and its predictions.



(d) $\tau_{m,4}$ and its predictions.



(e) $\tau_{m,5}$ and its predictions.



(f) $\tau_{m,6}$ and its predictions.

Figure 9.5: Experimental results: measured torques $\tau_{m,i}$, $i = 1, 2, \ldots, 6$, and predicted torques $\tau_{p,i}$ when performing parameter identification of the Comau Racer robot using the DTNRLS algorithm and the DTNRLS based CL (with DRP1, $\xi_\phi = I_{r_q}$, and $\xi_Z = I_{c_Z}$) algorithm: $P_0 = 100\, I_{r_\psi}$.

Table 9.2: Mean square torque prediction errors: $P_0 = 0.001\, I_{r_\psi}$, DTNRLS based CL is implemented with DRP1, $\xi_\phi = 50\, I_{r_q}$, and $\xi_Z = 0.1\, I_{c_Z}$.

	WLSE	DTNRLS	DTNRLS based CL
$\varepsilon_{\mathcal{T}_1} \times 1000$	432.24	431.87	$\mathbf{431.84}$
$\varepsilon_{\mathcal{T}_2} \times 1000$	533.56	525.37	$\mathbf{524.43}$
$\varepsilon_{\mathcal{T}_3} \times 1000$	157.43	154.78	$\mathbf{154.41}$
$\varepsilon_{\mathcal{T}_4} \times 1000$	6.65	6.34	$\mathbf{6.27}$
$\varepsilon_{\mathcal{T}_5} \times 1000$	5.38	5.17	$\mathbf{5.05}$
$\varepsilon_{\mathcal{T}_6} \times 1000$	3.35	3.15	$\mathbf{3.14}$

First, a Weighted Least Squares Estimation method is used off-line for system identification of the robot. It should be added that experiments with and results obtained from using the Weighted Least Squares method are the work of a team of researchers we have collaborated with.

We have for our part performed on-line parameter identification using both the standard discrete-time Normalized Recursive Least Squares algorithm and its Concurrent Learning modification. Considering the parameter estimated off-line via Weighted Least Squares Estimation as a reference to compare our results to, Figure 9.2 shows that our discrete-time Normalized Recursive Least Squares based Concurrent Learning algorithm is able to perform better than the standard algorithm when it comes to recovering the parameters obtained off-line. Experiments are then performed on the actual robot and as we see on Figure 9.5, there is quite a bit of mismatch between the on-line estimated parameters and their off-line counterparts. For better comparison, we then go on to compute, given each set of estimated parameters, the mean square errors between the measured torques and the predicted torques, the latter obtained using the aforementioned set of estimated parameters. Tables 9.1 and 9.2, both of which contain the mean square torque prediction errors for each dimension of the torque vector and for each estimation method used, show that the on-line methods, i.e., the standard discrete-time Normalized Recursive Least Squares and the discrete-time Normalized Recursive Least Squares based Concurrent Learning methods, generate less error. There however isn't a clear and distinct separation between the standard Least Squares algorithm and its Concurrent Learning modification given the results in Table 9.1. Hence, one could again be left wondering if there are any benefits to using the Concurrent Learning technique. However, as remarked in Chapter V, the standard algorithm can be thought of as a memory based method, therefore with the same underlining purposes as Concurrent Learning. From the experimental results of Table 9.1, it would then seem that it is not always the case that adding more memory can help the CL modification perform better. In Chapter V, we have seen that, when setting the minimum (and, thus, maximum) eigenvalue of P_0 (initial guess of the gain matrix) to a large value, it becomes difficult to distinctively tell how good the CL modification is compared to the standard algorithm. Choosing P_0 with small spectral properties, we can then clearly see, as Table 9.1 shows, the advantage(s) of using the Concurrent Learning modification of the discrete-time Normalized Recursive Least Squares from an application point of view. In any case, all theoretical properties guaranteed by using the discrete-time Normalized Recursive Least Squares Concurrent Learning algorithm (which the standard algorithm cannot or does not always provide) have been validated.

CHAPTER X

CONCLUSION AND FUTURE RESEARCH

Commonly used in science and engineering for various tasks such as function approximation, learning, and signal processing (among other), system identification is the process of using input and output data to build models of unknown or uncertain systems [1]. When approximating a function, one usually either deals with the structured uncertainty approximation case or the unstructured uncertainty approximation case. Regardless of the type of approximation encountered, putting together a function approximation scheme involves defining a regressor, made of computable signals, and a set of parameter estimates, updated using an adaptation rule in order to arrive at a better representation of some unknown, true parameters that would yield the best approximation on some predefined approximation set.

When used for standalone approximation, standard learning methods (typically, Gradient Descent and Recursive Least Squares), do help minimize the approximation error. However, convergence of parameter estimates to their true values or a neighborhood around the aforesaid true values when employing standard learning methods can only be theoretically proved if the regressor used for approximation is persistently exciting (PE) [2, 3, 4]. In Appendix B, we formally define what it means for a regressor to be persistently exciting. A look at the aforementioned definition shows that the PE condition means realizing complete span, for all time, of the approximation space. For that reason, it is hard, demanding, and impractical to achieve the PE condition, especially when implementing closed-loop control. Moreover, though many researchers have studied alternative conditions that could yield persistently exciting inputs when devising control algorithms [11, 12, 13], requiring persistency of excitation conditions remains restrictive and not easily verifiable on-line.

We have in recent years investigated *Concurrent Learning* (CL), a method that was first introduced as a continuous-time (CT) uncertainty approximation method [14]. CL, much like human learning, makes use of current data and stored, past data. Without requiring persistency of excitation, CL is shown, in CT settings, to guarantee global exponential convergence of the parameter error (defined as the difference between the parameter estimates and their true values) when employed for standalone identification of structured uncertainties, while yielding uniform ultimate boundedness of the parameter error when dealing with unstructured uncertainties instead instead [15, 16, 14, 17, 18]. Those results however necessitate that the stored, past data be (at least) rich enough. More specifically, there should be the same number of linearly independent columns in the matrix containing the stored, past data as there are (row) dimensions in the used regressor for approximation, which, as discussed about before, is less demanding than requiring persistency of excitation.

To this day, several research endeavors (see [15, 16, 14, 17, 19, 18, 20, 21, 22, 23, 24, 25, 26, 27]) on CL in the CT framework have been published. For our part, to add to its already existing literature, we have looked at how the concept of CL can be implemented in the discrete-time (DT) framework. To this end, the present research intends to provide a fundamental study of the CL method for function approximation in the DT domain while using both the DT Normalized Gradient and the DT Normalized Recursive Least Square techniques. The following is a summary of our problem definition and contributions.

- We formulate a very general discrete-time function approximation problem;

- We investigate both the DT Normalized Gradient (DTNG) and the DT Normalized Recursive Learning (DTNRLS) algorithms for function approximation in DT settings. Because of the generic nature of the approximation problem that we have set out to solve, our studies of the previously mentioned techniques are not mere repetitions of the current state of literature. Rather, we add to the already existing literature work on both standard algorithms;

- Our main contributions comes from developing a DTNG based CL algorithm and a DTNRLS based CL algorithm;

- For each studied and developed algorithm, we study both the structured and unstructured uncertainty approximation cases,

- Unlike in the cases of the standard DTNG and DTNRLS algorithms, we show analytically that the DTNG based CL and DTNRLS based CL algorithms yield better parameter identification (i.e., convergence of the parameter error to the origin or a neighborhood of the origin) provided that the CL condition for richness of the stored data is verified.

Last, it is worthwhile mentioning that the derived stability results in the present work are obtained via a Lyapunov proof, which therefore makes them conservative. In that regard, violating the conditions for which those results are obtained does not necessarily or automatically mean that the same results cannot be attained.

10.1 Future Research

10.1.1 Optimizing the Developed Concurrent Learning Algorithms

When developing the CL algorithms in both Chapters IV and V, we have derived upper bounds on the Frobenius of the parameter error, with those upper bounds dependent on many variables, including the spectral properties of the learning or gain matrices. It may be the case that the gain matrices can be optimally designed so as to yield smaller upper bounds or ultimate bounds of the parameter error and, thereby, improve approximation results.

10.1.2 Data Recording for Concurrent Learning

We have seen in our numerical simulations of Chapter VII that, in the case of the DTNG based CL algorithm, though still performing well as far as parameter estimation is concerned, the adjustment term based on recording data can potentially damage function approximation performance depending on the current state of the data saved in memory. Similarly, we have seen, in the case of the DTNRLS based CL algorithm that, because the standard DTNRLS algorithm essentially uses memory, it can be difficult to show the benefit of the DTNRLS based CL algorithm aside from its theoretical guarantees. Moreover, when it comes to the data recording procedures of VI, as backed by our theoretical derivations, if/when the CL rank condition is met and the history stack $Z \in \mathbb{R}^{rz \times cz}$ or $Z_G \in \mathbb{R}^{rz \times cz}$ is full, we seek to update them so as to maximize the the metric r_{CL}. At the software level, as depict Procedures 1 and 2, this is done by replacing each of the present column in the history stack by the newly considered one(s) and computing c_Z different metric r_{CL}. To do so we use a *for loop*. Hence, as c_Z increases (and/or r_Z increases, since $c_Z \geq r_Z$), more time is used during the data recording processes, which can be a detriment when it comes to real-time implementations. It would therefore be advisable that more study be done on the generation and maintenance of the memory bank when using CL methods. It should be noted such studies are present in the CL in CT framework literature.

10.1.3 Approximation Structure

The present work only uses an approximation structure that is linear in all parameters being estimated. Nonlinear in the parameter approximator such as Multi-Layer Perceptron have been shown in literature to be able to provide improved function approximation performances. Devising CL inspired algorithms using a nonlinear in the parameter approximator while working in DT settings could therefore be an avenue for future studies.

10.1.4 More Real-Time Applications

We have studied in Chapter VIII the use of the DTNG based CL algorithm when implementing indirect adaptive control of a class of single state plant. We have also used the DTNRLS based CL algorithm for system identification of a Comau Racer industrial robot, as shows Chapter IX. Nevertheless, for the most part, the present work has been much more concerned with fundamental work than applied work. Hence, compelling research directions could therefore involve using the derived CL algorithms and applying them to more problems, especially problems involving real-time implementations.

BIBLIOGRAPHY

[1] L. Ljung, *System identification: theory for the user*, ser. Prentice-Hall information and system sciences series. Prentice-Hall, 1987.

[2] K. J. Astrom and B. Wittenmark, *Adaptive Control*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1994.

[3] G. Tao, *Adaptive Control Design and Analysis (Adaptive and Learning Systems for Signal Processing, Communications and Control Series)*. New York, NY, USA: John Wiley & Sons, Inc., 2003.

[4] M. Krstic, P. V. Kokotovic, and I. Kanellakopoulos, *Nonlinear and Adaptive Control Design*, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1995.

[5] J. Spooner, M. Maggiore, R. Ordóñez, and K. Passino, *Stable Adaptive Control and Estimation for Nonlinear Systems: Neural and Fuzzy Approximation Techniques*. New York, NY, USA: John Wiley & Sons, Inc., 2001.

[6] T. Chen and B. Francis, *Optimal Sampled-Data Control Systems*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1995.

[7] D. Nei, A. Teel, and P. Kokotovi, "Sufficient conditions for stabilization of sampled-data nonlinear systems via discrete-time approximations," *Systems & Control Letters*, vol. 38, no. 45, pp. 259 – 270, 1999.

[8] O. Djaneye-Boundjou, "Particle Swarm Optimization Stability Analysis," Master's thesis, University of Dayton, Dayton, OH, USA, Dec 2013.

[9] O. Djaneye-Boundjou, R. Ordóñez, and V. Gazi, "Stable adaptive particle swarm optimization," in *Control, Automation and Systems (ICCAS), 2013 13th International Conference on*, Oct 2013, pp. 440–445.

[10] G. Zames, "Adaptive control: Towards a complexity-based general theory," *Automatica*, vol. 34, no. 10, pp. 1161–1167, 1998.

[11] S. Boyd and S. Sastry, “Necessary and sufficient conditions for parameter convergence in adaptive control,” *Automatica*, vol. 22, no. 6, pp. 629 – 639, 1986.

[12] M. Green and J. B. Moore, “Persistence of excitation in linear systems,” in *American Control Conference, 1985*, June 1985, pp. 412–417.

[13] M. Petreczky and L. Bako, “On the notion of persistence of excitation for linear switched systems,” in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, Dec 2011, pp. 1840–1847.

[14] G. Chowdhary, “Concurrent learning for convergence in adaptive control without persistency of excitation,” Ph.D. dissertation, Georgia Institute of Technology, Atlanta, GA, USA, November 2010.

[15] G. Chowdhary and E. Johnson, “Recursively updated least squares based modification term for adaptive control,” in *American Control Conference (ACC), 2010*, June 2010, pp. 892–897.

[16] ——, “Concurrent learning for convergence in adaptive control without persistency of excitation,” in *Decision and Control (CDC), 2010 49th IEEE Conference on*, Dec 2010, pp. 3674–3679.

[17] ——, “A singular value maximizing data recording algorithm for concurrent learning,” in *American Control Conference (ACC), 2011*, June 2011, pp. 3547–3552.

[18] G. Chowdhary, T. Yucelen, M. Muhlegg, and E. Johnson, “Concurrent learning adaptive control of linear systems with exponentially convergent bounds,” *Int. Journal of Adaptive Control and Signal Processing*, vol. 27.00, no. 4, pp. 280–301, 2013.

[19] M. Mühlegg, G. Chowdhary, and E. Johnson, “Concurrent learning adaptive control of linear systems with noisy measurements,” in *AIAA Guidance, Navigation, and Control Conference*, August 2012.

[20] G. D. L. Torre, G. Chowdhary, and E. N. Johnson, “Concurrent learning adaptive control for linear switched systems,” in *2013 American Control Conference*, June 2013, pp. 854–859.

[21] B. Reish, G. Chowdhary, K. Ure, and J. P. How, “Concurrent learning adaptive control in presence of uncertain control allocation matrix,” in *AIAA Guidance, Navigation, and Control (GNC) Conference*, August 2013.

[22] M. Mühlegg, G. Chowdhary, and F. Holzapfel, “Optimizing reference commands for concurrent learning adaptive-optimal control of uncertain dynamical systems,” in *AIAA Guidance, Navigation, and Control (GNC) Conference*, August 2013.

[23] M. Mühlegg, G. Chowdhary, J. P. How, and F. Holzapfel, “Adaptive-optimal control of constrained nonlinear uncertain dynamical systems using concurrent learning model predictive control,” in *AIAA Guidance, Navigation, and Control (GNC) Conference*, August 2013.

[24] G. Chowdhary, M. Mühlegg, J. P. How, and F. Holzapfel, *Concurrent Learning Adaptive Model Predictive Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 29–47.

[25] B. Reish and G. Chowdhary, "Concurrent learning adaptive control for systems with unknown sign of control effectiveness," in *53rd IEEE Conference on Decision and Control*, Dec 2014, pp. 4131–4136.

[26] R. Kamalapurkar, B. Reish, G. Chowdhary, and W. E. Dixon, "Concurrent learning for parameter estimation using dynamic state-derivative estimators," *CoRR*, 2015.

[27] S. B. Roy, S. Bhasin, and I. N. Kar, "Memory-based data-driven MRAC architecture ensuring parameter convergence," *CoRR*, vol. abs/1602.00482, 2016.

[28] O. Djaneye-Boundjou and R. Ordóñez, "Parameter identification in structured discrete-time uncertainty without persistency of excitation," in *European Control Conference (ECC), 2015*, Linz, Austria, July 2015.

[29] ——, "Discrete-time indirect adaptive control of a class of single state systems using concurrent learning for parameter adaptation," in *IEEE Multi-Conference on Systems and Control 2016*, Buenos Aires, Argentina, September 2016.

[30] P. A. Ioannou and J. Sun, *Robust Adaptive Control*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1995.

[31] J. A. Farrell and M. M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches (Adaptive and Learning Systems for Signal Processing, Communications and Control Series)*. Wiley-Interscience, 2006.

[32] I. M. Y. Mareels, M. Gevers, R. R. Bitmead, C. R. Johnson, R. L. Kosut, and M. A. Poubelle, "How exciting can a signal really be?" *Systems & Control Letters*, vol. 8, no. 3, pp. 197–204, 1987.

[33] L. Sciavicco, B. Siciliano, and L. Villani, "Lagrange and newton-euler dynamic modeling of a gear-driven robot manipulator with inclusion of motor inertia effects," *Advanced Robotics*, vol. 10, no. 3, pp. 317–334, 1995.

[34] W. Khalil and E. Dombre, *Modeling, identification and control of robots*. Butterworth-Heinemann, 2004.

[35] M. Spong, S. Hutchinson, and M. Vidyasagar, *Robot modeling and control*. Hoboken (N.J.): John Wiley & Sons, 2006.

[36] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[37] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotic, Modelling, Planning and Control*. London: Springer, 2009.

[38] H. Mayeda, K. Yoshida, and K. Osuka, “Base parameters of manipulator dynamic models,” *IEEE Transactions on Robotics and Automation*, vol. 6, no. 3, pp. 312–321, 1990.

[39] M. Gautier, “Numerical calculation of the base inertial parameters of robots,” in *Proceedings., IEEE International Conference on Robotics and Automation*, May 1990, pp. 1020–1025 vol.2.

[40] O. Khatib, “A unified approach for motion and force control of robot manipulators: The operational space formulation,” *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, February 1987.

[41] R. Kelly, “Global positioning of robot manipulators via PD control plus a class of nonlinear integral actions,” *IEEE Transactions on Automatic Control*, vol. 43, no. 7, pp. 934–938, Jul 1998.

[42] R. Colbaugh and H. Seraji, “Adaptive tracking control of manipulators: theory and experiments,” in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, May 1994, pp. 2992–2999 vol.4.

[43] A. Zavala-Rio and V. Santibanez, “Simple extensions of the PD-with-gravity-compensation control law for robot manipulators with bounded inputs,” *IEEE Transactions on Control Systems Technology*, vol. 14, no. 5, pp. 958–965, Sept 2006.

[44] X. Xu and R. Ordóñez, “Multi-input multi-output adaptive torque control of 9-DOF hyper-redundant robotic arm,” in *International Conference on Control, Automation and Systems, 2016. ICCAS 2016. IEEE International Conference on*, October 2016.

[45] J. Hollerbach, W. Khalil, and M. Gautier, “Model identification,” in *Springer Handbook of Robotics*. Springer, 2008, pp. 321–344.

[46] J. Wu, J. Wang, and Z. You, “An overview of dynamic parameter identification of robots,” *Robotics and computer-integrated manufacturing*, vol. 26, no. 5, pp. 414–419, 2010.

[47] B. Armstrong, “On finding exciting trajectories for identification experiments involving systems with nonlinear dynamics,” *The International journal of robotics research*, vol. 8, no. 6, pp. 28–48, 1989.

[48] M. Gautier and W. Khalil, “Exciting trajectories for the identification of base inertial parameters of robots,” *The International journal of robotics research*, vol. 11, no. 4, pp. 362–375, 1992.

[49] J. Swevers, C. Ganseman, D. B. Tukel, J. de Schutter, and H. V. Brussel, “Optimal robot excitation and identification,” *IEEE Transactions on Robotics and Automation*, vol. 13, no. 5, pp. 730–740, Oct 1997.

[50] O. Djaneye-Boundjou and R. Ordóñez, “Normalized recursive least square based discrete-time concurrent learning for system identification,” submitted for journal publication.

APPENDIX A

Matrix Identity and Propositions

Consider the expression

$$B = A + UCV,$$

where matrices A, B, C, U, and V are of appropriate size. The *Woodbury Matrix Identity* or *Matrix Inversion Lemma* stipulates that

$$\begin{aligned} B^{-1} &= (A + UCV)^{-1} \\ &= A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1}VA^{-1} \end{aligned} \tag{A.1}$$

provided A^{-1} and C^{-1} exist and are computable.

The following proposition, given without proof and obtained from [3], is about convergence of monotonic functions. However, first, we define what it means for a function to be monotonic. A function $h(k)$ is monotonically increasing (or decreasing) if, for discrete-time k_1 and k_2 such that $k_2 > k_1$, $h(k_2) \geq h(k_1)$ (or $h(k_2) \leq h(k_1)$).

Proposition A.1. If a function $h(k)$ is either monotonically decreasing and bounded from below or monotonically increasing and bounded from above, then $\lim\limits_{k\to\infty} h(k)$ exists and is finite.

APPENDIX B

Persistency of Excitation

By adopting and adapting the definitions in [3], we define here what it means for a signal to be *exciting* or *persistently exciting*.

Consider $\varpi(k) \in \mathbb{R}^{r_\varpi \times c_\varpi}$, r_ϖ, $c_\varpi \in \mathbb{N}_+$, to be a bounded signal.

Definition B.1. The signal $\varpi(k)$ is exciting over the time sequence set $\{\tau, \tau+1, \ldots, \tau+\delta_\varpi\}$, $\tau \geq k_0$ and $\delta_\varpi \in \mathbb{N}_+$, if for some $\beta_\varpi > 0$ it holds that

$$\sum_{k=\tau}^{\tau+\delta_\varpi} \varpi(k)\varpi^\top(k) \geq \beta_\varpi I_{r_\varpi}. \tag{B.1}$$

Definition B.2. The signal $\varpi(k)$ is persistently exciting (PE) if there exist $\delta_\varpi \in \mathbb{N}_+$ and $\beta_\varpi > 0$ such that

$$\sum_{k=\tau}^{\tau+\delta_\varpi} \varpi(k)\varpi^\top(k) \geq \beta_\varpi I_{r_\varpi}, \forall\, \tau \geq k_0. \tag{B.2}$$

APPENDIX C

Vector Norms, Matrix Norms, and Properties

The reader is encouraged to refer to [3, 5] for proofs. Consider:

- Vector $\bar{v} = \begin{bmatrix} \bar{v}_1 & \bar{v}_2 & \ldots & \bar{v}_n \end{bmatrix}^\top \in \mathbb{R}^n$, $n \in \mathbb{N}_+$. That is v_i's, $i = 1, 2, \ldots, n$, are scalars.
- Vectors $v_a \in \mathbb{R}^{r_{v_a}}$ and $v_b \in \mathbb{R}^{r_{v_b}}$, with $r_{v_a}, r_{v_b} \in \mathbb{N}_+$.
- For $n \in \mathbb{N}_+$, the discrete-time vector signal

$$\bar{x}(k) = \begin{bmatrix} \bar{x}_1(k) & \bar{x}_2(k) & \ldots & \bar{x}_n(k) \end{bmatrix}^\top \in \mathbb{R}^n.$$

- Square matrix $A_{sq} \in \mathbb{R}^{r_{A_{sq}} \times r_{A_{sq}}}$, where $r_{A_{sq}} \in \mathbb{N}_+$.
- Matrix $A = [a_{r_a c_a}]^{r_A \times c_A} \in \mathbb{R}^{r_A \times c_A}$, where $r_A, c_A \in \mathbb{N}_+$ and, for $r_a = 1, 2, \ldots, r_A$, $c_a = 1, 2, \ldots, c_A$, $a_{r_a c_a}$ are the elements of A.
- Matrices $m_a, m_b \in \mathbb{R}^{r_m \times c_m}$ and $m_{sq_1}, m_{sq_2} \in \mathbb{R}^{r_m \times r_m}$, with r_m and c_m such that $r_m, c_m \in \mathbb{N}_+$.

We make use of the following definitions and properties.

- Let tr$(\cdot)$ denote the trace operator, which can only be applied to square matrices, and vec$(\cdot)$ denote the vectorization operator. While $\mathrm{tr}\left(A_{sq}\right) = \mathrm{tr}\left(A_{sq}^\top\right) \in \mathbb{R}$ is the sum of the main diagonal entries, $\text{vec}(A) \in \mathbb{R}^{r_A c_A}$ is obtained by appropriately stacking up columns of matrix A one after the other, starting with the very first one. We have that

$$\text{tr}\,(A_{sq}) = \sum_{r=1}^{r_{A_{sq}}} \lambda_{\{A_{sq}\}_r}, \tag{C.1}$$

with $\lambda_{\{A_{sq}\}_r}$, $r = 1, 2, \ldots, r_{A_{sq}}$, being the eigenvalues of A_{sq},

$$\text{tr}(m_{sq_1} + m_{sq_2}) = \text{tr}(m_{sq_1}) + \text{tr}(m_{sq_2}). \tag{C.2}$$

and, if m_{sq_1} is neither function of m_{sq_2} nor m_a,

$$\frac{\partial \text{tr}\,\{m_{sq1} m_{sq2}\}}{\partial m_{sq_2}} = \frac{\partial \text{tr}\,\{m_{sq2} m_{sq1}\}}{\partial m_{sq_2}} = m_{sq_1}^{\top}, \tag{C.3}$$

and

$$\frac{\partial \text{tr}\,\left\{m_a^{\top} m_{sq_1} m_a\right\}}{\partial m_a} = \left[m_{sq_1} + m_{sq_1}^{\top}\right] m_a. \tag{C.4}$$

- The ℓ^p-norm, $p \in [1, \infty)$, of vector $\bar{v}$ is defined as

$$\|\bar{v}\|_p = \left(\sum_{r=1}^{p} |\bar{v}_r|^p\right)^{\frac{1}{p}}, \tag{C.5}$$

while

$$\|\bar{v}\|_\infty = \max_{1 \le r \le n} |\bar{v}_r|, \tag{C.6}$$

where $\max(\cdot)$ denotes the maximum operator, is the ℓ^∞-norm of $\bar{v}$.

- Let $\|\cdot\|_F$ represent the Frobenius (or Hilbert-Schmidt) matrix norm operator. By definition,

$$\|A\|_F = \left(\sum_{r_a=1}^{r_A} \sum_{c_a=1}^{c_A} |a_{r_a c_a}|^2\right)^{\frac{1}{2}}. \tag{C.7}$$

Given (C.7),

$$\left\|A^{\top}\right\|_F = \|A\|_F, \tag{C.8}$$

and

$$\|A\|_F^2 = \text{vec}\,(A)^\top \text{vec}\,(A) = \|\text{vec}\,(A)\|^2 = \text{tr}\left(A^\top A\right). \tag{C.9}$$

Among other properties, the Frobenius norm is subadditive, submultiplicative, and subordinate to the ℓ^2-norm. That is, for $n \in \mathbb{N}_+$ matrices m_i, $i = 1, 2, \ldots, n$, whose dimensions are suitably and appropriately chosen (depending on (C.10) and/or (C.11)),

$$\|m_1 \pm m_2 \pm \ldots \pm m_n\|_F \leq \|m_1\|_F + \|m_2\|_F + \ldots + \|m_n\|_F\,, \tag{C.10}$$

$$\|m_1\, m_2 \ldots m_n\|_F \leq \|m_1\|_F\, \|m_2\|_F \ldots \|m_n\|_F\,, \tag{C.11}$$

and, given a vector $v_a \in \mathbb{R}^{c_m}$,

$$\|m_a v_a\| \leq \|m_a\|_F\, \|v_a\|\,, \tag{C.12}$$

where $\|\cdot\|$ is the ℓ^2-norm operator. Additionally, for some positive scalar $\overline{\beta}$, such that

$$1 < \overline{\beta} < \infty,$$

and assuming that matrix A_{sq} is invertible,

$$I_{r_{A_{sq}}} = A_{sq} A_{sq}^{-1} < \overline{\beta} I_{r_{A_{sq}}}.$$

Given the submultiplicate property (see (C.11)) of the Frobenius norm, we can write

$$\left\|I_{r_{A_{sq}}}\right\|_F \leq \|A_{sq}\|_F\, \|A_{sq}^{-1}\|_F < \overline{\beta} \left\|I_{r_{A_{sq}}}\right\|_F.$$

Thus,

$$\frac{\left\|I_{r_{A_{sq}}}\right\|_F}{\|A_{sq}\|_F} \leq \|A_{sq}^{-1}\|_F < \overline{\beta} \frac{\left\|I_{r_{A_{sq}}}\right\|_F}{\|A_{sq}\|_F}. \tag{C.13}$$

- Consider the following:

– We have

$$\text{tr}\left(m_a^\top m_b\right) = \text{tr}\left(m_b^\top m_a\right) = \text{vec}(m_a)^\top \text{vec}(m_b).$$

Therefore, from (C.9),

$$\text{tr}\left(m_{sq_1}^\top m_{sq_1}\right) = \text{vec}(m_{sq_1})^\top \text{vec}(m_{sq_1}) = \|\text{vec}(m_{sq_1})\|^2 = \|m_{sq_1}\|_F^2\,, \tag{C.14}$$

$$\text{tr}\left(m_a^\top m_b\right) \leq \left|\text{tr}\left(m_a^\top m_b\right)\right| \leq \|m_a\|_F \|m_b\|_F\,, \tag{C.15}$$

and, given (C.11) and (C.15), for $n \in \mathbb{N}_+$ dimensionally well-chosen matrices m_i, $i = 1, 2, \ldots, n$,

$$\text{tr}\left(m_1 m_2 \cdots m_n\right) \leq \left|\text{tr}\left(m_1 m_2 \cdots m_n\right)\right| \leq \|m_1\|_F \|m_2\|_F \cdots \|m_n\|_F\,. \tag{C.16}$$

– Similarly, $\text{tr}\left(v_a v_b^\top\right) = v_a^\top v_b = v_b^\top v_a = \text{tr}\left(v_b v_a^\top\right)$, which means $\left|\text{tr}\left(v_a v_b^\top\right)\right| \leq \|v_a\| \|v_b\|$ and

$$\left\|v_a v_a^\top\right\|_F \triangleq \text{tr}\left(v_a v_a^\top\right) = v_a^\top v_a = \|v_a\|^2\,. \tag{C.17}$$

- Norms and signal spaces [3]:

  – For $p \in [1, \infty)$ and $\|\cdot\|_\ell$ representing any vector norms, the $\mathcal{L}^p$ signal norm of $\bar{x}(k)$ can be defined as

$$\|\bar{x}(k)\|_{\mathcal{L}^p} = \left(\sum_{k=0}^{\infty} \|\bar{x}(k)\|_\ell^p\right)^{\frac{1}{p}}. \tag{C.18}$$

  Moreover, the $\mathcal{L}^\infty$ signal norm of $\bar{x}(k)$ is given as

$$\|\bar{x}(k)\|_{\mathcal{L}^\infty} = \sup_{k \geq 0} \|\bar{x}(k)\|_\infty\,, \tag{C.19}$$

  where $\sup(\cdot)$ stands for the supremum operator

- The $\mathcal{L}^p$, with $p \in [1, \infty)$, and $\mathcal{L}^\infty$ signal spaces are respectively

$$\mathcal{L}^p = \left\{ \bar{x}(k) \in \mathbb{R}^n : \|\bar{x}(k)\|_{\mathcal{L}^p} < \infty \right\}, \tag{C.20}$$

$$\mathcal{L}^\infty = \left\{ \bar{x}(k) \in \mathbb{R}^n : \|\bar{x}(k)\|_{\mathcal{L}^\infty} < \infty \right\}. \tag{C.21}$$

- Some properties:

  * For $p_1, p_2 \in [1, \infty)$ such that $p_1 \leq p_2$, if $\bar{x}(k) \in \mathcal{L}^{p_1}$ then $\bar{x}(k) \in \mathcal{L}^{p_2}$. For instance, if $\bar{x}(k) \in \mathcal{L}^1$ then $\bar{x}(k) \in \mathcal{L}^p$, $p \in (1, \infty)$.

  * Moreover, if $\bar{x}(k) \in \mathcal{L}^p$, for $p \in [1, \infty)$, then $\bar{x}(k) \in \mathcal{L}^\infty$ and $\lim\limits_{k \to \infty} \bar{x}(k) = [0]^n$ or $\lim\limits_{k \to \infty} \|\bar{x}(k)\| = 0$.