TÉCNICO LISBOA

Automatic Design of Controller Parameters based on Reinforcement Learning

Beatriz Ventura dos Santos Pereira

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Dr. Paulo André Nobre Rosa
Prof. João Manuel Lage de Miranda Lemos

Examination Committee

Chairperson: Prof. João Manuel de Freitas Xavier
Supervisor: Dr. Paulo André Nobre Rosa
Member of the Committee: Prof. Rita Maria Mendes de Almeida Correia da Cunha

June 2023

I declare that this document is an original work of my own authorship and that it fulfils all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I would like to thank my parents and my sister for their friendship, encouragement, and caring over all these years, for always being there for me through thick and thin and without whom this project would not be possible. It was your unconditional love, understanding, and support that got me through all these years.

I would also like to acknowledge my dissertation supervisors Prof. João Miranda Lemos and Dr. Paulo Rosa for their insight, patience, support, and sharing of knowledge that has made this Thesis possible.

Thank you to all the people I met at Deimos offices in Lisbon, for welcoming me with open arms and being so available.

A special thank you to my close friends from Instituto Superior Técnico, without whom this experience would not have been so memorable since the very first day.

Last but not least, to all my incredible friends from my hometown and important people I met over the years, that helped me grow as a person and were always there for me during the good and bad times in my life. Thank you.

To each and every one of you – Thank you.

Abstract

Due to the increasing demand for Reusable Launch Vehicles (RLVs) to improve the sustainability of rocket launching, research in vertical landing and recovery of rockets has gained significance in recent years. Therefore, the objective is to investigate novel algorithms, implemented as a software package, for the automated design of linear controllers using approaches based on the Youla Parameterization (YP) and Reinforcement Learning (RL) to tackle the attitude control problem of a landing RLV.

The YP generates the set of all the linear controllers that stabilize a given linear plant by varying a stable transfer function known as the Youla parameter. We propose the adoption of a novel approach to adjust the Youla parameter, employing an RL algorithm, known as Episodic REINFORCE, for unstable systems. Limited research has been conducted on RL for feedback design in circumstances where instability can interfere with the learning or the hardware. The commonly used parameterizations tend to perform poorly in such cases. The combination of RL and YP offers stability, robustness, and performance advantages. The results obtained in this work demonstrate that the problem is successfully addressed, with the algorithm being validated and compared against a state-of-the-art optimization algorithm. An approach to accelerate the convergence of the learning algorithm was proposed.

The integration of RL with YP provides new possibilities for designing robust and efficient control systems for RLVs. This work contributes to the advancement of reusable launch technology by providing an approach to tackle the challenges associated with automatic attitude control.

Keywords

Automatic controller design, Youla-Kucera parameterization, reinforcement learning, REINFORCE, rocket landing

Resumo

Devido à crescente procura por Veículos de Lançamento Reutilizáveis (RLVs) para reduzir os efeitos do lançamento de foguetões, a investigação sobre a aterragem e recuperação de foguetões tem ganho importância nos últimos anos. Portanto, o objetivo deste trabalho é desenvolver um pacote de software para o projeto automático de controladores lineares através de uma abordagem baseada na Parametrização de Youla (YP) e na Aprendizagem por Reforço (RL), de modo a resolver um problema do controlo de atitude na aterragem de um RLV.

Através desta parametrização é possível obter o conjunto de todos os controladores lineares que estabilizam uma dada planta linear, ao variar uma função de transferência estável conhecida como parâmetro de Youla. Propomos uma nova abordagem para ajustar este parâmetro, com um algoritmo de RL, conhecido como Episodic REINFORCE, para sistemas instáveis. A investigação sobre RL no projeto de sistemas de feedback em circunstâncias em que a instabilidade pode interferir com a aprendizagem ou o *hardware* é limitada, e as parametrizações clássicas tendem a ter um desempenho fraco nestes casos. Por isso, a aprendizagem foi combinada com a YP, que oferece vantagens em termos de estabilidade, robustez e desempenho. Os resultados obtidos solucionam o problema proposto, validando-o com um algoritmo de otimização do estado da arte. Apresentou-se, também, uma abordagem para acelerar a convergência do algoritmo.

Este trabalho contribui para o avanço da tecnologia de lançamento reutilizável, visto que a junção de RL com a YP abre novas possibilidades para o projeto automático de controladores em sistemas complexos.

Palavras Chave

Projeto automático de controladores, parameterização de Youla-Kucera, aprendizagem por reforço, RE-

INFORCE, aterragem de lançadores

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

ARE	Algebraic Riccati Equation
BIBO	Bounded Input, Bounded Output
CAP	(Temporal) Credit Assignment Problem
CLTI	Continuous-time Linear Time-Invariant (LTI)
COG	Center of Gravity
LHP	Left Half of s-Plane
LQ	Linear Quadratic
LQG	Linear Quadratic Gaussian
LQR	Linear Quadratic Regulator
LTI	Linear Time-Invariant
MISO	Multiple-Input Single-Output
MPC	Model Predictive Control
MSE	Mean Squared Error
NCFMR	Normalized Coprime Factorization
NRMSE	Normalized Root Mean Squared Error
PDF	Probability Density Function
RL	Reinforcement Learning
RLV	Reusable Launch Vehicle
RMS	Root Mean Square
SISO	Single-Input Single-Output
SRL	Symmetric Root Locus
SSE	Sum of Squares Error
TVC	Thrust Vector Control

YK Youla-Kucera

YP Youla Parameterization

1

Introduction

Contents

1.1 Motivation and Problem Definition

The design of a controller that complies with prescribed specifications and constraints is a complex task, even for linear systems. To increase the efficiency of the design process, it is convenient to have an automatic design tool that allows an efficient exploration of the effects of different specifications, as well as of the trade-offs between performance and robustness.

This design tool, to be embedded in a software package, must rely on a suitable controller parameterization, that ensures that the search for the optimal controller is made only among stabilizing controllers, combined with an optimization algorithm that can tackle high-dimension problems with complex objective functions, possibly non-convex.

The main goal of this work is to develop a software package using MATLAB for the automatic design of controllers for linear, time-invariant, continuous-time systems, with application in aerospace vehicles' motion control. The algorithm will be exemplified using an attitude control problem for a space vehicle, in this case, a vertical landing rocket.

For such a system, the Youla Parameterization (YP) provides a way to parameterize all the controllers that stabilize it ([1], [2]). By varying the Youla parameter Q to optimize the cost functional that defines the goal, a technique called Q-Design, it is possible to find the controller that can satisfy the problem restrictions and guarantee stability. The automation of this process will be done with Reinforcement Learning (RL), with focus on the algorithm Episodic REINFORCE ([1], [5], [6]). The use of RL for feedback design systems when instability can disrupt the learning process is not yet heavily studied since the most common parameterizations perform poorly in those situations. On the other hand, through the Q-parameterization, the YP offers a set of robust stabilizing controllers. This approach will provide the tool used to develop the algorithm to tune controllers, given a linearized model of a rocket landing.

The possibility of extending the YP, present in the control literature, to more complex systems aligned with the use of RL represents a door to apply learning to real-world problems, such as the vertical landing of reusable rockets, a case study of importance to the field of Reusable Launch Vehicles (RLVs).

1.2 Scientific Framework and State of the art

The subjects adressed by this work include: automatic controller design and parameterization, optimization problems, and RL.

The Youla-Kucera (YK) parameterization is a popular control literature parameterization that had its origin in the 70s when Youla and others developed an analytical feedback design technique for the Single-Input Single-Output (SISO) case and the multivariable case (see [7], [8]) in continuous-time and Kucera added the extension to the discrete-time case [9]. These works proposed the YK parameterization, also commonly referred to as YP, capable of providing all linear stabilizing controllers for a given

Linear Time-Invariant (LTI) plant in a feedback control loop, based on the transfer function Q (Youla parameter), that guarantees the stability. This work was of engineering significance and launched an entirely new area of research, among other areas, in optimal and robust control [10].

Likewise, the dual YK parameterization presents all the linear plants that are stabilized by a given controller, based on the transfer function S (dual YK parameter). While the Q-parameterization is mostly used for stable controller reconfiguration, disturbance, and noise rejection control, S-parameterization works to solve closed-loop identification. These approaches have been studied for decades to achieve a higher control performance [11].

In Kucera's simplified version of the YK for teaching [10], it is stated that the time-varying system case was generalized later on by *e.g.* [12], infinite dimensional systems by *e.g.* [13], [14], and a class of non-linear systems by *e.g.* [15], [16] and [17].

Furthermore, [17] was the first survey paper that explored both YK and Dual YK parameterization. This work investigated solutions of Q-parameterization for H_∞ and H_2 problems with constrained pole positions and closed-loop identification problems, by reducing them to standard open-loop problems. Besides, it described early work on YK parameterization for nonlinear systems and disturbance rejection.

In 2020, a group of french researchers collected the advances in YK parameterization [18], classifying them according to the use of YK parameterization, Dual YK parameterization, or both. This paper presents the main applications of this tool in various control fields, such as optimal control, robust control, Q-based controller reconfiguration, noise rejection, and vibration control, S-Based closed-loop identification, (Q,S)-based adaptive control and fault tolerant control.

The article [19] studies control methodologies for precision positioning systems and presents a tutorial for loop-shaping control, that uses the YK parameterization as a tool for achieving the design goal. This can be done for flexible servo designs because a precision positioning system is designed to have an LTI plant and when a SISO plant is controlled by linear controllers, the position servo can be cast as a loop-shaping problem.

The Q-parameterization provides the set of all stabilizing controllers for a given plant that can be characterized by knowing a controller stabilizing the given plant [18]. Many important cost functions are convex in the $Q(s)$ because the closed-loop system is affine in $Q(s)$. Thus, the Youla parameter has been used as a tool for designing feedback controllers through optimization methods, as seen for example in [2] and [20].

More specifically, Hespanha [2], in lectures 24 and 25, shows how a given Linear Quadratic Gaussian (LQG)/Linear Quadratic Regulator (LQR) controller can be used to parameterize all feedback controllers capable of stabilizing a given LTI system, to be used as a control design method based on Q parameterization and numerical optimization. Furthermore, this reference mentions a finite-dimensional optimization technique known as the *Ritz Approximation*.

In [20] is proposed an affine representation of $Q(s)$ and, in chapter 15, the *Ritz Approximation* is discussed. Moreover, a tutorial on an approach for designing linear controllers based on numerical convex optimization is provided in [21].

The methods expressed in this paper can also be used to test the limits of performance of controllers when there are no non-convex constraints.

RL is the technique used by an agent that must learn behavior through trial-and-error interactions with a dynamic environment [22]. RL offers a set of methods for learning controllers with many control applications, especially in robotics *e.g.* [23], [24].

Another example of a work that mixes classic control and modern RL is [25]. For that sake, this paper proposes to use RL in the context of model-based control with application in a two-degree-of-freedom robot manipulator. For the feedback controller to be able to compensate for the uncertainties, the learning made in the framework of stabilizing controllers uses only little prior model knowledge.

Ronald J. Williams [6] presents a general class of associative RL algorithms, called REINFORCE algorithms, for connectionist networks containing stochastic units. This paper gives specific examples of these algorithms and presents how they might be used to help develop similar but potentially more powerful RL algorithms. Also, in [5], chapter 13.4, it is explained how to obtain the REINFORCE from a gradient algorithm.

However, the performance of RL in learning controllers is highly dependent on the controller parameterization used. A poorly chosen parameterization can result in an unstable controller, a higher cost function, and weak learning performance.

Over time, the successful cases of learning have focused on learning an open-loop trajectory, since those trajectories can use stabilizing controllers designed with traditional control techniques, such as Model Predictive Control (MPC) or LQR. Thus, situations in which the feedback policies have been learned directly on hardware are less frequent and not used when the instability of the feedback policies can disrupt the learning. As a consequence of not much work being done on systems where instability is an issue, even when the policy converges, RL has not been used for high-performance issues near the margin of stability. This has to do with the fact that most natural parameterizations tend to present a poor performance in this domain.

Roberts *et al.*, [1] explore four different parameterizations of linear feedback control in the context of REINFORCE with application in the control of a reaching task with a linearized flexible manipulator, where closed-loop instability can be an issue. Here, the manipulator is modeled as an open-loop stable linear system that is underactuated, does not have full-state information, and, with the wrong controller, can quickly become unstable.

This paper states that the two most natural-seeming parameterizations *i.e.*, state feedback gains with a fixed observer and a feedback controller transfer function do not guarantee stability, the set of stabiliz-

ing controllers are non-convex and even a small change in the parameters can worsen performance to the point of instability. On the other hand, LQR cost matrices with a fixed observer and YP do guarantee stability.

The two parameterizations offer a different set of advantages: the LQR cost function matrix offer high-performance and high-bandwidth controllers (when there is no delay), the noise is not excessively structured and the cost function is similar to the quadratic cost assumed by LQR. On the other hand, the YP provides a rich set of controllers and presents better performance when the cost function is non-quadratic and the noise is structured but non-Gaussian. Regarding the experimental results, it was concluded that the YP provided the best overall performance. It had a quick predictable convergence, which showed good learning performance and good ultimate controller performance. These results validate the premise that the representation for $Q(s)$ has a great influence on the performance of the parameterization and that with the appropriate choice of $Q(s)$ any stabilizing linear controller can be represented, even the LQR controller.

Despite being popular in the control literature and control theory research, the extension of the YP to more complex systems is still a new field that offers a large range of opportunities for study, such as: open-loop unstable plants, multivariable systems (*e.g.* [8], [20]), rapid switching between controllers (*e.g.* [26]), uncertain models (*e.g.* [27], [20], [28]), nonlinear systems (*e.g.* [17], [29], [16]), and decentralized systems (*e.g.* [30]).

Switching between controllers is a form of improving performance while controlling complex systems, acting on multiple control objectives, and guaranteeing stability. Moreover, switching between controllers using YK parameterization presents many advantages [18]. These include allowing stable switching between open-loop unstable controllers, the fact that switched controllers can be designed and adjusted separately via techniques like LQR, and guaranteeing closed-loop stability even under arbitrary controller reconfiguration.

In conclusion, it is possible to affirm that the application to nonlinear and uncertain systems, such as motion control of aerospace vehicles, is of real-world importance, and applying learning to hardware could benefit from the flexibility and guaranteed stability of the YP.

Research in vertical landing and recovery of rockets has gained prominence in recent years, with the increasing need for RLVs to minimize cost, time, and impact associated with rocket launching. The upswing is related to the successful breakthroughs made by SpaceX and Blue Origin in this field.

In the future, there will be an increasing search for reusable rocket development. Therefore, the robustness, reliability, and autonomy of entry guidance systems are of great importance. Z. Bonjun *et al.* [31] propose a new predictive atmospheric entry guidance algorithm based on dual-channel attitude control of a low-lift entry vehicle, to achieve an expected landing site before landing with powered descent. The results obtained suggest that the proposed algorithm is very robust, can accurately detect

aerodynamic acceleration error, and is capable of achieving higher accuracy of predictive guidance.

The fundamental principles of launch vehicle control analysis and design can be found in [32], applied to ARES-I Crew Launch Vehicle.

Although the synthesis of robust controllers for the ascent phase of a rocket is an already researched topic, research on the descent phase is still lacking. The progress and challenges in precision rocket landing on Earth and other planets are addressed, for example, in [33] and [34].

The work of M. Sagliano *et al.* [35] focuses on feedback control techniques applied to the descent phase of a reusable rocket, in particular robust control techniques such as the H_∞ concept.

A study case of spacecraft landing on asteroids with irregular shapes and low gravity can be found in [36].

1.3 Objective and Contributions

The objective of this work is to develop a linear controller design algorithm, based on the Youla Parameterization and Reinforcement Learning to adjust the Youla parameter Q, to solve an attitude control problem of a landing RLV. The results will be compared to a state-of-the-art optimization algorithm.

An important contribution of this study is the extension of the approach proposed in [1] to account for open-loop unstable plants. By combining the learning algorithm Episodic REINFORCE and YP, this study addresses not only the attitude control of an RLV but also the challenges that more complex systems pose for reference tracking.

1.4 Organization of the Document

This thesis is organized as follows: Chapter 1 presents the motivation behind the work conducted in this thesis, as well as the objectives and contributions. Additionally, an overview of previous studies that have dealt with the subject of this thesis is presented. In Chapter 2 the problem is described and the required control parameterization, known as the Youla parameterization, is defined. Chapter 3 addresses the algorithms required for optimization and learning in this work. In Chapter 4 the nonlinear model of a rocket vertical landing is derived and linearized. The baseline controllers are designed for this model. Chapter 5 presents the results, implementation details, and discussions of the conducted research. Chapter 6 conveys the conclusions of the investigation along with suggestions for further research.

2

Controller Design

Contents

In this chapter, we define the problem investigated in this thesis. For that sake, the parameterization that gives all linear stabilizing controllers for a determined LTI system, the YK Parameterization, is defined both in the transfer function and state-space frameworks. Furthermore, the parameterization is expanded to include open-loop unstable cases in addition to open-loop stable systems. Finally, an overview of Q design and numerical optimization of the Youla Parameter $Q(s)$ is presented.

2.1 Problem Definition

Let us define the general idea of the problem. The primary goal of this research is to develop an algorithm that can automatically design controllers for linear systems, with application in the motion control of aerospace vehicles. Using the parameterization described in the next section, called YK parameterization, it is possible to find all the controllers that meet the problem's constraints and ensure stability, by varying the Youla Parameter $Q(s)$.

The objective is to find the optimal vector of parameters that defines the controller as a finite dimension linear combination of basis transfer functions, using an RL algorithm called Episodic REINFORCE, to be defined in Chapter 3, as an optimization algorithm to adjust the stabilizing controller obtained, to minimize a defined cost function.

The optimal parameter ϕ of the Youla Parameter $Q(s)$, is in an infinite-dimensional space. Therefore, it is necessary to approximate it by a search over a finite-dimensional space, as to be detailed in the sections below. For that purpose, one takes

$$\hat{Q}(s) = \sum_{i=1}^{K} \phi_i \left(\frac{\alpha}{s+\alpha} \right)^{i-1}, \quad \phi := \begin{bmatrix} \phi_1 \\ . \\ . \\ . \\ \phi_K \end{bmatrix}, \tag{2.1}$$

where α is a fixed positive constant closed-loop pole that is selected a priori.

As previously mentioned, the primary application of the algorithm is to solve an attitude control problem of a space vehicle. In this case, a vertical landing model derived in the next chapter. The attitude control cost function

$$J = \sum_{j=1}^{t} (\theta_j - \theta_{goal_j})^2, \tag{2.2}$$

punishes deviation from the goal attitude angle of the vehicle, given that the error is the sum of the squared difference between the simulation output angle, θ, and the reference angle θ_{goal}.

The design tool embedded in this software package must be able to handle complex systems such as high-order systems and open-loop unstable plants with non-minimum phase zeros. For that scenario,

the cost function used can be

$$J = c_2 \sum_{j=1}^{t} (y_j - y_{mj})^2 + c_{inv} \sum_{j=1}^{t} (\textit{sign}(y_j \times y_{ref_j}) < 0), \tag{2.3}$$

which corresponds to the error is the sum of the squared difference between the response of the controlled system, y, and the response of the ideal transfer function (reference) to the impulse, y_m. This cost function also penalizes the inverse response of the closed-loop to the step impulse, by counting the negative values returned by the MATLAB function *sign*. The values c_2 and c_{inv} were used to balance the relative importance of the two terms in the cost.

Next, as to be detailed in Chapter 3, the optimization can be conducted using an RL algorithm or, for example, a MATLAB unconstrained optimization problem solver such as *Fminunc*. Finally, it will be possible to obtain the optimal stabilizing controller for the problem.

2.2 Youla-Kucera Parameterization

This section defines a parameterization that provides all linear stabilizing controllers for a given LTI plant, the YK parameterization. It is parameterized by a stable transfer function called Youla parameter, Q.

2.2.1 Control System Structure

Theorem 1

For the SISO case consider, in a general form, a linear system described by the continuous transfer function

$$\frac{B(s)}{A(s)}, \quad \partial B < \partial A. \tag{2.4}$$

Now, let $\frac{S^0(s)}{R^0(s)}$ be a stabilizing controller, interconnected to the plant as in figure 2.1.

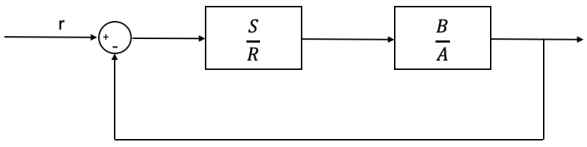


Figure 2.1: Feedback control loop block diagram.

All rational stabilizing controllers can be defined as

$$\frac{S(s)}{R(s)} = \frac{S^0(s) + Q(s)A(s)}{R^0(s) - Q(s)B(s)}, \tag{2.5}$$

where $Q(s)$ is stable.

Proof

First, we will start by proving that the controller given by (2.5) is stable. Consider that $Q(s)$ is stable and can be expressed as

$$Q(s) = \frac{Y(s)}{X(s)}, \tag{2.6}$$

where $X(s)$ and $Y(s)$ are polynomials. The stabilizing controller (2.5) can be rewritten as

$$\frac{S(s)}{R(s)} = \frac{S^0(s) + \frac{Y(s)}{X(s)}A(s)}{R^0(s) - \frac{Y(s)}{X(s)}B(s)} = \frac{X(s)S^0(s) + Y(s)A(s)}{X(s)R^0(s) - Y(s)B(s)}. \tag{2.7}$$

From figure 2.1 it is possible to infer the transfer function from r to y, given by

$$H(s) = \frac{\frac{B(s)}{A(s)}\frac{S(s)}{R(s)}}{1 + \frac{B(s)S(s)}{A(s)R(s)}} = \frac{B(s)S(s)}{A(s)R(s) + B(s)S(s)}. \tag{2.8}$$

The closed-loop characteristic polynomial is given by

$$AR + BS = A(XR^0 - YB) + B(XS^0 + YA) = X(AR^0 + BS^0). \tag{2.9}$$

Given that $Q(s)$ is stable, $X(s)$ must have all its roots located in the left half of the complex plane $Re(s) < 0$. Thus, the closed-loop is stable.

We will now prove that **all** the stabilizing controllers can be written in the form (2.5), with stable Youla parameter $Q(s)$.

Let $\frac{S(s)}{R(s)}$ be a stabilizing controller that yields the closed-loop characteristic polynomial

$$AR + BS = C, \tag{2.10}$$

with all roots of $C(s)$ located strictly in the left half of the complex plane.

Now, let us show that there is a stable function $Q(s)$, such that this controller can be written in the form (2.5).

From (2.5) it is possible to infer

$$S(R^0 - QB) = R(S^0 + QA) \equiv SR^0 - RS^0 = Q(AR + BS). \tag{2.11}$$

Thus, from (2.10) the Youla Parameter $Q(s)$ can be written as

$$Q(s) = \frac{S(s)R^0(s) - R(s)S^0(s)}{A(s)R(s) + B(s)S(s)}, \tag{2.12}$$

which is stable, since $C(s)$ has all its roots located strictly in the left half of the complex plane.

2.2.2 Special Case: Open-loop Stable System

Here, we will go over a special case example of the YK parameterization for a finite-dimensional SISO stable plant $P(s)$ and feedback controller $K(s)$ that was presented in [1]. In this case, the closed-loop system from u to y with positive feedback is

$$H(s) := \frac{Y(s)}{X(s)} = \frac{P(s)}{1 - P(s)K(s)}. \tag{2.13}$$

Furthermore, the Youla parameter associated with the controller $K(s)$ is given by

$$Q(s) := \frac{K(s)}{1 - K(s)P(s)}. \tag{2.14}$$

The controller $K(s)$ can also be written as a function of a plant $P(s)$ and a Youla parameter $Q(s)$:

$$K(s) = \frac{Q(s)}{1 + Q(s)P(s)}. \tag{2.15}$$

Thus, the closed-loop system can be presented as

$$H(s) = P(s)[1 + Q(s)P(s)], \tag{2.16}$$

that is affine in $Q(s)$. It is important to note that the relationship between the parameters of the feedback controller $K(s)$ and the closed-loop system $H(s)$ is nonlinear.

Moreover, it is clear that for any $K(s)$, the parameter (2.14) exists, and if the plant $P(s)$ is stable, then $H(s)$ is also stable, if and only if the Youla parameter $Q(s)$ is stable.

Figure 2.2 shows an intuitive approach to YK parameterization. Here, a stable system $P(s)$ with a possible perturbation ω is connected to a given feedback controller $K(s)$ consisting of a copy of the system dynamics without the perturbation, and an arbitrary stable system $Q(s)$ intervenes to compute the difference. It is the combination of $P(s)$ and $Q(s)$, where $Q(s)$ varies over all stable linear systems, that makes it possible to obtain all stabilizing controllers.

This representation of the YK parameterization presented by Roberts *et al.* [1] is a particular case of the one shown in section 2.2.1. It is possible to obtain the controller (2.15) in the form of (2.5), if the system is open-loop stable.

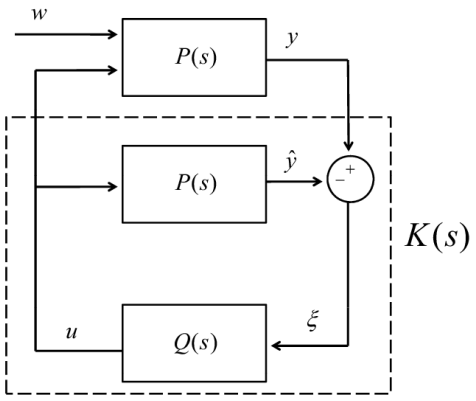


Figure 2.2: Intuitive view of the YK parameterization block diagram for an open-loop stable system from [1].

For that, consider the stabilizing controller $\frac{S^0}{R^0}$ to be $S^0(s) = 0$ and $R^0(s) = A(s)$, with positive feedback. Hence, one can infer (2.15) from (2.5) by rearranging the equation

$$\frac{S^0(s) + Q(s)A(s)}{R^0(s) + Q(s)B(s)} = \frac{Q(s)A(s)}{A(s) + Q(s)B(s)} = \frac{Q(s)}{1 + Q(s)\frac{B(s)}{A(s)}} = \frac{Q(s)}{1 + Q(s)P(s)}. \tag{2.17}$$

In conclusion, every feedback controller $K(s)$ can be represented by $Q(s)$. If and only if $Q(s)$ is stable and affine, the closed-loop system will also be stable and affine. Moreover, because the closed-loop system $H(s)$ is affine in $Q(s)$, many cost functions, including LQG, are convex in $Q(s)$.

As a result, we may state that the set of all stable $Q(s)$ is an affine parameterization of all stabilizing linear controllers for a plant $P(s)$.

2.2.3 Influence of the Youla-Kucera Parameter Q

We will now study the influence of different examples of Youla Parameter transfer functions $Q(s)$ on the step response. Several representations of stable $Q(s)$ could be considered. For these simulations, it is of the form

$$Q(s) = \frac{\prod_{i=0}^{n} \beta_i s^i}{\prod_{i=1}^{n} (s - \alpha_i)}. \tag{2.18}$$

A stable, second-order plant $P(s)$ with natural frequency $\omega = 30$ rad/s and damping ratio $\xi = 0.5$ was considered for the study. The set of parameters $Q(s)$ includes a second order system with two complex conjugate poles $Q_1(s)$, and two different stable transfer functions, $Q_2(s)$ and $Q_3(s)$, presented below.

$$Q_1(s) = \frac{400}{s^2 + 4s + 400}, \tag{2.19}$$

$$Q_2(s) = \frac{0.7s}{(s + 0.5)(s + 0.2)}, \tag{2.20}$$

$$Q_3(s) = \frac{0.001s}{s + 0.001}. \tag{2.21}$$

Figure 2.3 illustrates the closed-loop system step responses for the parameters $Q(s)$.

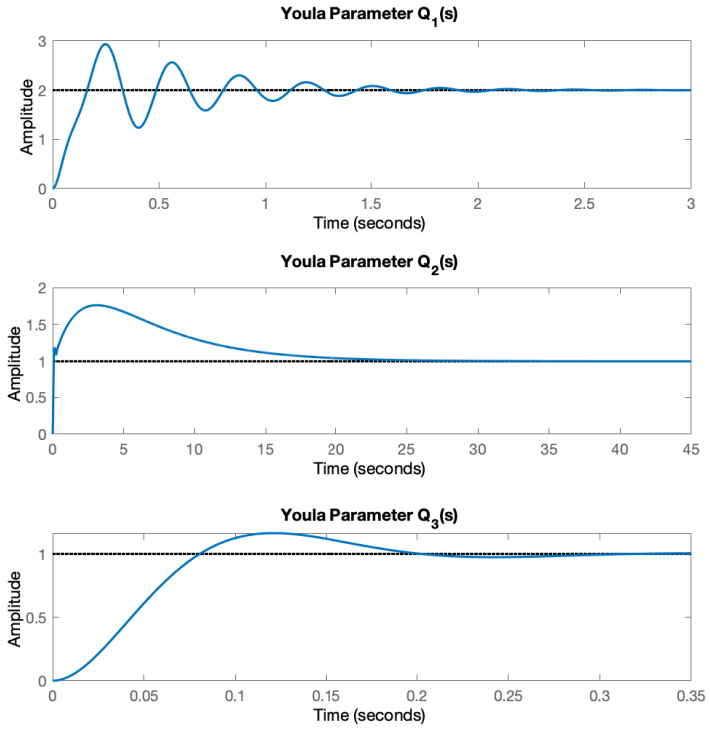


Figure 2.3: Influence of different YK parameters $Q(s)$ on the closed-loop system step response.

The first parameter, $Q_1(s)$, has damping ratio $\xi = 0.1$, which means the step response presents damped oscillations (decreasing amplitude). The second and third cases both tend to one over time, but the Youla Parameter $Q_2(s)$ leads to higher settling time, and $Q_3(s)$ has less overshoot. Note that, poles located closer to the origin are called dominant poles, which contribute more to the system response.

It is known that the closed-loop system will always be stable if and only if $Q(s)$ is stable. Nevertheless, it is not possible to guarantee that the controller $K(s)$ itself is stable, as an isolated system.

2.2.4 General Case: Open-loop Unstable System

As previously expressed in equation (2.5) that defines all rational stabilizing controllers, there is an initial stabilizing controller. This section introduces an LQG as the $\frac{S^0(s)}{R^0(s)}$ controller. The goal is to achieve a more robust controller and generalize the parameterization to a wider set of cases, such as open-loop unstable systems.

Let us consider the state-space framework. Take a Continuous-time LTI (CLTI) system written as

$$\dot{x} = Ax + Bu, \quad y = Cx, \quad x \in \mathbb{R}^n, u \in \mathbb{R}^k, y \in \mathbb{R}^m, \tag{2.22}$$

in which u represents the control signal and y the measured output.

A LQG/LQR output feedback controller can of the form

$$\hat{\dot{x}} = (A - LC)\hat{x} + Bu + Ly, \quad u = -K\hat{x}, \tag{2.23}$$

where $A - LC$ and $A - BK$ are stability matrices, and $\hat{x}$ is an estimate of the state x of the process, based on the past values of the measured output y and control signal u.

Hence, substituting the control law u in the system, it is possible to rewrite the state model for the controller (2.23) as

$$\hat{\dot{x}} = (A - LC - BK)\hat{x} + Ly, \quad u = -K\hat{x}. \tag{2.24}$$

The separation theorem shows that in a regulator obtained by feeding back the state estimate, the observer gains and the controller gains can be designed independently of each other [37]. Therefore, the poles of the overall system are grouped in two sets. The first set depends on the control gain vector K, assuming that all the components of the state are available. The second set depends on the observer gain vector L, as if the observer acts on the system without control. The characteristic polynomial of the full system corresponds to the product of the characteristic polynomials of the matrices $A - BK$ and $A - LC$.

The control gain vector K is computed such as to minimize the quadratic cost

$$J = \int_0^\infty (x^T Q x + R u^2) dt. \tag{2.25}$$

Then, by solving (2.25) it possible to find the positive definite matrix P that verifies the Algebraic Riccati Equation (ARE)

$$A^T P + P A^T - P B R^{-1} B^T P + Q = 0, \tag{2.26}$$

and the vector of controller gains K is obtained by the equation

$$K = R^{-1} B P. \tag{2.27}$$

Consider an infinite horizon. If the pair (A, B) is controllable and (C, A) is observable, a positive definite solution for the ARE (2.26) exists, is unique, and the close-loop system is asymptotically stable.

Hence, to design the controller (2.24) it is necessary to first assure the pair (A, B) is controllable,

so the closed-loop eigenvalues of a state feedback controller can be located anywhere in the complex plane. The pair (A, B) is controllable if the rank of the controllability matrix, given by the number of linearly independent rows or columns, is equal to the dimension of the state. In a controllable system there always exists a control input $u(t)$ that transfers any state of the system to any other in a finite time.

Next, it is necessary to study if the state realization is observable by checking if the observability matrix rank is equal to the state dimension. If so, it is possible to determine the state estimator gains in order to put eigenvalues of the error dynamics $(A - LC)$ in arbitrary points of the complex plane. Also, the initial state $x(t_0)$ can be estimated based on the knowlegde of the output $y(t)$ for a finite time.

The matrix Q_{LQ} determines the weights of the states, and the matrix R defines the weights of the control input in the cost function, so that the control system performs as desired. The influence of the weight matrices Q_{LQ} and R, can be understood as

1. A large R means that the system is being stabilized with less weighted energy, *i.e.* "expensive control strategy", by forcing a higher penalization of the control signal. Also, the response to changes in the output will be slower/more cautious.

2. A large Q_{LQ} means that one is trying to stabilize the system with fewer changes in states. Therefore, by forcing the noise to be small, the response will be faster to changes.

A Kalman filter can be used to determine the observer gain vector L so that the estimating error converges to zero. From this point forward, the estimate will provide the feedback, rather than the state, which is now considered to be unavailable for the measure.

Adjusting the parameters Q_{LQ} and R does not eliminate the steady-state error. For that purpose, one can include the integral effect. As shown in figure 2.4, the integral of the tracking error was added by forcing an integrator in series with the process.

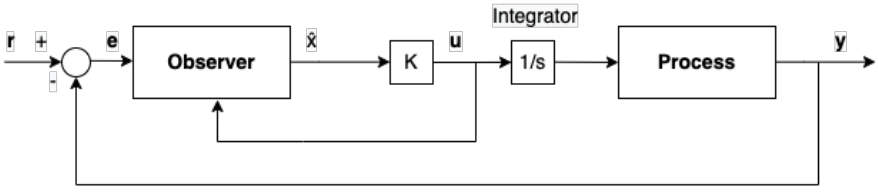


Figure 2.4: Stabilizing LQG Controller $\frac{S^0}{R^0}$ with Integral Effect.

The initial stabilizing controller $\frac{S^0(s)}{R^0(s)}$ is then represented as an LQG controller using an equivalent transfer function. The controller connected to the YK parameterization is then calculated by inserting the result into (2.5). The controller will then be able to handle complex cases such as unstable plants with non-minimum phase roots.

2.2.5 State-Space Approach: Q-Augmented LQG/LQR Controller

The next step is to deduce a state-space approach on the parameterization of all feedback controllers capable of stabilizing an LTI system, with a given LQG/LQR controller, as proven by Hespanha [2].

Suppose that instead of the classic LQG/LQR output feedback controller demonstrated in (2.23), the controller is now of the form

$$\dot{\hat{x}} = (A - LC)\hat{x} + Bu + Ly, \quad u = -k\hat{x} + v, \tag{2.28}$$

in which $v \in \mathbb{R}^k$ is the output of an asymptotically stable system driven by the output estimation error (2.29)

$$\tilde{y} := y - C\hat{x} \in \mathbb{R}^m. \tag{2.29}$$

The Q system is of the form

$$\dot{x}_Q = A_Q x_Q + B_Q \tilde{y}, \quad v = C_Q x_Q + D_Q \tilde{y}, \quad \tilde{y} \in \mathbb{R}^n, v \in \mathbb{R}^k, \tag{2.30}$$

with A_Q as a stability matrix.

From (2.28) it is possible to infer the negative-feedback control architecture as

$$\dot{\hat{x}} = (A - LC - BK)\hat{x} + Ly + Bv, \quad u = -K\hat{x} + v, \quad \tilde{y} = -C\hat{x} + y. \tag{2.31}$$

The controller is known as the Q-augmented LQG/LQR controller. If the transfer function is equal to zero, it is possible to recover the original LQG/LQR controller.

The state-space realization of the Q-augmented LQG/LQR controller can be defined from (2.29), and (2.30)-(2.31) as

$$\begin{aligned} \begin{bmatrix} \dot{\hat{x}} \\ \dot{x}_Q \end{bmatrix} &= \begin{bmatrix} A - LC - BK - BD_Q C & BC_Q \\ -B_Q C & A_Q \end{bmatrix} \begin{bmatrix} \hat{x} \\ x_Q \end{bmatrix} + \begin{bmatrix} L + BD_Q \\ B_Q \end{bmatrix} y, \\ u &= \begin{bmatrix} -K - D_Q C & C_Q \end{bmatrix} \begin{bmatrix} \hat{x} \\ x_Q \end{bmatrix} + D_Q y. \end{aligned} \tag{2.32}$$

As illustrated in figure 2.5, the Q-augmented LQG/LQR controller can include a vector $\omega(t)$ of exogenous inputs, composed by a reference signal $r(t)$, a measurement noise $n(t)$, and a disturbance signal $d(t)$. This controller also has a vector $z(t)$ of controlled outputs that contain the process output, the tracking error, and the control input.

Mind that the Q-augmented controller produces the same asymptotic closed-loop behavior as the original controller without a reference signal, measurement noise, or disturbances, even though different

transients and closed-loop transfer functions may result from the process.

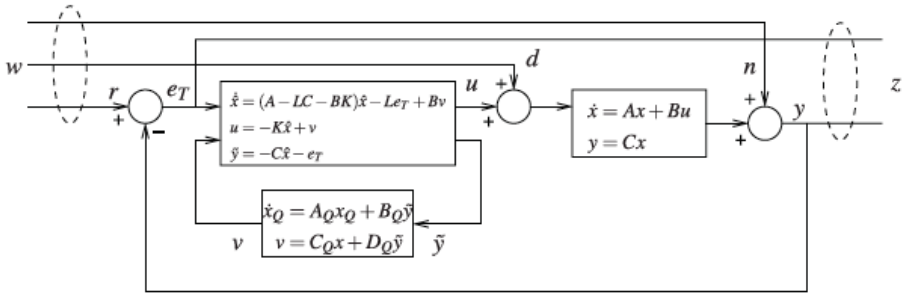


Figure 2.5: Q-augmented LQG/LQR controller [2].

We will now prove that the controller defined by (2.30)-(2.31) stabilizes the original CLTI.

The state estimation error ($e := x - \hat{x}$), without noise and disturbance, converges to zero for any input u. Analysing the architecture of the system interconnection between the process CLTI and (2.31) in figure 2.5 and given that the output estimation error (2.29) can be rewritten as $\tilde{y} := C(x - \hat{x})$, it is possible to infer that $\tilde{y}$ converges to zero.

If the input $\tilde{y}$ to (2.30) converges to zero, the output v converges to zero. Thus, the controller (2.30)-(2.31) makes the closed-loop system asymptotically stable for every stability matrix A_Q, since all signals converge to zero.

Now we will analyse the properties of the closed-loop transfer functions that result from the Q-augmented controller illustrated in figure 2.5.

The state-space closed-loop dynamics are of the form

$$\dot{\bar{x}} = \bar{A}\bar{x} + \bar{B}\begin{bmatrix}\omega\\ v\end{bmatrix}, \quad \begin{bmatrix}z\\ \tilde{y}\end{bmatrix} = \bar{C}\bar{x} + \bar{D}\begin{bmatrix}\omega\\ v\end{bmatrix}, \tag{2.33}$$

$$\dot{x}_Q = A_Q x_Q + B_Q\tilde{y}, \quad v = C_Q x_Q + D_Q\tilde{y}, \tag{2.34}$$

with the states x of the process and $\hat{x}$ of the state estimated, assembled in a single column vector $\bar{x}$.

Consider a transfer function of (2.33) as an LTI system with input vector $\begin{bmatrix}\omega' & v'\end{bmatrix}'$ and output vector $\begin{bmatrix}z' & \tilde{y}'\end{bmatrix}'$ written as

$$\begin{bmatrix}\hat{z}\\ \hat{y}\end{bmatrix} = \begin{bmatrix}\hat{P}_{z\omega}(s) & \hat{P}_{zv}(s)\\ \hat{P}_{\tilde{y}\omega}(s) & 0\end{bmatrix}\begin{bmatrix}\hat{\omega}\\ \hat{v}\end{bmatrix}, \quad \begin{bmatrix}\hat{P}_{z\omega}(s) & \hat{P}_{zv}(s)\\ \hat{P}_{\tilde{y}\omega}(s) & 0\end{bmatrix} := \bar{C}(sI - \bar{A})^{-1}\bar{B} + \bar{D}, \tag{2.35}$$

in which $\hat{z}$, $\hat{y}$, $\hat{\omega}$ and $\hat{v}$ are the Laplace transforms of z, $\tilde{y}$, ω, and v. All the transfer matrices represented by $\hat{P}_{z\omega}(s)$, $\hat{P}_{zv}(s)$ and $\hat{P}_{\tilde{y}\omega}(s)$ have Bounded Input, Bounded Output (BIBO), so are said to be BIBO

stable. Note that the transfer function from ω to z is the same as the one obtained with the original LQG/LQR controller. Moreover, the bottom right corner transfer matrix of (2.35) represents the transfer matrix from v to $\tilde{y}$, previously proven to be zero.

The closed-loop transfer function matrix from any exogenous input signal to a given controlled output, $\hat{H}(s)$, can be expressed as

$$\hat{H}(s) = \hat{H}_0(s) + \hat{L}(s)\hat{Q}(s)\hat{R}(s). \tag{2.36}$$

Here, $\hat{Q}(s) := C_Q(sI - A_Q)^{-1} + D_Q$ is the transfer function of (2.30) and is BIBO stable, as well as the transfer matrices $\hat{H}_0(s)$, $\hat{L}(s)$, $\hat{R}(s)$. It is clear that different inputs will lead to different transfer matrices $\hat{H}_0(s)$, $\hat{L}(s)$, $\hat{R}(s)$, but will still be affine in $\hat{Q}(s)$.

Finally, there are matrices A_Q, B_Q, C_Q and D_Q, with a stability matrix A_Q, for every controller transfer function $\hat{C}(s)$ that asymptotically stabilizes (CLTI), and the controller (2.30)-(2.31) is a realization of $\hat{C}(s)$, as stated in [2].

From these properties, it is valid to state that one can obtain every controller that stabilizes an LTI process and every stable closed-loop transfer function matrix with a Q-augmented controller of any given classic LQG/LQR controller, keeping $\hat{Q}(s)$ in the range of all BIBO stable transfer matrices.

In conclusion, one can obtain a parameterization of all stabilizing controllers for the process defined in (2.22) provided by the controller (2.30)-(2.31), as well as the closed-loop transfer function matrix $\hat{H}(s)$ that provides the parameterization of all stable closed-loop transfer matrices for the CLTI system.

2.3 Q Design

This section describes a control design method based on the Q parameterization and numerical optimization of parameter Q, as shown in [2].

2.3.1 Control Objectives

The goal of Q design is to enhance the performance of a controller unable to fulfill all the required specifications by augmenting it, Q-augmented LQG/LQR controller (2.32).

As described in section 2.2.5, the Q-augmented controller is based on the original LQG/LQR controller architecture with the control structure illustrated in figure 2.5. The search for the most appropriate Q parameter can be done through methods of numerical optimization.

The control objectives are met through closed-loop specifications that make the designed control system achieve the desirable performance. Note that Q design cannot directly handle gain/phase margins and open-loop gain specifications. A stable prefilter could be applied to the reference r on the

tracking error

$$e_T := r - y, \tag{2.37}$$

where y is the measured output.

We will now go through some classical closed-loop control specifications for the time domain, and the input and the outputs, before discussing Q design. For that, it is useful to rewrite the vectors ω and z, previously defined in section 2.2.5, as

$$\omega := \begin{bmatrix} r \\ d \\ n \end{bmatrix}, \quad z := \begin{bmatrix} y \\ e \\ u \end{bmatrix}.$$

The time domain specifications influence the response of the closed-loop system to a given exogenous input. One should consider multiple specifications that refer to different inputs and controlled outputs. However, the set of time domain specifications generally include:

1. Norm bounds.

   There are typically considered four different known bounds for a given constant value $c > 0$. The L_1 *norm bound*

   $$\int_0^\infty \|\bar{z}(t)\|\, dt \leq c, \tag{2.38}$$

   the L_2 *norm bound*

   $$\int_0^\infty \|\bar{z}(t)\|^2\, dt \leq c, \tag{2.39}$$

   and the L_∞ *norm bound*

   $$\|\bar{z}(t)\| \leq c\,, \forall_t \geq 0, \tag{2.40}$$

   where the vector $\bar{z}(t)$ may contain one or more entries of vector $z(t)$ of controlled outputs, for a given test input $w(t)$, $t > 0$.

2. Interval bounds.

   These specifications are used to impose conditions such as undershoot, overshoot and settling times for a step response, when the test input is selected to be a unit step. The interval bounds guarantee that for a given exogenous input signal $w(t)$, the correspondent *i*-th entry to the controlled output $z_i(t)$, satisfies the condition

   $$s_m(t) \leq z_i(t) \leq s_M(t), \quad \forall_t \geq 0, \tag{2.41}$$

   in which $s_m(t)$ and $s_M(t)$ correspond to chosen time functions. This condition can be used to assure the control signal does not overpass the safe ranges for a given exogenous input, $w(t)$.

As for the input-output specifications, they refer to the goal properties of different closed-loop transfer functions between different entries of ω and z. These specifications classically include:

1. Frequency domain.

   Consider a transfer function that maps a vector $\overline{\omega}$, with one or more entries of ω, to a vector $\overline{z}$, with one or more entries of z, satisfying

   $$\|\hat{H}(j\omega)\| \leq \ell(\omega), \quad \forall\omega \in [\omega_m, \omega_M]. \tag{2.42}$$

   For example, when $\omega_m = 0$, $\omega_m = \infty$ and $\ell(w) = \gamma$, $\forall w \geq 0$ it is possible to guarantee, that for every $\overline{\omega}(t)$ with $t \geq 0$,

   $$\left(\int_0^\infty \|\overline{z}(t)\|^2\, dt\right)^{1/2} \leq \gamma \left(\int_0^\infty \|\overline{\omega}(t)\|^2\, dt\right)^{1/2},$$

   when other entries of vector ω are zero and the closed-loop has zero initial conditions, as shown in [2].Thus, the $H - infinity$ norm, or *Root Mean Square (RMS) gain*, from input $\bar{\omega}$ to output $\bar{z}$ of the system is smaller than γ.

2. Impulse response.

   Consider an impulse response $\overline{h}(t)$ from a vector $\overline{\omega}$ to a vector $\overline{z}$. For a given constant ρ, this impulse response should comply to the inequality

   $$\int_0^\infty \|\overline{h}(t)\|\, dt \leq \rho.$$

   For every $\overline{\omega}(t)$, $t \geq 0$, and when all other entries of ω are zero and the closed-loop has zero initial conditions, the specification above guarantees the following

   $$|\overline{z}(t)| \leq \rho \sup_{\tau \geq 0} \|\overline{\omega}(\tau)\|, \quad \forall t \geq 0.$$

   Having said this, it is possible to affirm that the closed-loop system has L_1 *norm*, or *peak gain*, from $\overline{\omega}$ smaller than ρ.

In addition, it is important to note that exist numerous control specifications that can be applied to the Q design method. More control specifications examples can be found in *e.g.* Chapter 3 of [20].

2.3.2 Feasibility Problem

By definition, a feasibility problem is a problem in which the goal is to determine if a given set of constraints is feasible, that is, they can be satisfied by some controller. A feasibility problem for Q design is formulated as

$$\begin{array}{rcll} \text{find} & \hat{Q}(s) & \text{BIBO stable} & \\ \text{such that} & \hat{H}_0(s) + \hat{L}(s)\hat{Q}(s)\hat{R}(s) & \text{satisfies} & \mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_k, \end{array} \tag{2.43}$$

where $\hat{H}_0(s)$, $\hat{L}(s)$, and $\hat{R}(s)$ are the transfer matrices that define the closed-loop transfer function expressed in (2.36). As stated in [2], the Q Design method is about finding a Q system that is capable of meeting all the specifications defined in the feasibility problem (2.43) and then use this system to define the Q-augmented LQG/LQR controller, given a family $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_k$ of time domain and input-output closed-loop specifications. To solve (2.43) it is useful that all specifications are convex.

2.3.3 Finite Dimension Approximation

One of the biggest obstacles in Q Design is that it is necessary to search over an infinite-dimensional set of all stable transfer functions Q. However, as proven in [20], the *Ritz approximation* can be used to solve infinite-dimensional optimization problems by converting them into finite-dimensional subsets.

The steps to achieve the Ritz approximation are:

1. Select a sequence of $k \times m$ BIBO stable transfer functions that is *complete* such as

$$\hat{Q}_1(s), \hat{Q}_2(s), ..., \hat{Q}_i(s), ...,$$

   A *complete* Q sequence [2] is a set of all BIBO stable transfer functions $\hat{Q}(s)$, for that there is a *finite* linear combination of $\hat{Q}_i(s)$ that is arbitrarily close to the original $\hat{Q}(s)$. In the case of the Multiple-Input Single-Output (MISO) system, such linear combination is obtained by selecting all entries of each $\hat{Q}_i(s)$ to be equal to zero but one, defined as

$$\left(\frac{\alpha}{s+\alpha}\right)^{\ell}, \quad \alpha_i \in \mathbb{R}, \tag{2.44}$$

   for $\ell \geq 0$ and a chosen fixed constant $\alpha > 0$. In the SISO case, instead of a set of matrices there is only a transfer function of the form (2.44) for each entry of $\hat{Q}_i(s)$.

2. Restrict $\hat{Q}(s)$ to be

$$\hat{Q}(s) := \sum_{i=1}^{K} \alpha_i \hat{Q}_i(s), \qquad \alpha_i \in \mathbb{R}, \tag{2.45}$$

so that the search is limited to linear combinations of the first K matrices in the sequence.

3. Write the general closed-loop transfer function (2.36) as

$$\hat{H}(s) = \hat{H}_0(s) + \sum_{i=1}^{K} \alpha_i \hat{H}_i(s), \qquad \hat{H}_i(s) := \hat{L}(s)\hat{Q}_i(s)\hat{R}(s). \tag{2.46}$$

4. Define the feasibility problem as

$$\begin{aligned} &\text{find} && \alpha_1, \alpha_2, ..., \alpha_N && \in \mathbb{R} \\ &\text{such that} && \hat{H}_0(s) + \sum_{i=1}^{K} \alpha_i \hat{H}_i(s) && \text{satisfies} \quad \mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_k \end{aligned} \tag{2.47}$$

5. Use the obtained Q system from the feasible problem (2.47) to compute the Q-augmented LQG/LQR controller (2.30)-(2.31). However, if the problem is not feasible one can try to make it feasible by increasing the number of iterations N until it becomes inadmissible to compute the numerical optimization or the order of the Q-augmented controller becomes unreasonable. In that case, the problem might not have a viable solution for the determined specifications.

For an example of the application of the Ritz Approximation in Q design with MATLAB and numerical optimization using the CVX toolbox [38], one can consult Lecture 25 of [2].

3

Learning Algorithm

Contents

This section defines the two main algorithms for this work. The objective is to obtain the optimal parameter ϕ to compute the YK transfer function Q, using a general class of associative RL algorithms containing stochastic units called Episodic REINFORCE. Nonetheless, in order to compare and validate the results achieved through Episodic REINFORCE, an unconstrained optimization problem function, *Fminunc*, is used.

3.1 Unconstrained Optimization Problem

The optimal solution for a problem can be achieved through a variety of optimization algorithms and solvers. In most cases, these solvers use one or more of these optimization techniques to discover the ideal solution to an optimization problem.

Fminunc is a MATLAB function from the Optimization toolbox that uses a quasi-Newton method to solve an unconstrained optimization problem. This solver finds the minimum of an unconstrained multivariable function specified by

$$\min_{x} \quad f(x) \ , \tag{3.1}$$

where x is a vector or a matrix and the objective function of the problem $f(x)$ returns a scalar. The solution is found by iteratively approximating the objective function's Hessian matrix of second derivatives using gradient information. Since the quasi-Newton method does not need to calculate and invert the Hessian at each iteration, it is substantially less computationally expensive than the Newton method.

The solver starts at the point $x0$ and tries to find a local minimum x of the function fun. This function is minimized with a set of optimization $options$ shown in [39]. *Fminunc* returns the value of the objective function $fval$ at the solution, the $exitflag$ that describes the exit condition of the solver, and a structure output that contains information about the optimization process. There are six different exit conditions mentioned in [40]. Moreover, can return the gradient of the function at the point x with $grad$, as well as the estimated hessian with $hessian$.

3.2 Episodic REINFORCE

The form of REINFORCE algorithms can be described by its name since it is an acronym for "REward Increment = Nonnegative Factor x Offset Reinforcement x Characteristic Eligibility". Thus, any algorithm that is of that form is a REINFORCE algorithm. This class of learning algorithms are defined in detail in [6]. The algorithm used for learning in this work is an extension of the class of REINFORCE algorithms called Episodic REINFORCE, mainly used for learning tasks that have a temporal credit-assignment component, known as a (Temporal) Credit Assignment Problem (CAP). Theorems on REINFORCE and Episodic REINFORCE are also proven in [6].

In Episodic REINFORCE the learning is performed episode-by-episode *i.e.* the system states are reset at the end of each policy evaluation and the stochasticity of the policy π(y,ϕ) is on the parameters ϕ, not the outputs of the system y. For this work, it is used a specific update derived in [1] - from the update that appears in [6] for learning the mean of a Gaussian element - that learns a vector of parameters with identical noise and learning rate.

From this point on, the notation is as follows: $\phi^{i'}$ are the actual parameters used on trial i, b is a cost baseline, J($\phi^{i'}$) is the cost associated with the policy parameters $\phi^{i'}$, and g($\phi^{i'}$) is the probability of using parameters $\phi^{i'}$ in trial i. The REINFORCE update

$$\phi^{i+1} = \phi^{i} - \eta(J(\phi^{i'}) - b))\frac{\partial}{\partial \phi^{i'}} ln(g(\phi^{i'})), \tag{3.2}$$

is formulated for cost instead of reward and the learning rate η is the same for the vector of parameters ϕ. In this case, the eligibility $\frac{\partial}{\partial \phi^{i'}} ln(g(\phi^{i'}))$ can be defined as

$$\frac{\partial}{\partial \phi^{i'}} ln(g(\phi^{i'})) \propto (\phi^{i'} - \phi^{i}), \tag{3.3}$$

in which $\phi^{i'} = \phi^{i} + \phi^{p_i}$, and considering that g($\phi^{i'}$) is a multivariate Gaussian distribution with mean ϕ^{i}, that has independent noise on each element with covariance σ^2.

Thus, it is possible to formulate the update

$$\phi^{i+1} = \phi^{i} - \eta(J(\phi^{i} + \phi^{p_i}) - J(\phi^{i}))\phi^{p_i}. \tag{3.4}$$

While in literature it is common to use an average baseline [1], we employed a second policy evaluation in order to learn in fewer iterations at the expense of having to carry out two evaluations per update. First, the policy given by ϕ^{i} is executed, in iteration i, to obtain the baseline

$$b = J(\hat{Q}(\phi^{0})).$$

Then, the policy is perturbed by a small value ϕ^{p_i}, and that perturbed policy is executed. The generated white noise follows the standard normal distribution, with Probability Density Function (PDF) of the form

$$\frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}, \tag{3.5}$$

in which x is a generated random real variable, with mean $\mu = 0$ and constant variance $\sigma = 1$.

The introduction of small white noise in the algorithm adds variability and randomness. This can prevent the agent from prematurely converging to local minima. In addition, stochastic strategies encourage exploration of different courses of action and possibly find better strategies than a deterministic strategy.

Finally, the difference in performance is calculated, and the policy update (3.4) is computed. The system is run for N iterations until it converges to an optimal value and the cost function is minimized.

3.2.1 Algorithm Convergence

Even when REINFORCE is successful, the convergence to a local minimum is very slow. It has been discovered that the main algorithm in this study, Episodic REINFORCE, is particularly slow, but this is also not unexpected given that it carries out temporal credit-assignment by effectively dispersing credit or punishment over all previous times [6].

Analytical calculation of the convergence rate of stochastic algorithms such as the Episodic REINFORCE can be difficult, since it relies on a series of random factors, and the probability that the algorithm converges to a local maxima or minima depends on the decision made for the reinforcement baseline.

The use of a variation of the conjugate gradient method was one of the possible solutions implemented to make the convergence faster. The gradient conjugate method is used for solving linear systems and function optimization, *i.e.* finding the minimum of a quadratic function. The literature proposes several definitions of the conjugate gradient method for minimizing quadratic functions, *e.g.* [41].

The method consists of a sequence of conjugate directions that are designed to search the solution space more efficiently. Here, we perform the Episodic REINFORCE update (3.4) every two iterations, and at the third, we use the conjugate gradient method to compute the next search direction using the linear combination of the previous search direction, as illustrated in figure 3.1. This helps minimize oscillations and speeds convergence by ensuring the new search direction is conjugate with the prior direction. Up until the defined stopping criterion is met, this process is repeated iteratively.

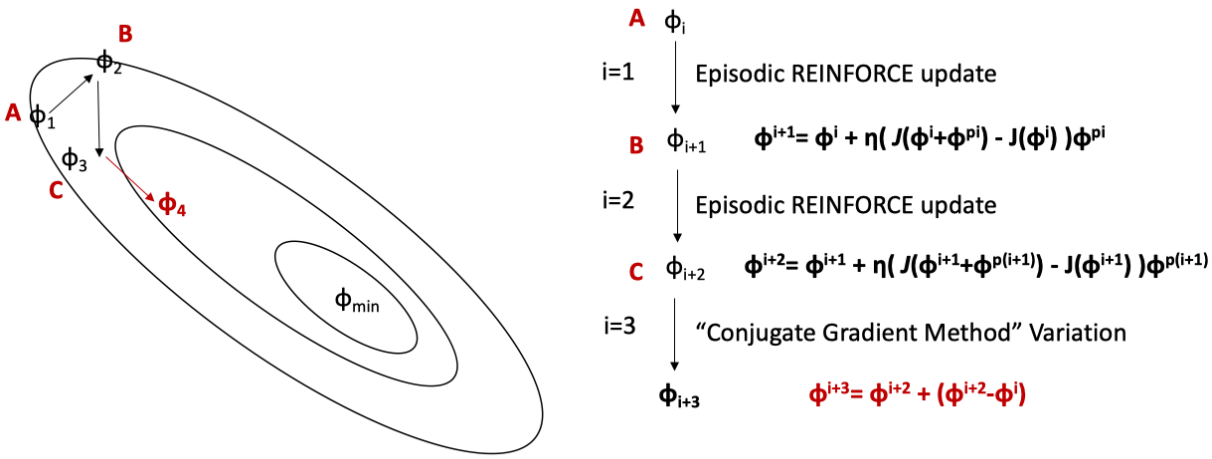


Figure 3.1: Diagram of a variation of the conjugate gradient method implemented for convergence of the RL algorithm.

In figure 3.2 is depicted the convergence comparison between the algorithm with the conjugate gradient technique variation, and the vanilla method. The initial conditions of the optimization problem are the same in this example; the only distinction is in the convergence approaches. Using the regular Episodic REINFORCE update, it can be deduced that the highest cost decrease occurs 200 iterations later. Although both algorithms converge to the same value, the minimum cost is found at iteration 907 using the conjugate gradient technique, but only at iteration 1418 when utilizing the regular update. As a result, it has been proven to boost speed by approximately 64%.

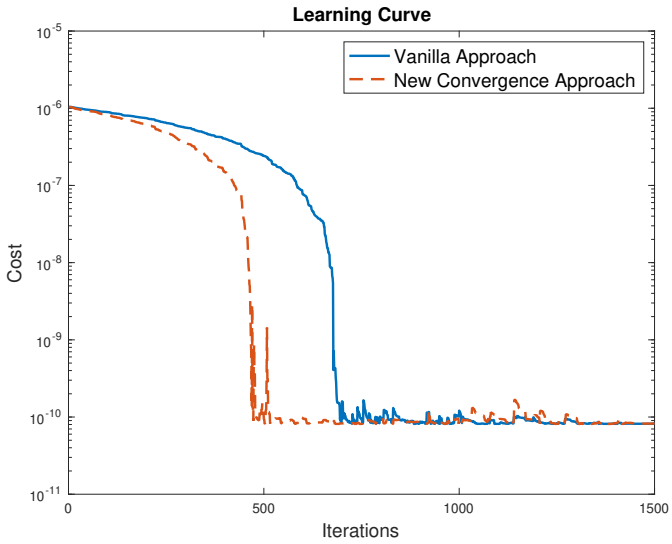


Figure 3.2: Comparison of the convergence of the algorithm with and without the conjugate gradient method modification.

The step size is also an important factor to be considered for convergence speed. The learning rate is the hyperparameter that controls the size of the update on each iteration of the algorithm.

Finding the proper step size for a given optimization problem is crucial for ensuring convergence of the algorithm, given that if the step size is too large, the process is faster but the algorithm may pass the minimum of the loss function and fail to converge. Instead, if the step size is too small, the algorithm might take a long time to converge or get stuck in a local minimum. This process can be done through trial and error or with known techniques such as adaptive learning rate methods. A variety of step size adaptation methods for stochastic learning have been presented in the literature (see [42], [43]). In this work we concentrate on adapting the learning rate η based on percentual variation of cost

$$\Delta J(\%) = \frac{J(i) - J(i-1)}{J(i-1)} \times 100, \tag{3.6}$$

starting from an arbitrarily large value of step size and adapting it based on optimization performance.

For that, when the cost increases a defined percentual value instead of decreasing, the learning rate

is altered and the algorithm returns to the previous value of the optimization parameter and cost.

In the last iterations, usually the final 5%, it is important to reduce the learning rate significantly to avoid surpassing the minimum value. Thus, during these last iterations, the learning rate progressively decreases proportionally to the number of iterations left.

The step gain is also normalized based on the value of cost before it is multiplied by the search direction, *i.e.* the learning rate in the Episodic REINFORCE policy update (3.4) is divided by that iteration's cost $J(\phi^i)$, making the step gain $\eta/J(\phi^i)$. Hence, in an ideal example, if the random noise constant ϕ^{pi} in that iteration makes the perturbed policy zero, the search direction is $J(\phi^i)\phi^{pi}$. In these circumstances, the next iteration's value of ϕ will represent cost zero as $J(\phi^i + \phi^{pi})$.

4

Rocket Landing

Contents

In this chapter, the nonlinear model of a rocket is derived, linearized, and validated through simulations. The goal is to successfully land the rocket. The attitude and altitude state-space control systems are identified using the rocket model. Furthermore, we design the baseline attitude and altitude LQG controllers and run closed-loop simulations.

4.1 Thrust Vector Control

Let us clarify the concept of Thrust Vector Control (TVC). The thrust's direction changes in relation to the rocket's center of gravity when the nozzle is moved as depicted in figure 4.1 [3], known as the "normal" flight configuration. The thrust line is angled toward the rocket center line in the left and right illustrations at a position known as the gimbal angle. When the rocket's nozzle moves, the thrust's direction changes in relation to its center of gravity, changing the gimbal angle from zero. If the rocket is gimbaled back along the center line or deviated to the right, as opposed to the center picture, where the thrust goes through the center of gravity, a torque is generated about the center of gravity, which causes the rocket's nose to turn left or right.

To maintain a "normal" flight configuration and follow the reference trajectory, TVC will be used to control the rocket's attitude and keep the gimbal angle as close to zero as possible.

In a real-world system, the nozzle moves along three dimensions. However, the simulations in this thesis were conducted in two dimensions. The gimbal can be represented by a rotating ball joint at the rocket's lower end since the flow of the thrust is assumed to act along a single directional vector.

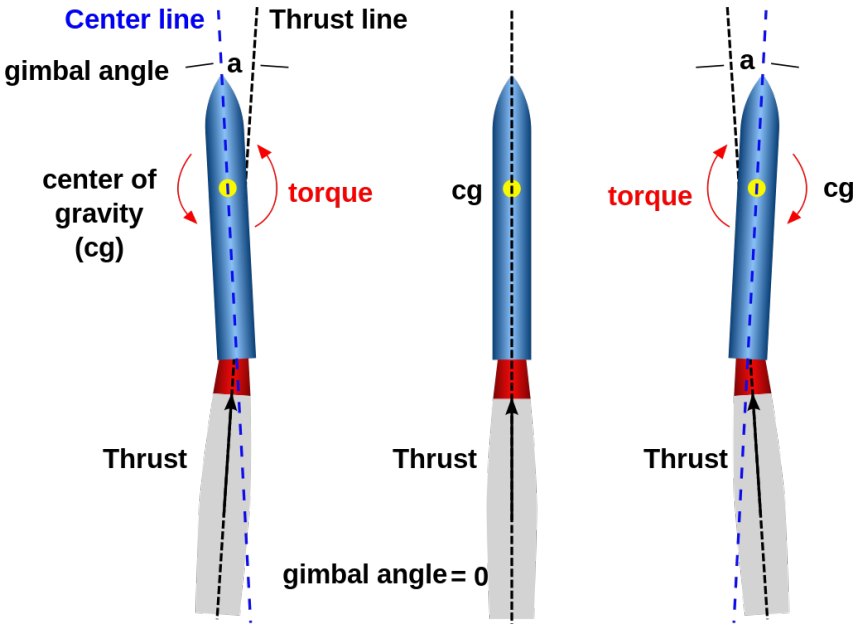
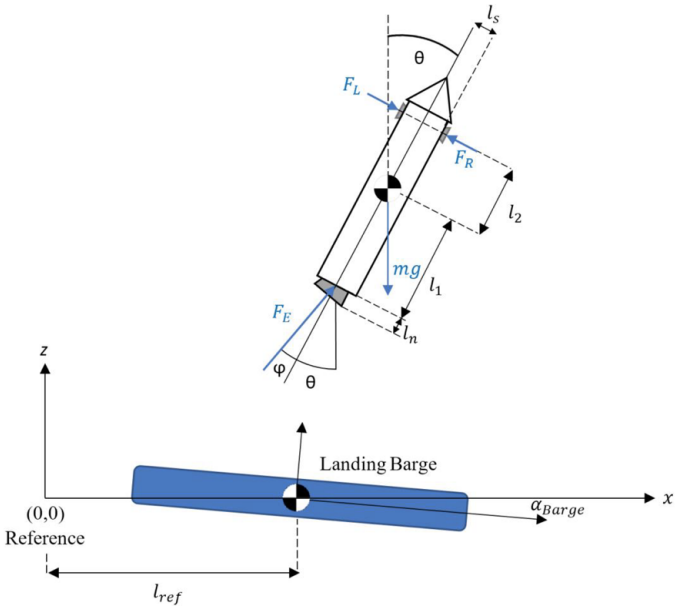


Figure 4.1: Gimbaled Thrust diagram [3].

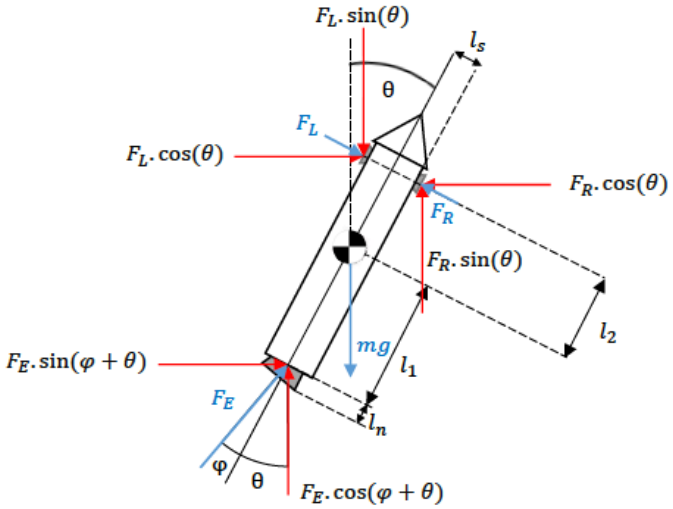
4.2 Nonlinear Model

The mathematical derivation of the rocket will be based on Newton's 3rd law of motion.

Figure 4.2(a) illustrates a simplified representation of a rocket landing and 4.2(b) depicts all the considered forces.



(a) Diagram of a rocket landing in a barge.



(b) Diagram of a rocket with all forces represented.

Figure 4.2: Rocket simplified 2D representation [4].

The notation [4] used moving forward is

- F_E: main thruster force;
- F_R: right thruster force;
- F_L: left thruster force;
- F_S = F_L - F_R : right and left thruster forces as a single input;
- θ: angle between the z axis of the plan and the longitudinal axis of the rocket;
- φ: angle between the x axis of the plan and the longitudinal axis of the rocket;
- l_1: longitudinal length between the Center of Gravity (COG) of the rocket and F_E;
- l_2: longitudinal length between the COG of the rocket and F_R, F_L;
- l_n: length of the nozzle;
- m: sum of the rocket's dry mass and fuel mass;
- x: horizontal position of the rocket;
- z: vertical position of the rocket;
- α, β: real constants.

The state of the rocket dynamics depends on the horizontal and vertical positions of the rocket (x, z) and its velocity ($\dot{x}$, $\dot{z}$), as well as the angle between the z axis and the longitudinal axis of the rocket (θ) and the angular velocity $\dot{\theta}$. This can be defined by the vector of states

$$X = \begin{bmatrix} x & \dot{x} & z & \dot{z} & \theta & \dot{\theta} \end{bmatrix}'. \tag{4.1}$$

The inputs, or main control variables of the rocket, are the main engine thrust F_E, the right and left side Nitrogen gas thrusters, that can be simplified into one input $F_S = F_R - F_L$, and the thrust angle φ. Nitrogen gas thrusters can be set to zero for simplicity even though they provide a more stable control.

The control outputs are x, z and θ.

We simplify the next equations for small angles, in which $cos(\theta) = cos(\varphi) \approx 1$, $sen(\theta) \approx \theta$, and $sen(\varphi) \approx \varphi$. Now, it is possible to infer the expressions of the translational forces with respect to the COG:

$$\begin{aligned} \ddot{x} &= \frac{F_E \cdot sen(\theta + \varphi) + F_S \cdot cos(\theta)}{m} \\ \Leftrightarrow \ddot{x} &= \frac{F_E \cdot cos(\varphi) \cdot sen(\theta) + F_E \cdot cos(\theta) \cdot sen(\varphi) + F_S \cdot cos(\theta)}{m} \\ &\approx \frac{F_E \cdot \theta + F_E \cdot \varphi + F_S}{m}, \end{aligned} \tag{4.2}$$

$$\begin{aligned} \ddot{z} &= \frac{F_E \cdot cos(\theta + \varphi) - F_S \cdot sen(\theta) - mg}{m} \\ \Leftrightarrow \ddot{z} &= \frac{F_E \cdot cos(\varphi) \cdot cos(\theta) - F_E \cdot sen(\varphi) \cdot sen(\theta) - F_S \cdot sen(\theta) - mg}{m} \\ &\approx \frac{F_E - F_E \cdot \varphi \cdot \theta - F_S \cdot \theta - mg}{m}. \end{aligned} \tag{4.3}$$

Since the shape of the rocket does not change, the moment of inertia equation can be described by

$$J_T \cdot \ddot{\varphi} = \tau, \tag{4.4}$$

in which τ represents an applied torque on a rocket to the angular acceleration, $\ddot{\varphi}$.

When the nozzle angle, φ, which typically moves in three dimensions, is different from zero, a torque is generated. However, in this work, the problem will be reduced to a two-dimensional system, which results in a unidimensional rotation. The rotation torque with respect to the COG can be computed by

$$\begin{aligned} \ddot{\theta} &= \frac{-F_E \cdot sen(\varphi)(l_1 + l_n \cdot cos(\varphi)) + F_S \cdot l_2}{J_T} \\ &\approx \frac{-F_E \cdot \varphi(l_1 + l_n) + F_S \cdot l_2}{J_T}, \end{aligned} \tag{4.5}$$

where J_T is the moment of inertia, and $\ddot{\theta}$ is the angular acceleration.

The fuel burn of the rocket is modeled by

$$\dot{m} = -\alpha(\beta \cdot F_E - F_S), \tag{4.6}$$

which is directly proportional to the thrust.

4.3 Analytical Linearization

From the nonlinear rocket model, we now deduce its analytical linearization.

If the problem is LTI, it can be written in the state-space form shown in (4.7), as previously stated, where A is the system matrix, B and C are the input and output matrices, and D is the feed-forward matrix.

$$\begin{aligned} \dot{x} &= Ax + Bu, \\ y &= Cx + Du. \end{aligned} \tag{4.7}$$

However, the aforementioned model is not linear. Therefore, it must be linearized around an equilibrium point in order to design a controller.

Consider the general nonlinear differential equation written as

$$\dot{x}(t) = f(x(t), u(t)), \tag{4.8}$$

and take the function f, that maps $\mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, an equilibrium point $\overline{x} \in \mathbb{R}^n$ and an equilibrium input $\overline{u} \in \mathbb{R}^m$, so that

$$f(\overline{x}, \overline{u}) = 0. \tag{4.9}$$

A point is considered to be in equilibrium if, given a certain equilibrium input, all changing states go to zero. We set $\ddot{x}$, $\ddot{z}$, and $\ddot{\theta}$ to zero. Hence, by solving equations (4.2)-(4.5) it is possible to obtain the equilibrium input given by

$$\overline{u} = [mg, 0, 0]. \tag{4.10}$$

Now, it is possible to perform the Jacobian linearization [44] on a nonlinear differential equation about the point $(\overline{x}, \overline{u})$.

The theory states that if one starts the simulation in the equilibrium point, $x(t_0) = \overline{x}$, and applies the equilibrium input $u(t) = \overline{u}$, the system will remain in equilibrium for all t. Nevertheless, if one starts near $\overline{x}$, there will be deviations associated, such as $\Delta_x(t)$ and $\Delta_u(t)$. Given that, it is possible to rewrite (4.8) as

$$\dot{\Delta}(t) = f(\overline{x} + \Delta_x(t), \overline{u} + \Delta_u(t)). \tag{4.11}$$

After applying the Taylor Expansion to (4.11), ignoring the high-order terms, and considering (4.9), one obtains

$$\dot{\Delta}_x(t) \approx \left.\frac{\partial f}{\partial x}\right|_{x=\overline{x}, u=\overline{u}} \Delta_x(t) + \left.\frac{\partial f}{\partial u}\right|_{x=\overline{x}, u=\overline{u}} \Delta_u(t), \tag{4.12}$$

that can be of the form

$$\dot{\Delta}_x(t) \approx A\Delta_x(t) + B\Delta_u(t), \tag{4.13}$$

with A and B being the (4.7) system matrices.

To obtain the matrices A and B analytically, the problem (4.13) is equivalent to the computation of the partial differentiation on the state equations (4.1) and the inputs, such as

$$A = \nabla_x f = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}, \quad B = \nabla_u f = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_n} \end{bmatrix}. \tag{4.14}$$

Given (4.14), the resulting matrices are

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\overline{F_E}}{m} & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{(-\overline{F_E}\cdot\overline{\varphi}-\overline{F_S})}{m} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ \frac{(\overline{\theta}+\overline{\varphi})}{m} & \frac{1}{m} & \frac{\overline{F_E}}{m} \\ 0 & 0 & 0 \\ \frac{(1-\overline{\theta}\cdot\overline{\varphi})}{m} & -\frac{\overline{\theta}}{m} & \frac{(-\overline{F_E}\cdot\overline{\theta})}{m} \\ 0 & 0 & 0 \\ \frac{(-\overline{\varphi}(l_1+l_n))}{J_T} & \frac{l_2}{J_T} & \frac{(-\overline{F_E}(l_1+l_n))}{J_T} \end{bmatrix}. \tag{4.15}$$

4.4 Model Validation: Open-loop Simulations

In this section, we perform open-loop simulations to validate the rocket's nonlinear model derived in section 4.2. For the simulations, the mass of the rocket was assumed to be constant and the Nitrogen gas thrusters F_S were set to zero for simplification.

The constant values considered for the simulations are presented in Table 4.1. The dimensions of the rocket were chosen accordingly to the first-stage rocket data of Falcon 9 [45], [46].

Table 4.1: Simulation constants.

m [Kg]	g [m/s^2]	l_1 [m]	l_2 [m]	l_n [m]	J_T [Kg m^2]
549054	9.8	60	10	1	40267

Figure 4.3 depicts the MATLAB *Simulink* diagram used, where the inputs for the function blocks "ddx" (4.2), "ddz" (4.3), and "ddtheta" (4.5) are F_E, F_S, and φ with F_S set to zero. The outputs are the states X represented in (4.1).

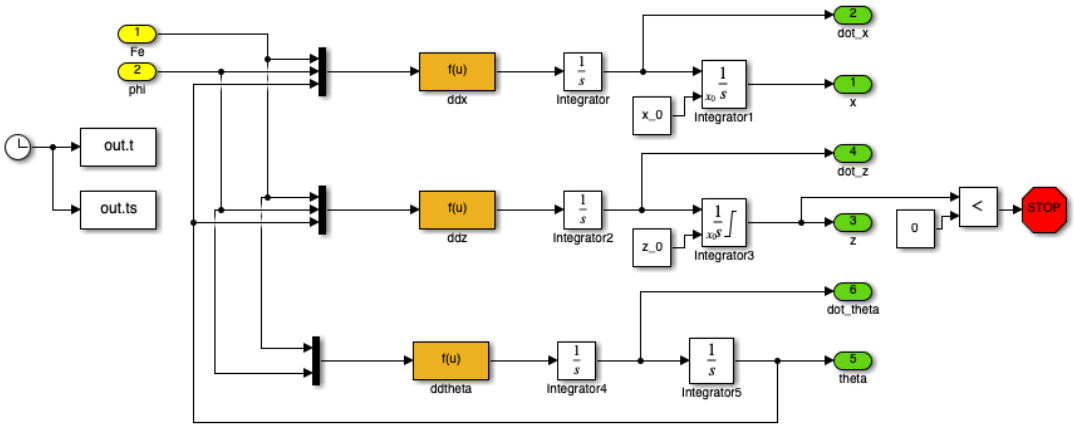


Figure 4.3: MATLAB *Simulink* model of the nonlinear rocket model.

The block "Integrator 3" in figure 4.3 also lower bounds the output of variable z to zero. The simulation automatically stops when the altitude of the rocket is smaller or equal to zero meters, indicating it reached

ground level. Also, the Integrator blocks 1 and 3 accept an external initial condition x_0 and z_0, for the states x and z of vector (4.1).

In order to validate the outputted data from the simulations it is possible to compute the equations of motion. For the altitude

$$z = z_0 + v_0 t + \frac{1}{2} a t^2, \tag{4.16}$$

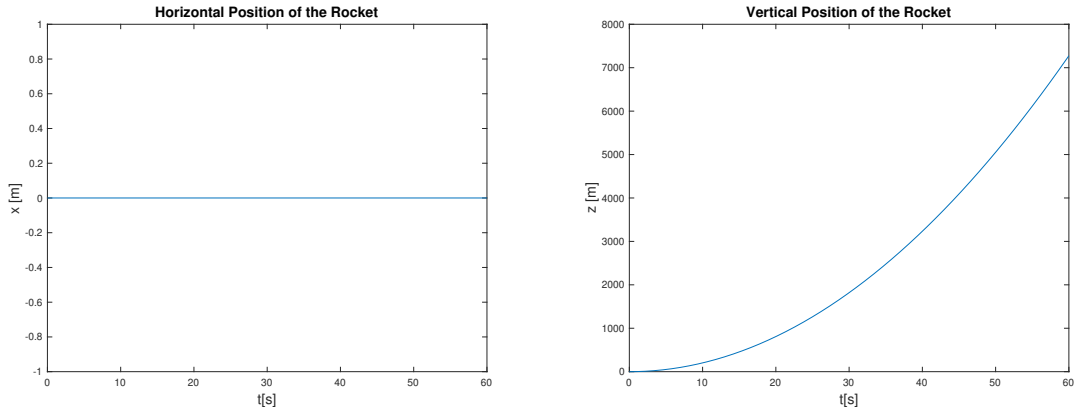
where the variable "a" is the acceleration and v_0 is the initial velocity. The attitude angle θ is obtained by

$$\theta = \theta_0 + w_0 t + \frac{\alpha t^2}{2}, \tag{4.17}$$

in which α is the angular acceleration and w_0 corresponds to the initial angular velocity.

Several simulations were run. Here, we will discuss three distinct scenarios where the main thrust force varies with respect to the equilibrium value.

Test 1



(a) Horizontal position of the rocket x.

(b) Vertical position of the rocket z.

Figure 4.4: Position and velocity of the rocket over time for test 1.

For the first test, the main thrust force is an arbitrary value larger than the value of the mass times the gravity ($F_E > m \cdot g$). Therefore, the rocket should go straight up in altitude. Also, the horizontal position remains zero, as seen in figure 4.4(a), and since the gimbal angle is zero, there is no torque and θ is zero radians. Figure 4.5 illustrates the trajectory in 3-Dimensions.

When substituting the final values in (4.16) one can conclude that after $t = 60$s with acceleration $a = -g + \frac{F_E}{m}$ and force $F_E = 7.6 \times 10^6$N the rocket would achieve an altitude z of, approximately, 7.276×10^3m, which is proven in figure 4.4(b).

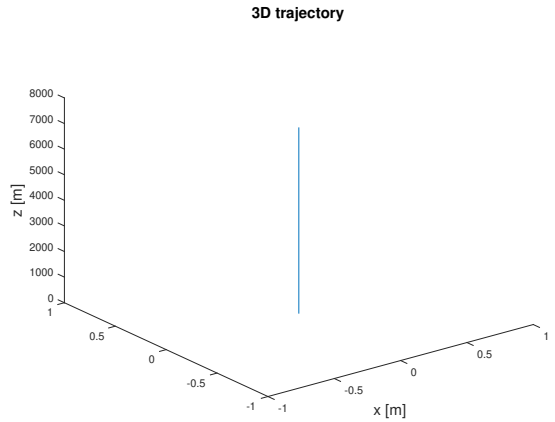
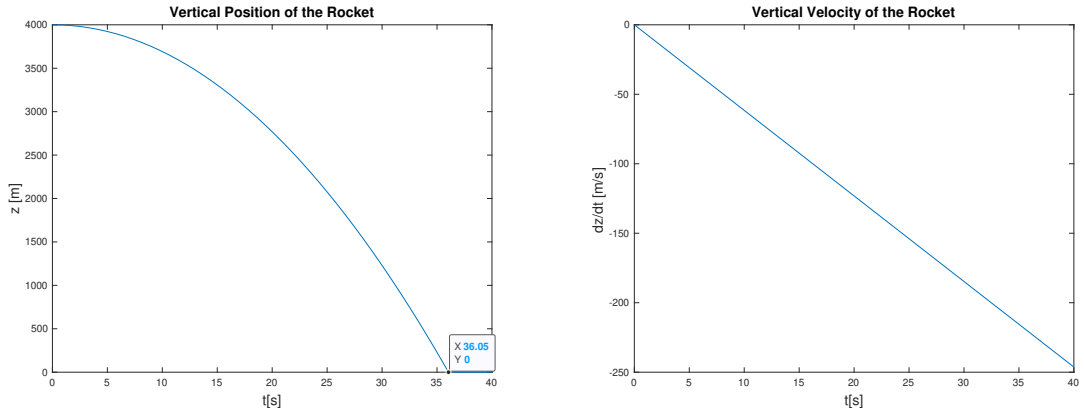


Figure 4.5: 3-Dimension trajectory for test 1, with horizontal position in the x axis, vertical position in z axis, and y set to zero.

Test 2

For this test, the main thrust force is smaller than the equilibrium value ($F_E < m \cdot g$). Thus, the rocket should not be able to go up in altitude as depicted in figure 4.6(a). The vertical velocity $\dot{z}$ should be negative as shown in figure 4.6(b). The horizontal position remains at zero, as seen in "Test 1" and figure 4.4(a). Since the gimbal angle is zero radians, there is no torque and θ will also remain zero.

To verify the outputted data, one can compute the expected time that the rocket would hit the ground, *i.e.* $z = 0$m, using equation (4.16). It is possible to infer that when free falling with an initial velocity equal to zero from 4km, the rocket would take, approximately, $t = 36.05$ seconds to be at $z = 0$m with acceleration $a = -g + \frac{F_E}{m}$ and force $F_E = 2 \times 10^6$N, which can be confirmed in figure 4.6(a). The trajectory of the rocket is fully vertical, as shown in figure 4.7.



(a) Vertical position of the rocket z.

(b) Vertical velocity of the rocket $\dot{z}$.

Figure 4.6: Position and velocity of the rocket over time for test 2.

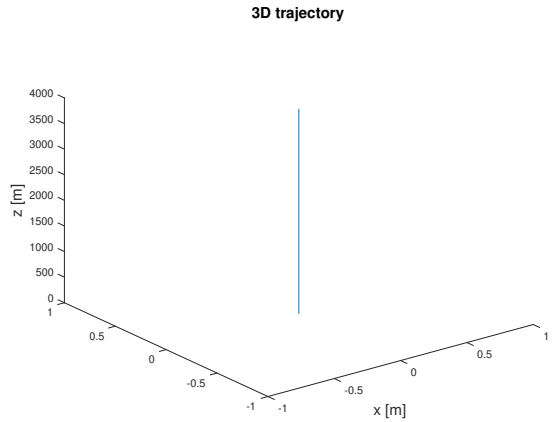


Figure 4.7: 3-Dimension trajectory for test 2, with horizontal position in the x axis, vertical position in z axis, and y set to zero.

Test 3

This test works as a “sanity check”, since the main thrust force is equal to the mass times the gravity ($F_E = m \cdot g$). Thus, the rocket should not move as proven in figure 4.8. Here, the horizontal position and θ also remain zero. The horizontal and vertical velocities remain zero over time, proving that the rocket is not moving.

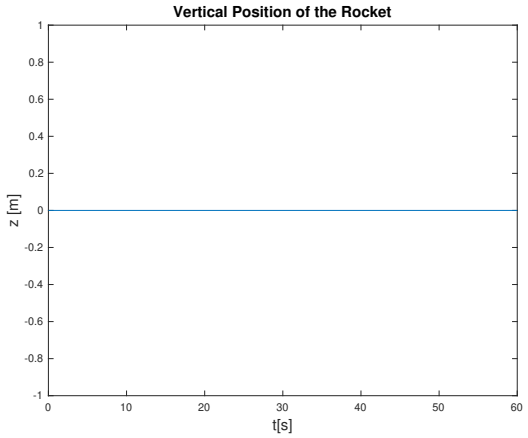
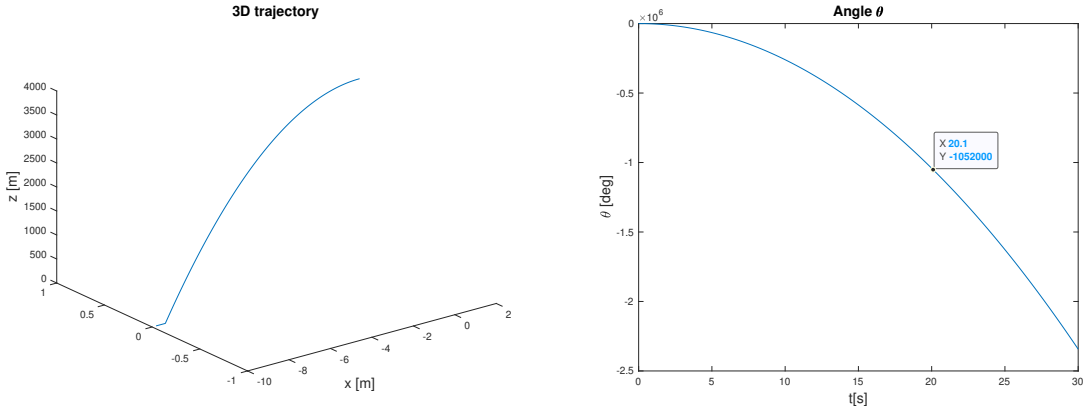


Figure 4.8: Vertical velocity of the rocket over time $\dot{z}$ for test 3.

Test 4

Lastly, the scenario in which the gimbal angle has a positive value of $\varphi = 1.7°$ and the main thrust force is constant and smaller than the value of the mass times the gravity $F_E < m \cdot g$. The gimbal angle φ is positive, so the rocket should go further to the right side of the x axis and start falling from the initial



(a) 3-Dimension trajectory.

(b) Angle between the z axis of the plan and the longitudinal axis of the rocket, θ.

Figure 4.9: 3-Dimension trajectory and output angle of the rocket over time for test 4.

altitude of 4000 meters. The 3-Dimensional trajectory of the rocket is depicted in figure 4.9(a).

The gimbal angle is $\varphi \neq 0$, therefore a negative torque is generated. The output θ obtained is shown in figure 4.9(b). To verify the attitude of the rocket, one can compute the expected angle θ at a certain time, using equation (4.17). From there, it is possible to infer that when falling from 4km, with an initial angular velocity equal to zero, the rocket would have a θ angle of, approximately, $1.052 \times 10^{6}{}^{\circ}$, at $t = 20.1$s, which can be confirmed in figure 4.9(b). Here, the angular acceleration α is in (4.5), for $F_S = 0$.

4.5 Nominal Trajectory

Next, we will define the rocket's nominal trajectory, which will serve as a reference throughout this work.

Nominal, in the context of the trajectory of a launch vehicle, is a flight in which all the vehicle aerodynamic parameters are as expected and all vehicle internal and external systems perform exactly as planned, and there are no extraneous perturbing factors, except atmospheric drag and gravity [47].

The goal is to find a nominal trajectory for the rocket, in order to reach the ground with a final velocity equal to zero.

For that, we consider a *bang-bang* control trajectory, or "on-off", where the main thrust force signal switches abruptly between two states *i.e.* completely OFF ($F_E \approx 0$N) and fully ON ($F_E = F_{E_{(max)}} = 7 \times 10^{6}$N). Here, the thrust force will have a minimum value of 5% of the maximum force value and, therefore, will not be completely OFF so it is still possible to control the gimbal angle, φ. As illustrated in figure 4.10, the moment of the switch is represented as t_{ON}. Since the desired trajectory is vertical, the

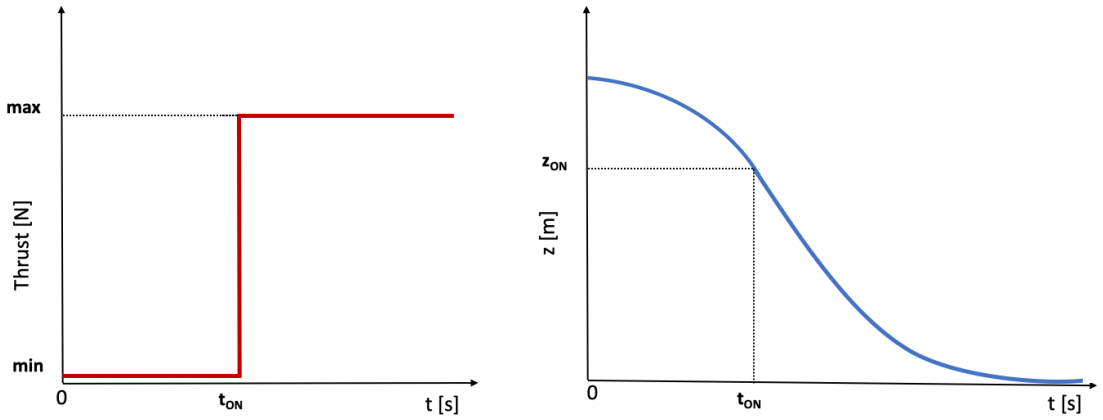


Figure 4.10: *Bang-Bang* profile.

gimbal angle is assumed to be zero.

The resulting vertical position, z, of the rocket when the thrust force depicted in figure 4.10 is applied, can be computed analytically using the equation of motion in (4.16). For the first $[0, t_{ON}]$ seconds, the acceleration that acts on the body is the sum of the gravity with the quotient of the minimum thrust force applied and the mass. Therefore, considering the initial velocity to be zero, the position z_{ON} is of the form

$$z_{ON} = z_0 + \frac{1}{2}(-g + \frac{F_E}{m})(t_{ON})^2. \tag{4.18}$$

The trajectory is influenced by different "switch" time instants, t_{ON}, in which the state changes from zero to the maximum force value. Figure 4.11 shows the open-loop simulation for different arbitrary values of t_{ON}. For a chosen time instant of $t_{ON} = 14.6$, which results in figure 4.12(a), the vertical

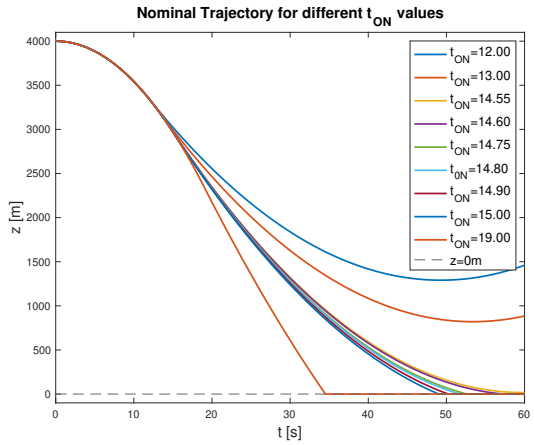
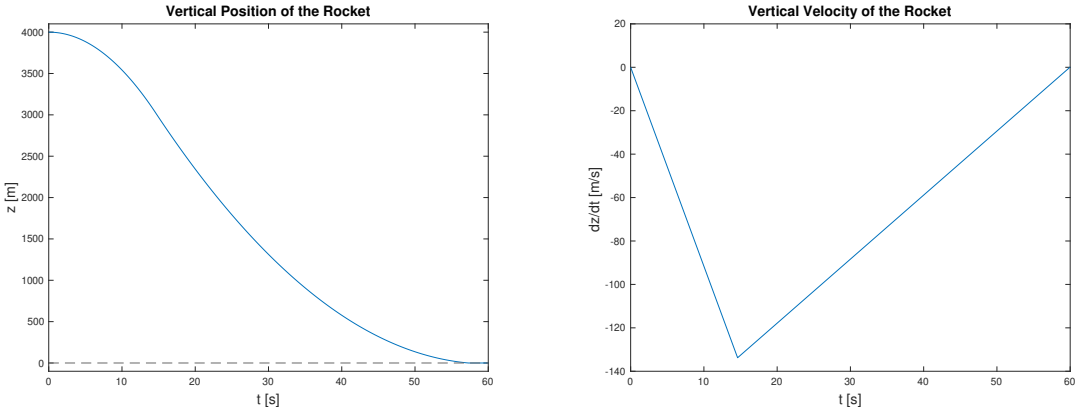


Figure 4.11: Vertical Position z of the rocket over time for different values of t_{ON}.

position z_{ON} on time instant t_{ON} is, approximately, 3023m. When analyzing the simulation results, it is

proven that the rocket is descending fast for the first t_{0N} seconds until the thrust value switches and the trajectory changes, slowly reaching zero. In figure 4.12(b) it is possible to verify that the rocket reaches the ground with a velocity of approximately zero.



(a) Vertical Position z of the rocket over time.

(b) Vertical Velocity $\dot{z}$ of the rocket over time.

Figure 4.12: Verification of the vertical position z and velocity $\dot{z}$ of the rocket over time, for $t_{ON} = 14.6$ seconds.

4.6 System Identification

The F_L and F_R forces were adjusted to zero, and the 6th-order multivariate system was reduced to a direct input-output relationship to make the identification process simpler. Therefore, two different identification processes must be carried out to build the model. The double integral effect must be eliminated before applying the identification methods to get appropriate identification results.

Using data from the open-loop simulation, the MATLAB function *ssest* is used to estimate a continuous-time state-space model of a specific order. In order to build the attitude and altitude models, one should: 1. Remove the double integral effect of $\ddot{\theta}$ and $\ddot{z}$, deriving it twice. 2. Identify the model that relates: the input gimbal angle φ with the angle θ, and the input thrust force F_E with the vertical position of the rocket z; 3. Add a double integrator to the model;

4.6.1 Attitude System

The input u of the attitude model can be determined by

$$\Delta\varphi = \varphi - \overline{\varphi}, \tag{4.19}$$

in which the gimbal angle φ is an input of the system, and the equilibrium value $\overline{\varphi}$ is zero radians as is illustrated in figure 4.13.

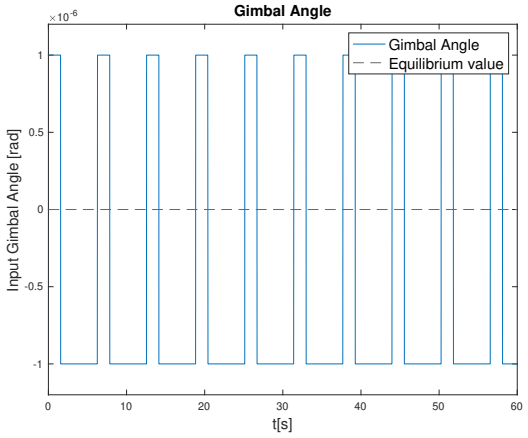


Figure 4.13: Square wave with 25% duty cycle, representing the input gimbal angle oscillating around the equilibrium value.

The output y is the second derivative of the angle θ, from the *Simulink* simulation. The rocket is "hovering" since only the gimbal angle oscillates. The thrust force input is considered constant, equal to the mass times gravity of the system.

It is possible to relate the output ($\ddot{\theta}$) and the input (gimbal angle φ) by a gain $\alpha = -8146$, with the MATLAB function *ssest*.

Analytically, equation (4.5) can be used to compute the gain as

$$\frac{-\overline{F_E} * (l_1 + l_n)}{J_T} \approx -8151. \tag{4.20}$$

The ratio between the amplitude of the signals depicted in figures 4.13 and 4.14 is

$$\frac{-8.151 \times 10{-}3}{1 \times 10{-}6} = -8151, \tag{4.21}$$

which is in accordance with the analytical gain obtained from equation (4.20).

Therefore, the fit to estimation data, of the form 100(1-Normalized Root Mean Squared Error (NRMSE)), is 96.51% and has a Mean Squared Error (MSE) of 6.44×10^{-08}. The comparison between the response of the estimated system and the measured output is illustrated in figure 4.14. As shown in the simplified block diagram of figure 4.15, the double integrator needs to be added to design the linear controller.

From that, the double integrator matrices of the attitude system are of the form

$$A = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & -8.146 \times 10^3 \end{bmatrix}, \quad D = \begin{bmatrix} 0 \end{bmatrix}. \tag{4.22}$$

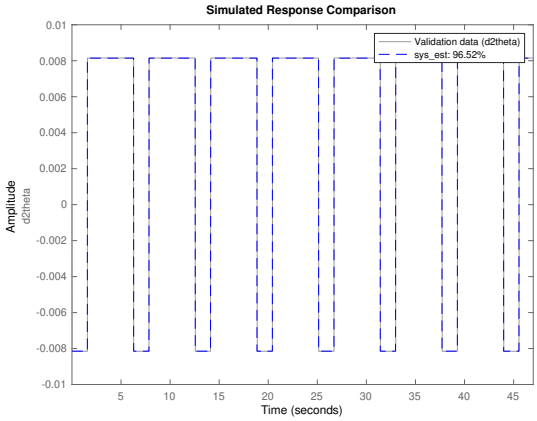


Figure 4.14: Comparison of the response of the system and the measured data.

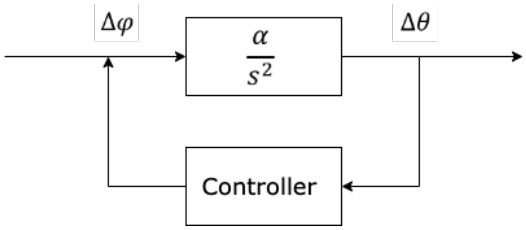


Figure 4.15: Relation between the system and the attitude controller.

The LTI problem with

$$x = \begin{bmatrix} \dot{\theta} & \theta \end{bmatrix}', \tag{4.23}$$

can now be written in the classic state-space form defined in (4.7) as

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \ddot{\theta}, \\ y = \begin{bmatrix} 0 & -8.146 \times 10^3 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \theta \end{bmatrix}. \end{cases} \tag{4.24}$$

4.6.2 Altitude System

For the altitude system identification, the input u is given by

$$\Delta F_E = F_E - \overline{F_E}, \tag{4.25}$$

where F_E is the thrust force illustrated in figure 4.16 and the equilibrium value is $\overline{F_E} = mg$ N.

In this case, the output y is the second derivative of the position z from the *Simulink* simulation. The gimbal angle is forced to zero radians, so that the trajectory remains vertical.

As previously explained, given the input signal, it is possible to relate the output $\ddot{z}$ and the thrust force

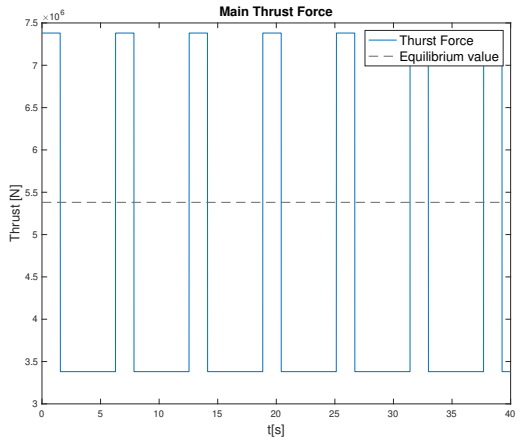


Figure 4.16: Square wave with 25% duty cycle, amplitude 2×10^6N and an offset of $\overline{F_E}$, representing the input thrust force oscillating around the equilibrium value.

F_E by a gain $\beta = 1.820 \times 10^{-6}$ obtained using the MATLAB function *ssest*.

The gain β can be computed analytically using the previously derived equation (4.3) from the nonlinear model as

$$\frac{1-\overline{\varphi}}{m} = 1.821 \times 10^{-6}. \tag{4.26}$$

The relation between the amplitude of both the amplitude of the force signal and the input signal in figure 4.16 is of the form

$$\frac{3.642}{2 \times 10^6} = 1.821 \times 10^{-6}, \tag{4.27}$$

that complies with the analytical output.

Here, the fit to estimation data is of 96.52% and the MSE is 0.128. The comparison between the estimated system and the measured output data is illustrated in figure 4.17.

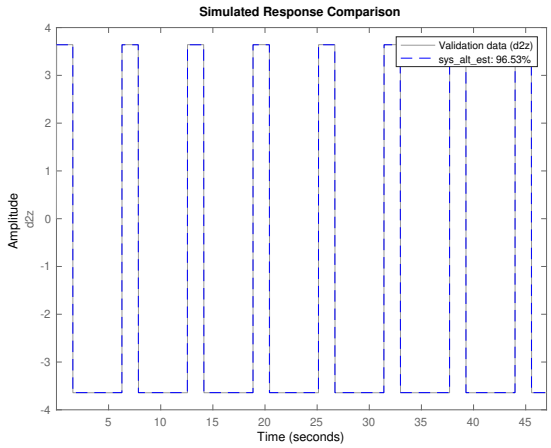


Figure 4.17: Comparison of the response of the system and the measured data.

As stated for the attitude system, it is necessary to add the double integrator in order to design the linear controller, as represented in figure 4.18.

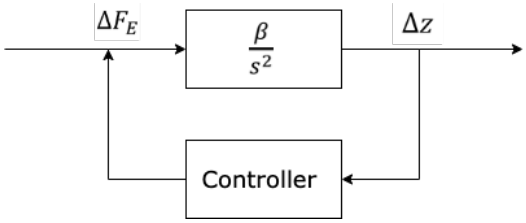


Figure 4.18: Relation between the system and the altitude controller.

From that, it is possible to infer the system

$$A_2 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad C_2 = \begin{bmatrix} 0 & 0.1820 \times 10^{-5} \end{bmatrix}, \quad D_2 = \begin{bmatrix} 0 \end{bmatrix}, \tag{4.28}$$

and write the LTI problem in the classic state-form (4.7) as

$$\begin{cases} \begin{bmatrix} \ddot{z} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{z} \\ z \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \ddot{z}, \\ y = \begin{bmatrix} 0 & 0.1820 \times 10^{-5} \end{bmatrix} \begin{bmatrix} \dot{z} \\ z \end{bmatrix}. \end{cases} \tag{4.29}$$

4.7 Baseline Controller Design

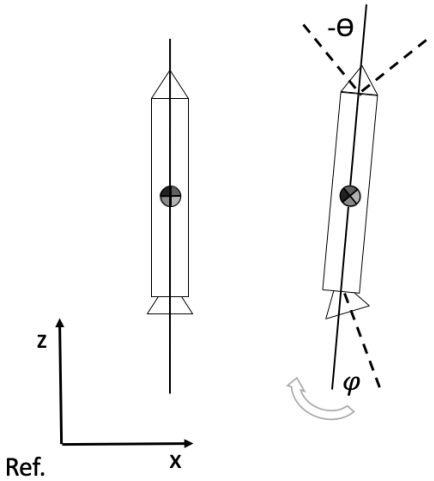


Figure 4.19: Orientation of the rocket.

It is necessary to design a baseline state feedback LQG, to control the attitude and altitude of the rocket. We will use it as an initial stabilizing controller in the YK parameterization.

As one might infer from the diagram shown in figure 4.19, if the input gimbal angle φ is positive, a

negative torque will be generated. In order to balance the angle of the rocket, the controlled angle θ must be negative.

The closed-loop *Simulink* diagram of the LQR controllers with access to all the components of the states of the state-space systems for the attitude (4.24) and altitude (4.29) of the rocket is shown in figure 4.20. To design the attitude LQR controller for the angle between the z axis of the plan and the

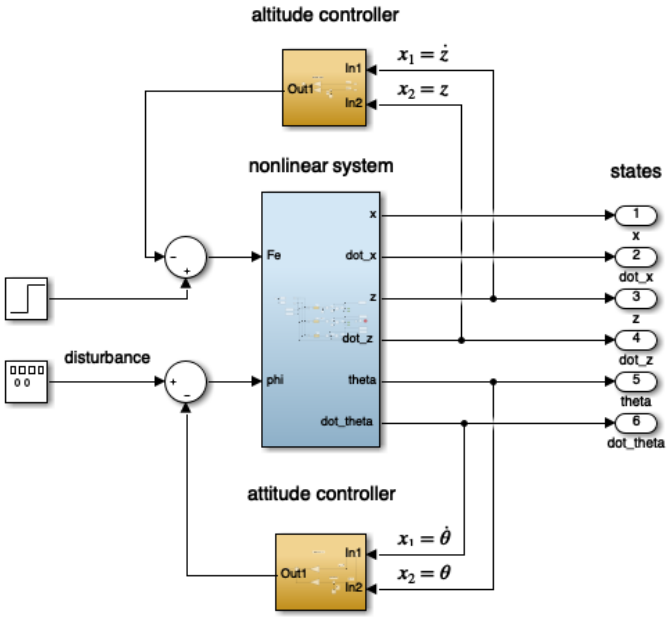


Figure 4.20: *Simulink* diagram for the LQR controllers of rocket model.

longitudinal axis of the rocket, θ, the matrix B identified for the LTI system in (4.22) must be multiplied by the output matrix C, to obtain a negative controller gain. Hence, after properly adjusting the weight matrices R and Q_{LQ}, the controller gain vector K, obtained with MATLAB function *lqr*, is

$$K_\theta = \begin{bmatrix} -0.496, & -1000 \end{bmatrix}. \tag{4.30}$$

The eigenvalues determine the evolution of the system, by analyzing whether a fixed point is stable or unstable. A fixed point, or equilibrium point, is stable if when disturbed returns to its initial value or remains in the same location. The poles of the closed-loop system $A - BK$ can be obtained by computing the eigenvalues of that system. It is known that $A - BK$ is a stability matrix if all the poles have a negative real part. The time measure is in seconds and the closed-loop poles of the system are

$$\begin{bmatrix} -2.0307 + 2.0307i \\ -2.0307 - 2.0307i \end{bmatrix} \times 10^3, \tag{4.31}$$

which proves the system is stable.

As detailed in section 2.2.4, the separation theorem allows the regulator and estimator gains of the

LQG controller to be designed independently from each other and guarantees that the controller makes the closed-loop system asymptotically stable, given that both $A-BK$ and $A-LC$ are stability matrices. Now, we adjust the weight matrices Q_E and R_E and obtain the optimal LQG estimator gain L using the MATLAB function *lqe*,

$$L_\theta = \begin{bmatrix} 316.228, & 25.169 \end{bmatrix}. \tag{4.32}$$

It is now possible to compute the state-space model for a negative output feedback LQG controller for the process system with regulator gain K_θ and estimator gain L_θ. The attitude baseline controller is used as an initial stabilizing LQG controller, $\frac{S^0(s)}{R^0(s)}$, detailed in section 2.2.4.

Similarly, it is necessary to design the LQR controller for the vertical position of the rocket, z. The goal is to follow the nominal trajectory detailed in section 4.5. For that, the gain matrix obtained was

$$K_z = \begin{bmatrix} 0.79527, & 5.7556e-07 \end{bmatrix}. \tag{4.33}$$

Here, the equilibrium point for the altitude controller is different than zero. Hence, it is necessary to subtract the z_{ON} position obtained for the nominal trajectory from the input altitude.

The time is measured in seconds, and the eigenvalues of the closed-loop system $(A_2 - B_2K_z)$ are

$$\begin{bmatrix} -0.6918 + 0.4781i \\ -0.6918 - 0.4781i \end{bmatrix}, \tag{4.34}$$

which means the system is stable.

The optimal LQG estimator gain L obtained is

$$L_z = \begin{bmatrix} 0.70711, & 1.3836 \end{bmatrix}. \tag{4.35}$$

Lastly, a negative output feedback LQG controller for the process system with regulator gain K_z and estimator gain L_z is computed.

4.7.1 Symmetric Root Locus

Control systems utilize the Symmetric Root Locus (SRL) as a graphical tool to examine how a closed-loop system responds to changes in the gain or other feedback controller parameters. The symmetric root locus is primarily used to provide insight into the stability and performance characteristics of the closed-loop system while designing a controller.

Therefore, this technique is applied to determine the controller gain range that results in a stable closed-loop behavior and to select among the trade-offs between performance indicators.

The SRL gives, for different values of ρ, the poles (natural frequencies) of a controlled system with optimal control. The poles of a closed-loop system with infinite horizon Linear Quadratic (LQ) control are

$$\frac{1}{\rho}\frac{b(s)b(-s)}{a(s)a(-s)} = -1. \tag{4.36}$$

In this case, ρ is the LQR weight matrix R, that influences the regulator controller gains K. When R varies, the roots change. As briefly mentioned in section 2.2.4, if R is large, $i.e.$ the control is expensive, the controlled system is slower and the close-loop poles are either the open-loop poles for a stable system or their symmetric. If the control is too expensive, it is better to leave the poles as they initially are. When a close-loop pole is more to the left of the axis, the control energy required is higher, but the energy dissipated by the state is lower. Thus, the positioning of the pole must be a trade-off between these factors. Alternatively, if R is small, the poles are located further from the origin. Therefore, the controlled system is faster and the bandwidth increases, which can make the closed-loop system unstable.

Here, the Plant $P(s)$ of the attitude system defined in section 4.6.1 is a double integrator. It is necessary to obtain the SRL of

$$P(s)P(-s) = \frac{-8146}{s^2}\frac{-8146}{(-s)^2} = \frac{6.636 \times 10^7}{s^4}. \tag{4.37}$$

From here, as there are no zeros and four poles, the *root locus* will have four branches that begin at the poles at the origin and tend to infinity along the asymptotes with an angle of $45°$. Now, to compute the four fourth squares of $-\frac{1}{\rho}$ as expressed in (4.36), it is necessary to represent them in polar coordinates of the form

$$re^{j\theta},$$

where r represents the absolute value of the root and θ is the phase. From that,

$$r^4 e^{j5\theta} = \frac{1}{\rho}e^{j\pi}.$$

From absolute value equality and congruency of phases proven in [37], it is possible to compute the four roots as

$$r_1 = \frac{1}{\sqrt[4]{\rho}}e^{j\frac{\pi}{4}}, r_2 = \frac{1}{\sqrt[4]{\rho}}e^{j\frac{3\pi}{4}}, r_3 = \frac{1}{\sqrt[4]{\rho}}e^{j\frac{5\pi}{4}}, r_4 = \frac{1}{\sqrt[4]{\rho}}e^{j\frac{7\pi}{4}},$$

in which r_2 and r_3 are the optimal poles located in the Left Half of s-Plane (LHP). Figure 4.21 illustrates the four roots of the SRL computed for the value of the regulator weight $R = 1 \times 10^{-6}$ chosen for the baseline attitude controller designed in section 4.7, intersecting the SRL branches, and a circle with

radius $1/\sqrt[4]{\rho}$.

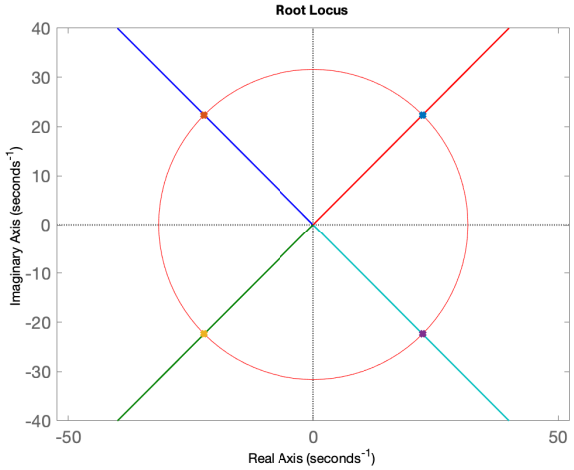
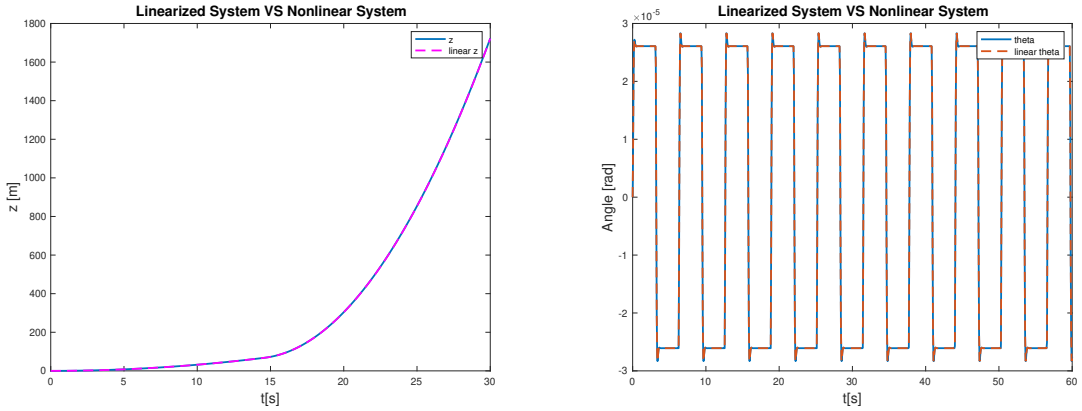


Figure 4.21: SRL of the Plant $P(s)$ with the fourth roots of $-1/\rho$ and a circle with a radius of $1/\sqrt[4]{\rho}$.

4.7.2 Identification Validation



(a) Comparison between the output z of the linear and nonlinear systems to the same step impulse.

(b) Comparison between the output θ of the linear and nonlinear systems to the same step impulse.

Figure 4.22: Identification systems validation simulations results.

The response of the linear and nonlinear systems will now be compared to demonstrate that the system was correctly identified and, consequently, linearized. First, to validate the altitude linear system one must compare the responses of the nonlinear system and the identified Plant to the same step impulse, controlled by the same altitude controller designed in section 4.7. The results are shown in

figure 4.22(b).

Similarly, the process of validating the attitude linear system consists of comparing the output angle θ from both the nonlinear system and the Plant, in close-loop with the attitude baseline controller. Figure 4.22(a) expresses this simulation results.

4.8 Closed-loop Simulations

In this section, we perform several closed-loop simulations using the LQG controllers designed in section 4.6.2 for the rocket model.

The main goal of the attitude controller is to ensure that by changing the input angle φ, the trajectory remains vertical, keeping the angle θ as close to zero as feasible.

The gimbal angle limit of oscillation should remain between

$$-15^{\circ} < \varphi < 15^{\circ},$$

equivalent to, approximately, 0.26 radians.

The goal of the altitude controller is to assure that the rocket follows the defined nominal trajectory. For that, the thrust force applied in all simulations is a *bang-bang* profile force, illustrated in figure 4.23.

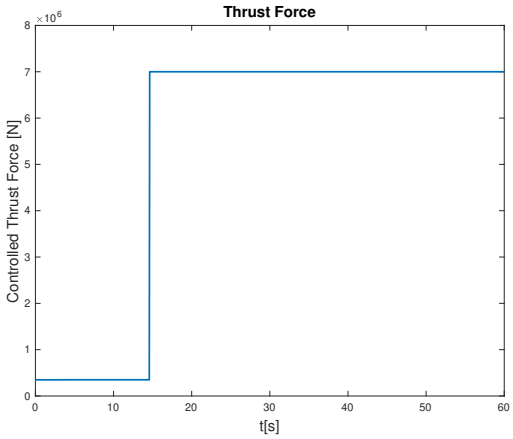
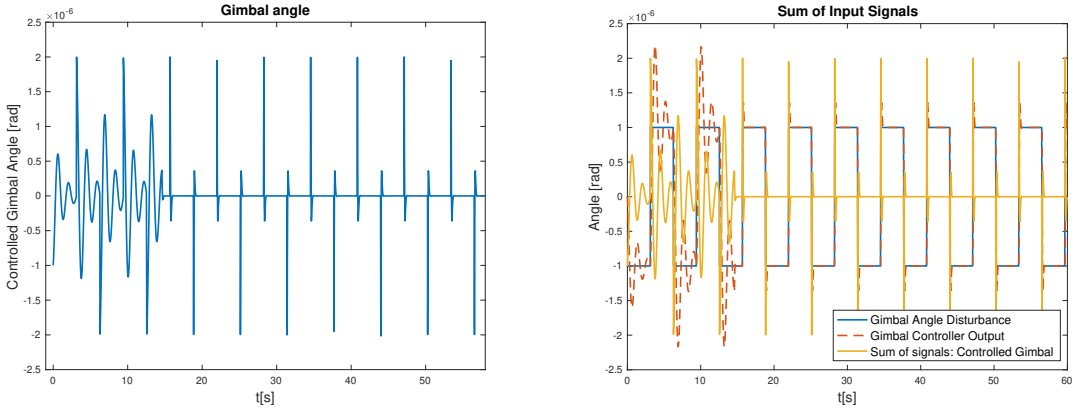


Figure 4.23: Main thrust force F_E using a *Bang-Bang* control profile.

Test 5

The first simulation has a square wave input corresponding to the gimbal angle φ, shown in figure 4.13, to disturb the rocket on its descent trajectory.

The gimbal angle stabilizes around the approximate value of zero. The gimbal angle φ is the difference between the sum of the disturbance signal with the control signal, as shown in 4.24.



(a) Controlled gimbal angle input φ over time.

(b) Sum of input signals.

Figure 4.24: Gimbal angle φ for test 5.

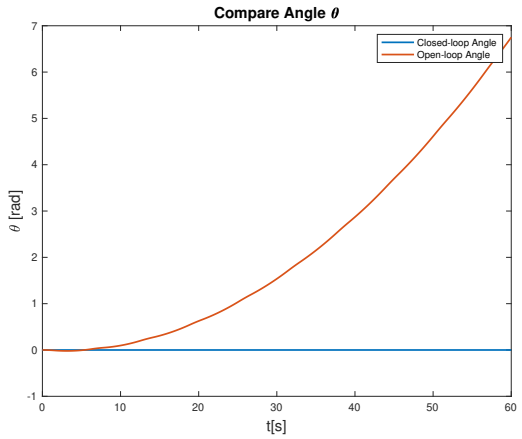


Figure 4.25: Comparison of angle θ for the open-loop and closed-loop simulation of test 5.

The more pronounced oscillations for the first 14.6 seconds in figures 4.24 align with the *bang-bang* control trajectory "switch" time instance t_{ON}, in which the thrust is considered OFF. Therefore, since the force is not zero it is still possible to control the gimbal angle, but not as well as after t_{ON}s. The controlled angle θ exhibits an identical effect during the same time period.

The peaks observed in the controlled gimbal angle for each square in the square wave input do not represent overshoot, but rather the fact that the signal experiences a slight delay in its response to a quick change in the input signal.

The resulting controlled angle θ is illustrated in figure 4.26, showing that the angle oscillates near

zero, with a peak maximum value approximately around $\theta = 5 \times 10^{-5}$ rad, and the angular velocity $\dot{\theta}$ value stabilizes around zero. The resulting trajectory in figure 4.27(a) is the nominal trajectory with a

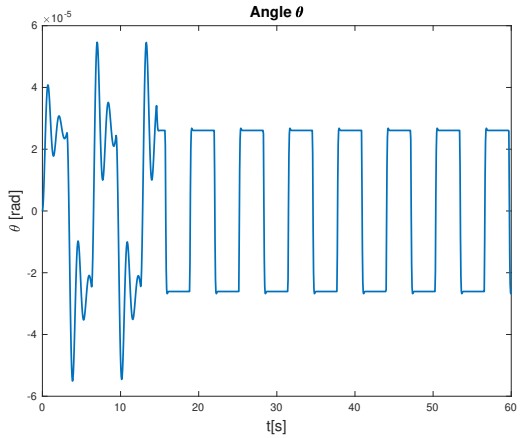
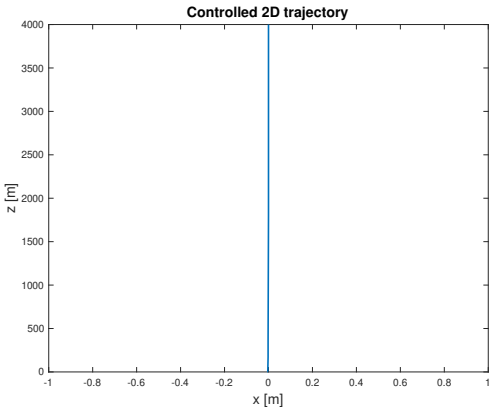
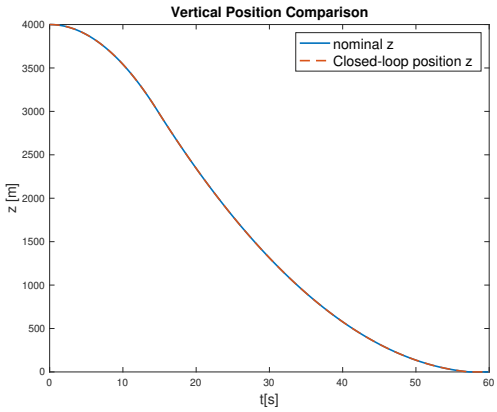


Figure 4.26: Controlled angle between the z axis of the plan and the longitudinal axis of the rocket, θ, over time for test 5.

horizontal deviation of $x = 3 \times 10^{-4}$m. The vertical position of the rocket over time is in figure 4.27(b). In figure 4.25 is shown the comparison between the open-loop and closed-loop simulation for the same



(a) 2-Dimensional trajectory, with horizontal position in the x axis, and vertical position in z axis.



(b) Vertical position of the rocket over time, z.

Figure 4.27: Representation of the rocket's trajectory for test 5.

conditions. It is proven that the angle θ now remains close to zero over time, even when the gimbal angle is disturbed.

Test 6

Now, we input a random band-limited white noise disturbance for the gimbal angle φ, with power 1×10^{-10} and time sample of 1s with seed $[23341]$, as illustrated in figure 4.28. The thrust force applied is a *bang-bang* profile force, illustrated in figure 4.23.

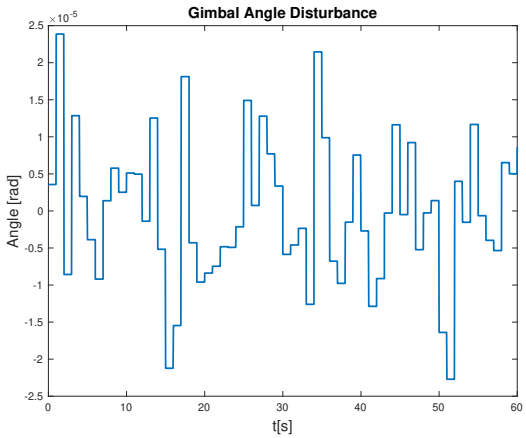
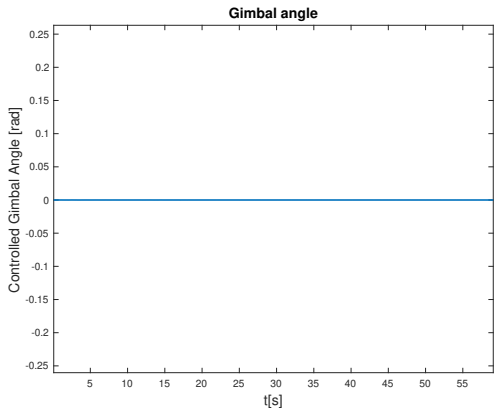
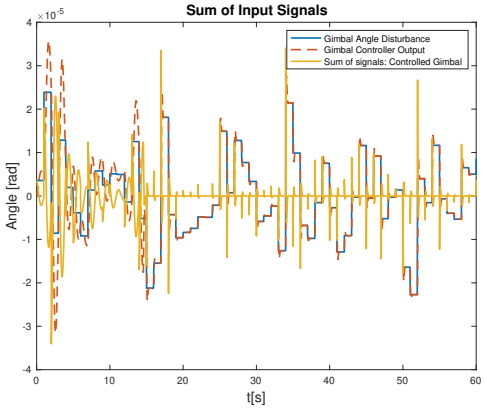


Figure 4.28: Gimbal angle φ band-limited white noise disturbance input for test 6.

When controlled, the gimbal angle input stabilizes around the approximate value of zero as seen in figure 4.29(a). The vertical scale of figure 4.29(a) was altered to a gimbal angle interval of, approximately, $-0.26 < \varphi < 0.26$ radians. The gimbal angle φ is the difference between the sum of the disturbance signal with the control signal, as shown in figure 4.29(b). This result proves the controller is compensating the disturbance.



(a) Controlled gimbal angle input φ over time for test 6.



(b) Sum of input signals for test 6.

Figure 4.29: Gimbal angle φ for test 6.

The resulting controlled angle θ is illustrated in figure 4.30, showing that the angle oscillates near zero and the angular velocity $\dot{\theta}$ value stabilizes around zero. The vertical position of the rocket over time is represented in figure 4.31(b), which approximates the nominal trajectory. The resulting 2-dimensional trajectory in figure 4.31(a) has a horizontal deviation of $x = 0.61$ meters from the vertical trajectory.

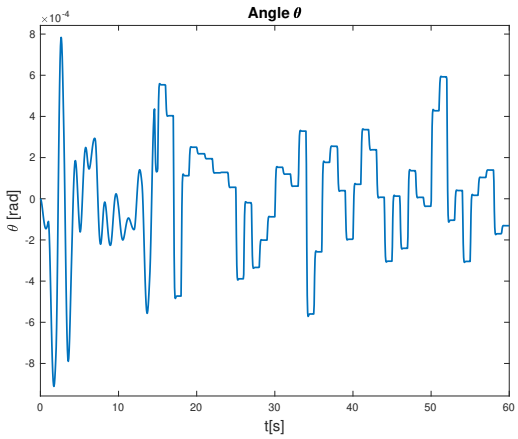
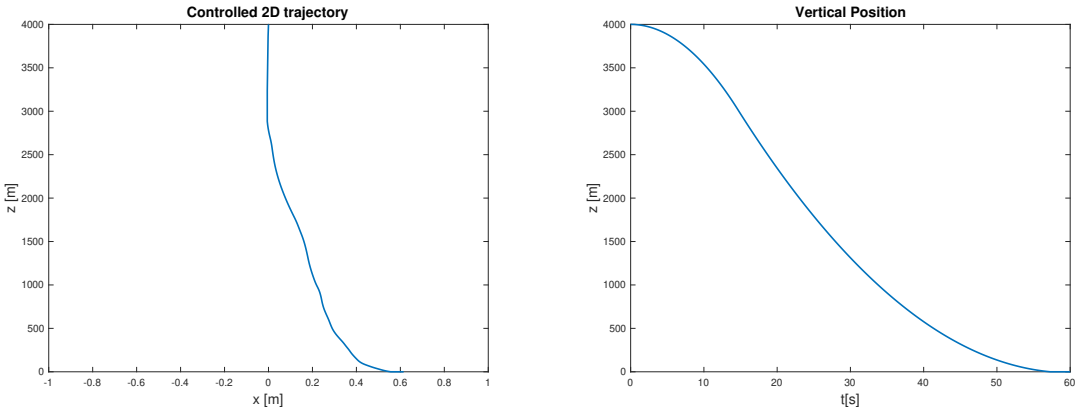


Figure 4.30: Controlled angle between the z axis of the plan and the longitudinal axis of the rocket, θ, over time, for test 6.



(a) 2-Dimension trajectory

(b) Vertical position of the rocket over time, z.

Figure 4.31: Representation of the rocket's trajectory for test 6.

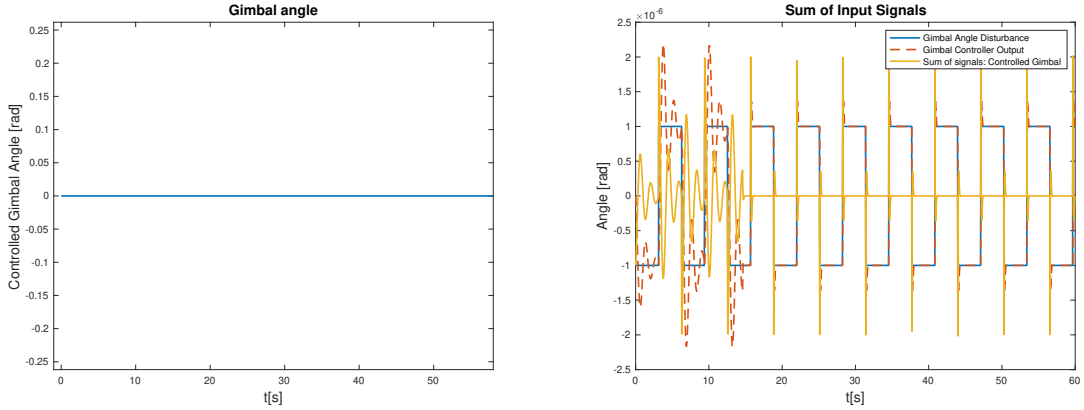
Test 7

For this test, band-limited white noise was added to the horizontal acceleration of the rocket in the nonlinear model using the *Simulink* block that introduces white noise into the continuous system. The

horizontal “wind”/ noise, is a random band-limited white noise with power 1×10^{-4}, sample time (correlation time of noise) of 1, and starting seed [23341].

Similarly to the simulation performed on “Test 5”, the closed-loop simulation on “Test 7” inputs a square wave input gimbal angle seen in figure 4.13, as a disturbance. The thrust force applied is a *bang-bang* profile force, previously seen in figure 4.23.

For these conditions, it is expected that the trajectory will have more horizontal deviation, because of the added “horizontal wind” acting on the rocket, but it will still be possible to control the nominal position of the rocket and attitude. The attitude control results remain unaltered from “Test 5”, since the white noise added only affects the horizontal acceleration equation. Figures 4.32(a) and 4.32(b) are the resulting controlled gimbal angle φ and the data overlap of the disturbance and control signal.



(a) Controlled gimbal angle input φ over time.

(b) Sum of input signals.

Figure 4.32: Controlled φ angle input for test 7.

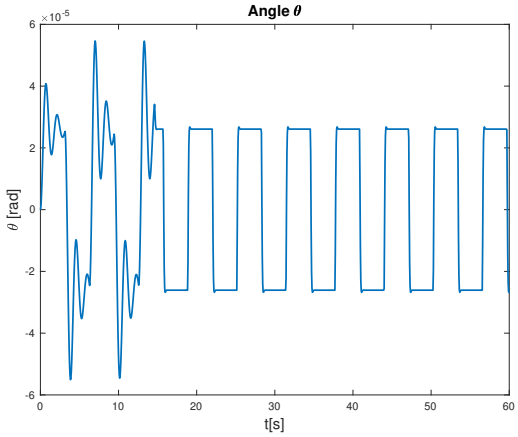
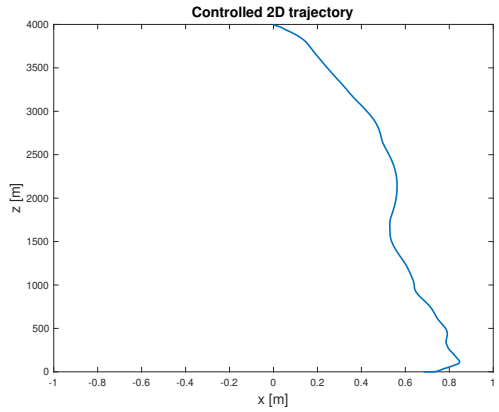


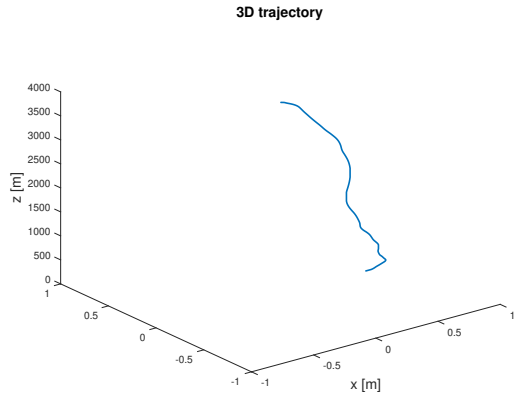
Figure 4.33: Controlled angle, θ, over time, for test 7.

The output variable of the angle θ and angular velocity $\dot{\theta}$ remain close to zero radians over time, proven in figure 4.33.

The vertical position of the rocket remains the nominal trajectory. Although, the trajectory is now more horizontally disturbed in comparison with "Test 5" with a deviation of $x = 0.69$m, as proven in figure 4.34(a) that shows the 2-dimensional trajectory of the rocket, and in figure 4.34(b) that illustrates the 3-dimensional space.



(a) 2-Dimension trajectory, with horizontal position in the x axis, vertical position in z axis.



(b) 3-Dimension trajectory with horizontal position in the x axis, vertical position in z axis, and y set to zero.

Figure 4.34: Representation of the rocket's trajectory with addiction of horizontal noise.

5

Experiments and Results

Contents

In this section we present the most relevant results and discussions from this study. The implementation details are also shown. First, under different initial conditions, we solve the attitude control problem using both the MATLAB solver *Fminunc* described in section 3, and the learning algorithm Episodic REINFORCE, applying the YK parameterization to the rocket model. Then, we test the solver *Fminunc* in optimization problems of varying complexity in order to get baseline values for validation of the optimization process.

5.1 Implementation Details

This section presents details of the algorithm implementation and initialization for the experiments.

The choice of the initial parameters ϕ^0, α, and η is important since it influences the evolution of the system. The definition of the appropriate base function that computes the parameter $Q(s)$ and represents the stabilizing controllers, shown in (2.1), as well as the type of input signal to which the system will respond, affect the optimization process and the speed of convergence to a minimum.

The learning rate must be dynamic. The technique used, as expressed in section 3.2.1, consisted of making the parameter an arbitrarily large number and adapting it by considering the optimization status, making sure the value was small when closer to the minimum cost, so it would not get lost.

It has been found that the search for the minimal cost function becomes more challenging and may result in high-order controllers as the order of the vector of parameters ϕ increases. Moreover, the choice of the initial vector ϕ^0 can determine whether the algorithm becomes "stuck" in local minima. Also, it is known that the chosen constant closed-loop pole α is faster if it is further away from the imaginary axis. The pole α should be chosen within the range of frequencies for which the closed-loop response is expected to behave interestingly [2].

The number of iterations N is important because increasing the number of iterations until the computation is no longer feasible or the controller order is too high is one of the only methods to ensure that a solution that minimizes the cost can be found.

In stochastic algorithms such as Episodic REINFORCE, the seed is important since it initializes the random number generator, ensuring the reproducibility of results. Therefore, by altering the seed value, it is possible to provide information about the algorithm's sensitivity to the chosen initial conditions and help in the identification of robust approaches that function well under different randomization circumstances.

Algorithm 5.1 shows the approach used for Q design using RL, for solving the attitude control problem of the rocket model plant $P(s)$ identified in Chapter 4.

5.1.1 Controller Order Reduction

High-order controllers can be generated when the optimization problem is addressed to a vector of parameters ϕ with large dimensions, as a consequence of the *Ritz Approximation* exponential in (2.1), which can lead to stability issues and other difficulties. For that, one can use the MATLAB function *reduce* to decrease the controller order maintaining similar steady-state error performance, and closed-loop characteristics. The *reduce* function allows to specify several options, such as choosing through the model reduction algorithms (*e.g.* "ncf" meaning Normalized Coprime Factorization (NCFMR)), error types (*e.g.* "ncf" referring to the normalized unstable gain and phase "nugap" error), and the specific order of the desired reduced controller.

Although it can make the process simpler to implement, and easier to understand, reducing the order of the controller using this function, is not always accurate, which can result in performance loss. Hence, before applying it in a real-world system, it is crucial to thoroughly assess the reduced controller's performance, for instance by comparing the Bode diagrams of both controllers.

5.2 Youla-Kucera Parameterization Applied to the Rocket Model

This section presents the experimental results of the attitude control problem of the rocket model derived in Chapter 4, using the Episodic REINFORCE algorithm. The problem is formulated in section 2.1, to find the optimal Youla parameter that provides all the stabilizing controllers for the linear plant $P(s)$.

The objective of the attitude controller, such as the closed-loop simulations using the LQG baseline controllers carried out in section 4.8, is to guarantee that by changing the input gimbal angle φ, the trajectory remains vertical, which entails maintaining the angle θ close to zero regardless of the disturbance applied. The gimbal angle limit of oscillation should continue to range between $[-15, +15]^{\circ}$.

The altitude controller is the LQG baseline controller designed in section 4.7, which assures that the rocket follows the defined nominal trajectory in section 4.5. For that, the thrust force applied in all simulations is a *bang-bang* profile force, illustrated in figure 4.23.

The metric used to evaluate the deviation of the controlled systems from the goal angle θ and the nominal trajectory is the Sum of Squares Error (SSE). The cost function used is expressed in (2.2), and all results will be compared to the ones obtained using the baseline controllers to validate the optimization.

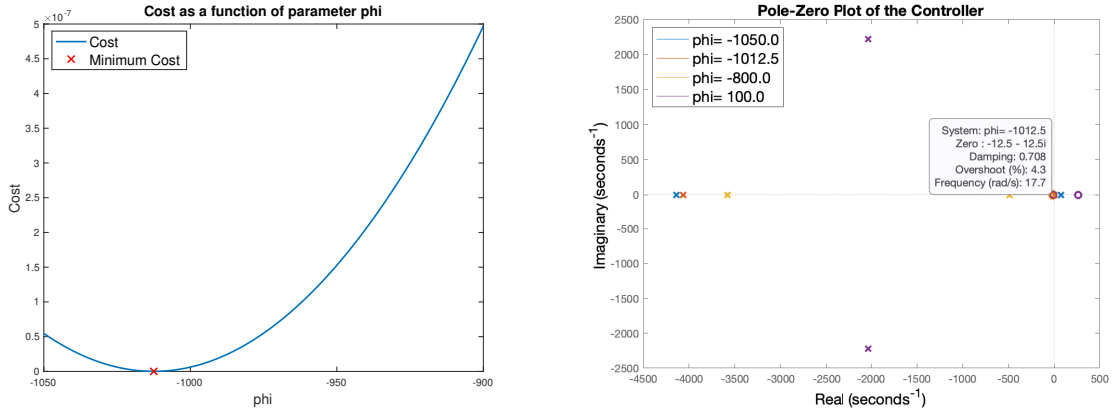
5.2.1 Results and Discussion

The constant values considered for the simulations are presented in Table 4.1. The dimensions of the rocket were chosen accordingly to the first-stage rocket data of Falcon 9 [45], [46].

Test 8

The first test has a square wave input corresponding to the gimbal angle φ, shown in figure 4.13, to disturb the rocket on its descent trajectory. The parameter ϕ^0 is set to zero, to ensure the initial condition of the parameter is equal to the baseline LQG attitude controller, since that if the Youla parameter $Q(s)$ is zero the controller is the initial stabilizing controller S^0/R^0 as proven in (2.5).

Initially, a test of the parameter ϕ was done, in which the cost was manually calculated for a range of values of this parameter to speculate the one that would give the minimum cost. The relation between the cost function value and ϕ is in figure 5.1(a), in which is possible to conclude that the goal final value of ϕ is around -1012.5.



(a) Cost as a function of ϕ.

(b) Pole-Zero plot of the Youla controller.

Figure 5.1: Influence of the parameter ϕ on the minimization of the cost and pole-zero distribution.

Figure 5.1(b) depicts the pole-zero map of the Youla Controller (2.5), from which one concludes that $\phi = -1012.5$ puts, simultaneously, the farthest and the closest pole to the origin before it gets unstable. It is known that for the system to be able to reject input disturbances the controller has to have a pole at the origin. The cost function used in this work is "blind" to elements such as the effect of fuel burn on the rocket's mass and other factors that influence performance. Thus, the controller prioritizes speed over robustness. Also, it is possible to see in figure 5.1(b) that when ϕ is positive, the poles are no longer purely real negative, but complex conjugate poles.

The location of the poles of the controller has a great impact on the system's closed-loop behavior. Poles closer to the imaginary axis lead to a slower response, since they have slowly decaying components, but can result in potential oscillations or instability, especially on marginally stable systems. Poles located farther from the imaginary axis provide a faster response because they decay rapidly, and have a larger magnitude which leads to a greater margin of stability.

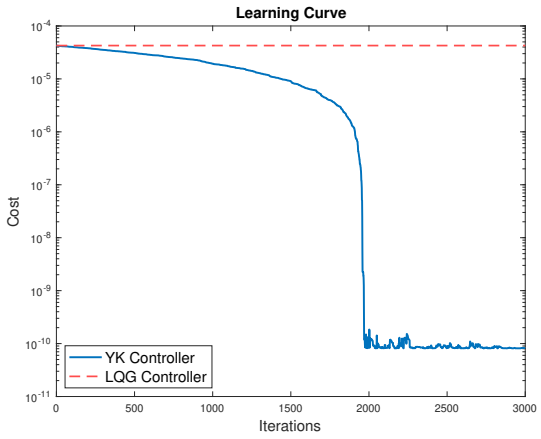
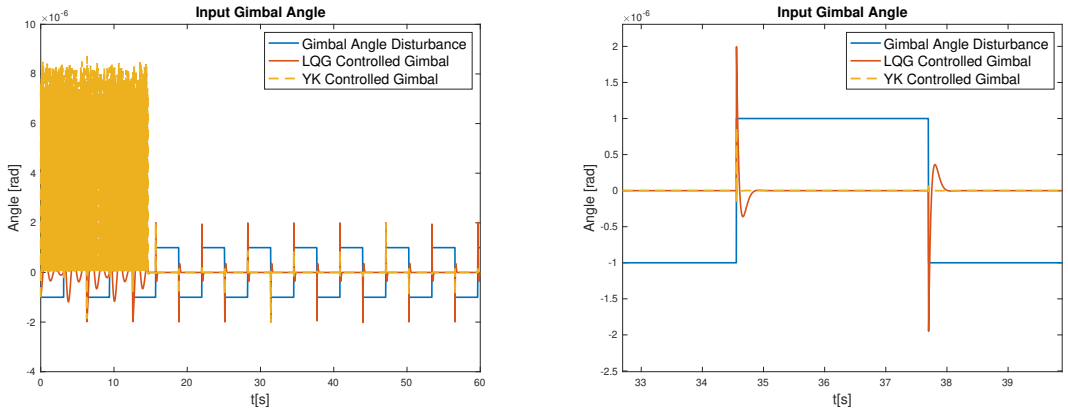


Figure 5.2: Learning curve of the Episodic REINFORCE algorithm compared to the cost of the attitude LQG controller for test 8.

The comparison between the learning curve of the Episodic REINFORCE and the constant cost of the LQG controller is found in figure 5.2. The optimization started with a learning parameter of $\eta = 100$ and started gradually adapting it after, approximately, 1800 iterations where the cost significantly decreases. After that, the parameter that gives the minimum cost was found at $\phi^{2732} = -1012.5$, which is consistent with the test results from figure 5.1, and similar to the solver *Fminunc* that achieved a final value of, approximately, $\phi = -1012$.



(a) Sum of input signals.

(b) Close up of the input signals.

Figure 5.3: Input noise and gimbal angle φ using the YK controller compared to the LQG controller for test 8.

The gimbal angle stabilizes around the approximate value of zero. The gimbal angle φ is the difference between the sum of the disturbance signal with the control signal shown in 5.3. As previously explained in section 4.8, the heavy oscillations for the first 14.6s present in figure 5.3(a) correspond to the *bang-bang* control trajectory "switch" time instance t_{ON}, in which the thrust is considered OFF. The

peaks, seen in detail in figure 5.3(b), are not overshoot but a consequence of the signal's abrupt changes and are less pronounced with the Youla controller.

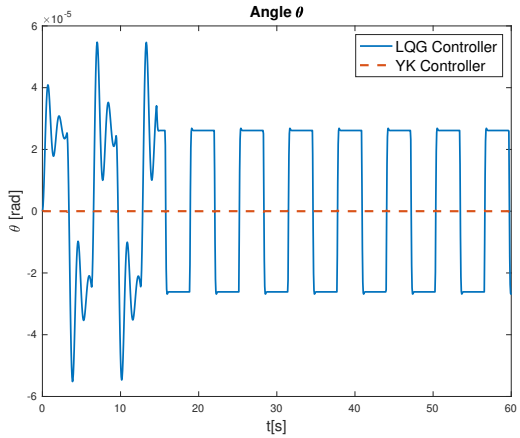
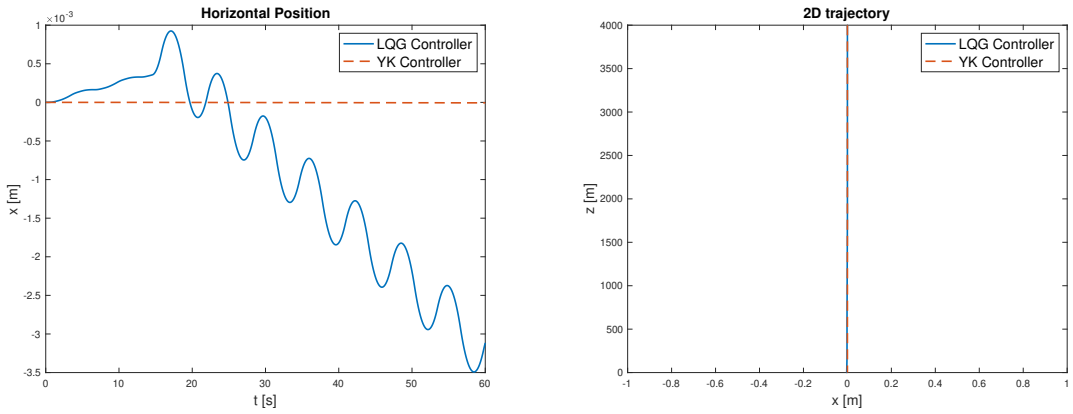


Figure 5.4: Comparison of the controlled angle between the z axis of the plan and the longitudinal axis of the rocket, θ, using the YK or the LQG controller for test 8.

In figure 5.4 are depicted the two output angles from the two different controllers. As shown, the Youla controller lead to an attitude angle much closer to zero, which results in a more vertical trajectory, that is, with less horizontal deviation as seen in figure 5.5.



(a) Horizontal position x.

(b) 2-Dimensional trajectory of the rocket.

Figure 5.5: Comparison between the controlled trajectories using the YK or the LQG controller for test 8.

The step responses of the initial, baseline, and Youla controllers are in figure 5.6, where it is proven that the overshoot is less prominent with the YK controller. The YK controller obtained was

$$K(s) = \frac{-1013s^2 - 2.533e04s - 3.162e05}{s^2 + 4061s - 2150}. \tag{5.1}$$

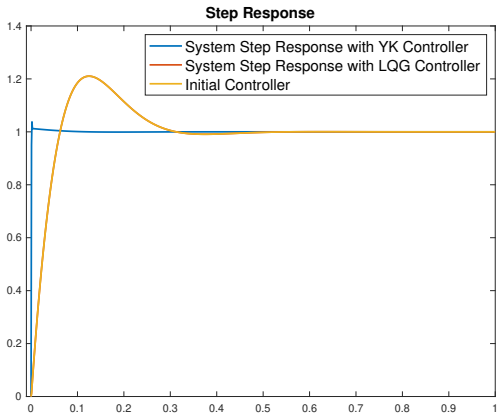


Figure 5.6: Representation of the step responses of the closed-loop systems for test 8.

The SSE obtained with the YK controller is $SSE_{yp} = 8.41 \times 10^{-11}$, when with the baseline controller is $SSE_{LQG} = 4.25 \times 10^{-5}$, which corresponds to an, approximately, 99% optimization.

Initial results were obtained with a starting seed value of 100 to ensure reproducibility of results. However, subsequent experiments with different randomly selected seed values yielded similar results. Regardless of the seed value used, the algorithm converged to the same minimum cost value. This indicates that the performance and convergence behavior of the algorithm is robust and stable, showing some degree of independence from the initial seed value. These results confirm the reliability and consistent performance of the algorithm, even with random variations in the seed.

Test 9

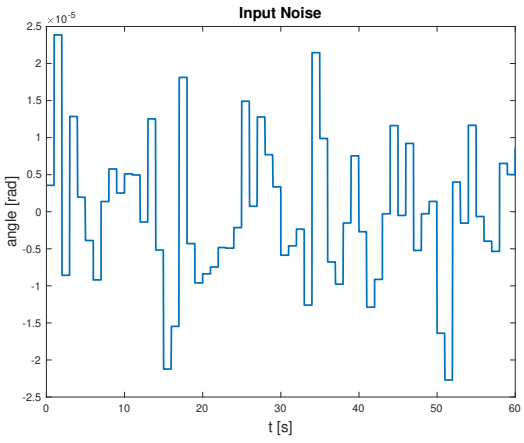


Figure 5.7: Gimbal angle φ band-limited white noise disturbance input for test 9.

This test is a parallel of “test 6” conducted in section 4.8 with a Youla controller instead of an LQG

controller, in which we input a random band-limited white noise disturbance for the gimbal angle φ, with power 1×10^{-10} and time sample of 1s with seed $[23341]$, as illustrated in figure 5.7. The thrust force applied is a *bang-bang* profile force, illustrated in figure 4.23.

Figure 5.8 shows the comparison between the learning curve of the Episodic REINFORCE and the constant cost of the baseline controller. The optimization started with a learning rate of $\eta = 100$ and it was gradually adapted using the techniques detailed in section 3.2.1. After that, the parameter that gives the minimum cost was found at $\phi^{2990} = -1012.5$, which is consistent with the results obtained by the solver *Fminunc*, that achieved a final value of, approximately, $\phi = -1012$.

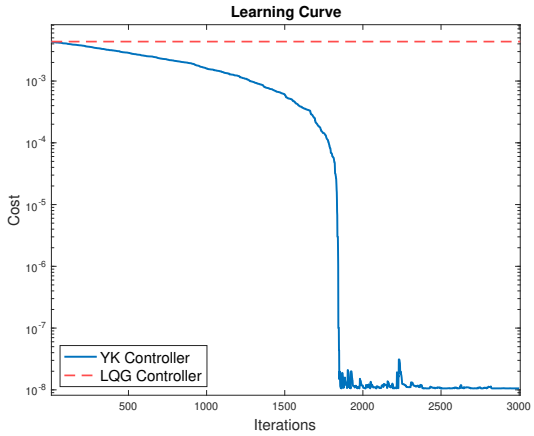


Figure 5.8: Learning curve of the Episodic REINFORCE algorithm compared to the cost of the attitude LQG controller for test 9.

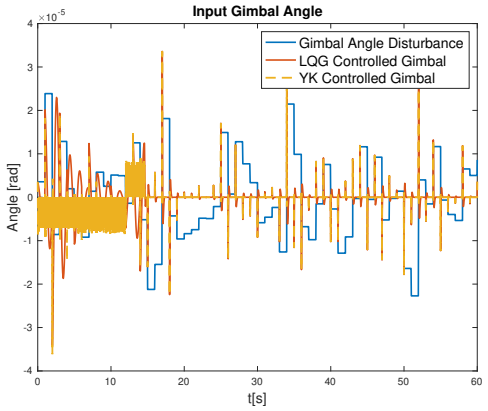
As expected, when controlled, the gimbal angle input is around zero. The gimbal angle φ is the difference between the sum of the disturbance signal with the control signal, as shown in figure 5.9(a), and its derivative does not oscillate past the authorized range values.

The output angle θ for both controllers is depicted in figure 5.9(b), in which the Youla controller provides an output angle that is less oscillatory and closer to the reference value. This result proves the controller is compensating the random noise disturbance. The resulting 2-dimensional trajectory shown in figure 5.10(b) controlled by the Youla controller has a significantly smaller horizontal deviation from the vertical trajectory, than the one with the baseline controller, as proven in figure 5.10(a).

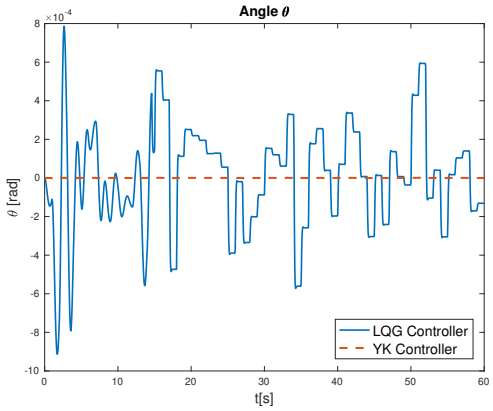
When compared with the baseline controller, the YK controller provides an SSE of around 1.0548×10^{-8}, which is an improvement of more than 99% from $SSE_{LQG} = 4.32 \times 10^{-3}$.

Test 10

An extra test was carried out in light of the importance of wind in controlling the attitude of a rocket. Using a *Simulink* block created to include white noise into the continuous system, band-limited white

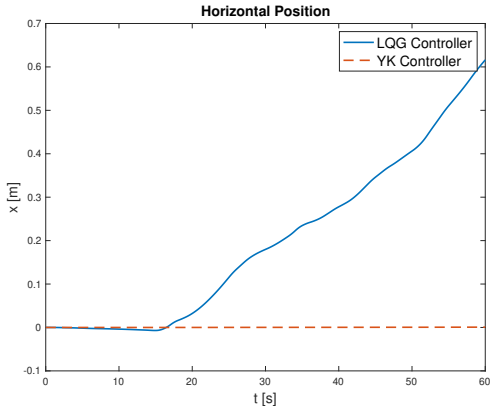


(a) Input noise and gimbal angle φ using the YK controller compared to the LQG controller.

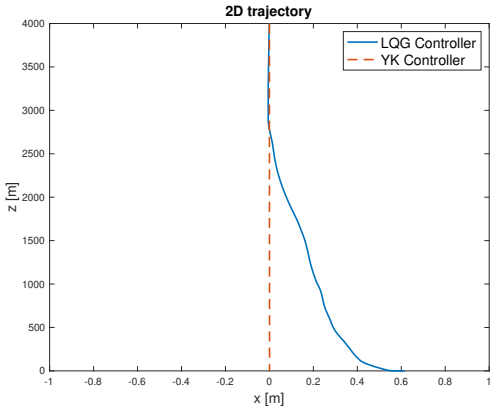


(b) Comparison of the controlled angle, θ, using the YK or the LQG controller.

Figure 5.9: Comparison of the controlled angles for test 9.



(a) Horizontal position of the rocket x.



(b) 2-Dimensional trajectory of the rocket.

Figure 5.10: Comparison between the controlled trajectories using the YK or the LQG controller for test 9.

noise was added to the horizontal acceleration of the rocket within the nonlinear model to imitate real-world circumstances. It had a seed value of [23341], a sample time (correlation time of noise) of 1, and a power setting of 1×10^{-4}, for concordance with the closed-loop simulations carried out in section 4.8. This wind factor was used in the study to take external factors' effects on rocket attitude control into consideration.

Figure 5.11 shows a comparison between the LQG controller's constant cost and the learning curve of the Episodic REINFORCE algorithm. The optimization began with a learning rate of 100 and gradually adjusted, similarly to the previous experiments. The minimum was found at $\phi^{2733} = -1012.5$, corresponding to an SSE of 8.16×10^{-11}. The solver *Fminunc* stopped at the parameter $\phi = -1011.99$

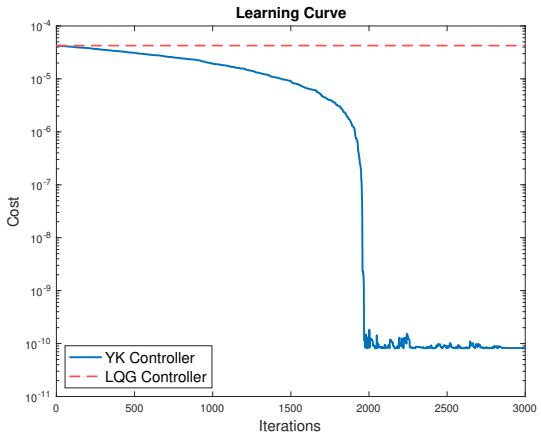
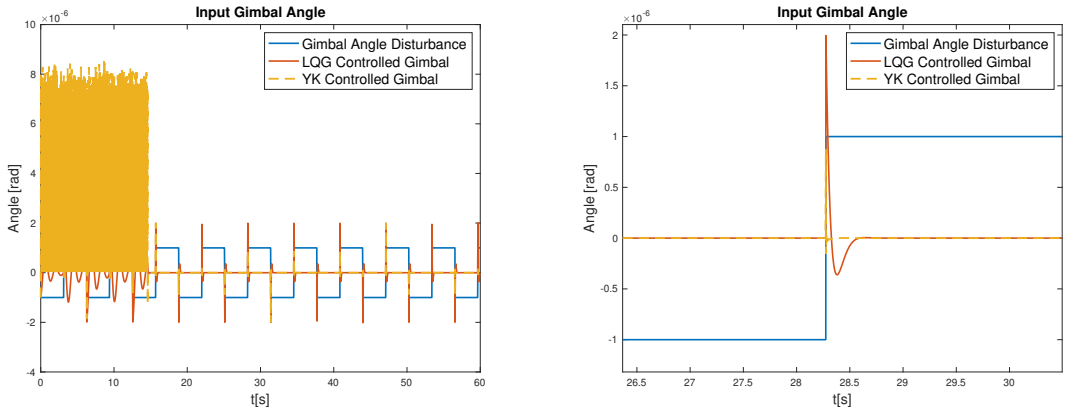


Figure 5.11: Learning curve of the Episodic REINFORCE algorithm compared to the cost of the attitude LQG controller for test 10.

corresponding to an SSE of 9.57×10^{-11}. The baseline controller error is $SSE_{LQG} = 4.26 \times 10^{-5}$, which represents an, approximately, 99% optimization.

The gimbal angle input oscillates around zero as seen in figure 5.12, and its derivative does not oscillate past the authorized range values of $[-15, +15]^{\circ}$. The controlled angle θ is closer to the reference value using the Youla parameter than the baseline controller as proven in figure 5.13(a).

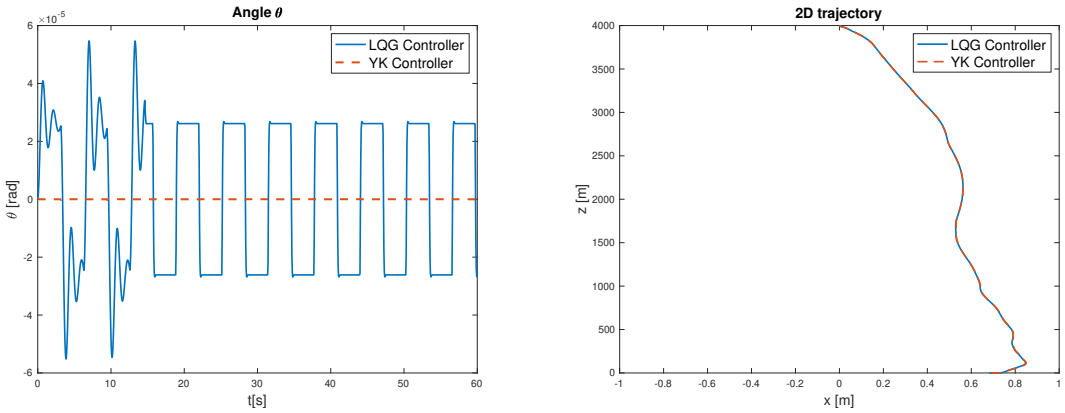


(a) Sum of input signals and controlled gimbal angle φ.

(b) Close up of the input signals.

Figure 5.12: Input noise and gimbal angle φ using the YK controller compared to the LQG controller for test 10.

With the addition of horizontal wind on the second derivative of the position x on the *Simulink* in figure 4.3 of the nonlinear model of the rocket, it is expected to have some horizontal deviation, even with an optimized controller for the attitude angle, as one can see in figure 5.13(b). It is possible that the cost function (2.2) mainly highlights the error between the reference and output attitude angle without



(a) Comparison of the controlled angles θ.

(b) 2-Dimensional trajectory of the rocket.

Figure 5.13: Comparison between the attitude angle output for the YK and LQG controller, and representation of the trajectory of the rocket for test 10.

explicitly considering other trajectory-related elements. Hence, the cost is optimized but the trajectory is still deviated.

5.3 Reference Following Tests

This section presents the results of systems of different orders response to input signals such as a sum of square waves of different frequencies that work as a reference. However, for the sake of brevity and to provide a comprehensive analysis, one example of an unstable open-loop system with non-minimum phase roots is shown here, while other case study is presented in Appendix A. The selected example presented here serves as a representative case and provides valuable insight into reference tracking and controller order reduction, for a more complex system.

The goal is to find the optimal transfer function that makes the Plant $P(s)$ track the reference $T(s)$. The algorithm is equivalent to 5.1, but instead of a $Simulink$ simulation of the rocket model, it is a simulated time response of a dynamic system, with the MATLAB function *lsim*, of both the goal transfer function $T(s)$ and the closed-loop system. The cost function used is expressed in (2.3).

5.3.1 Results and Discussion

The input consisted of a sum of two square wave signals with different frequencies and amplitudes, to provide a signal that has more information. For that, two square wave signals were generated. The first square wave u_0 has a period of $\tau_0 = 15$s, and a duration of 60s, in minimum increments of $\tau/64$, but with a chosen sampling time of 0.01s. The second wave u_1 has the same settings but a period of $\tau_1 = 40$s.

The final square wave signal is of the form

$$u(t) = (2u_0 + u_1) \times (1 \times 10^{-3}). \tag{5.2}$$

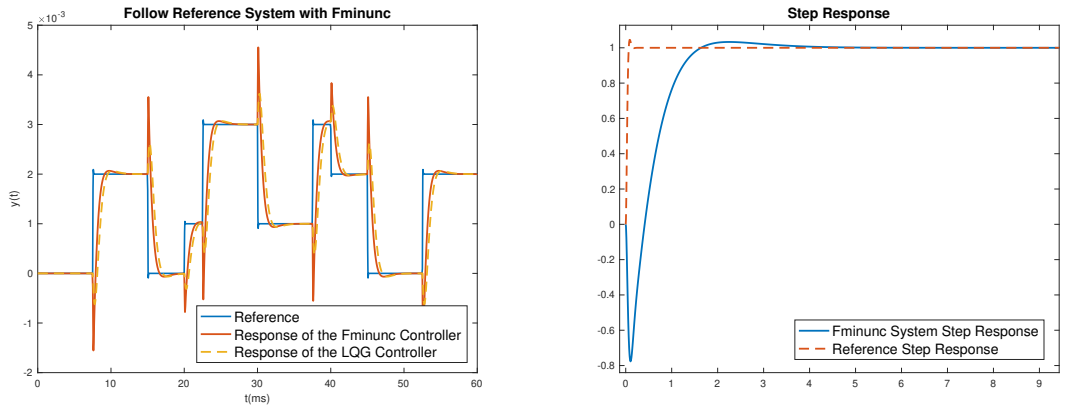
Test 11

Consider the transfer function of the second order closed-loop control system $T(s)$ and Plant $P(s)$

$$T(s) = \frac{\omega^2}{s^2 + 2\xi\omega s + \omega^2} = \frac{50^2}{s^2 + 2 \cdot 0.7 \cdot 50s + 50^2}, \quad P(s) = \frac{s-2}{s(s+1)(s+3)}, \tag{5.3}$$

that is an open-loop unstable system with a non-minimum phase root.

Now, we will perform the optimization of a baseline LQG controller that stabilizes the system using the MATLAB solver *Fminunc*. Figure 5.14(a) shows the response of the controlled system to the impulse compared to the reference response for $\alpha = 10$, and the initial vector of parameters ϕ^0 with a dimension equal to 3. This shows that the controller stabilizes the system and follows the reference, with a steady-state error equal to 1, as proven in figure 5.14(b).

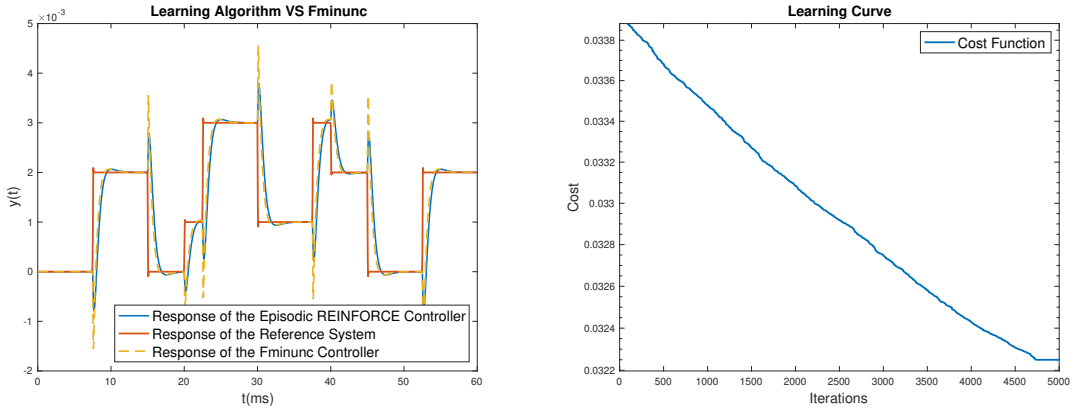


(a) Response of the resulting controller compared to the reference transfer function.

(b) Step response of the resulting controller compared to the reference.

Figure 5.14: Optimization results obtained with *Fminunc* for test 11.

The same circumstances using the learning algorithm Episodic REINFORCE lead to similar results, but very slowly. The response of the learning controller to the input signal is in figure 5.15(a), and the learning curve is illustrated in figure 5.15(b), which shows the cost function is decreasing with the iterations. Given the stochastic nature of the Episodic REINFORCE method, it is natural that a larger vector of parameters ϕ that minimize the cost is more difficult and more time expensive to optimize. Also, it is very unlikely to find the same final vector of ϕ as *Fminunc*, even initializing the random factors with the same seed.



(a) Response of the resulting controllers compared to the reference transfer function.

(b) Learning curve of the Episodic REINFORCE algorithm.

Figure 5.15: Episodic REINFORCE algorithm results for test 11.

The step response of both closed-loop systems and the reference transfer function is depicted in figure 5.16, where is possible to conclude that the inverted response of the closed-loop system obtained by the Youla controller, despite the cost, is not as accentuated. The steady-state error is 1. The initial

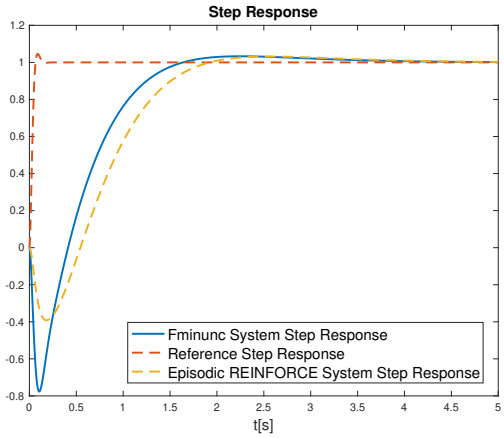
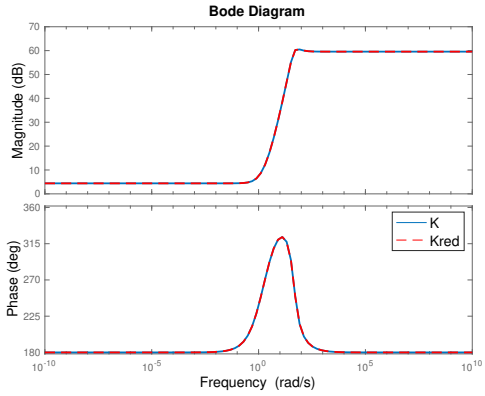


Figure 5.16: Step responses of the closed-loop systems compared to the step response of the reference system.

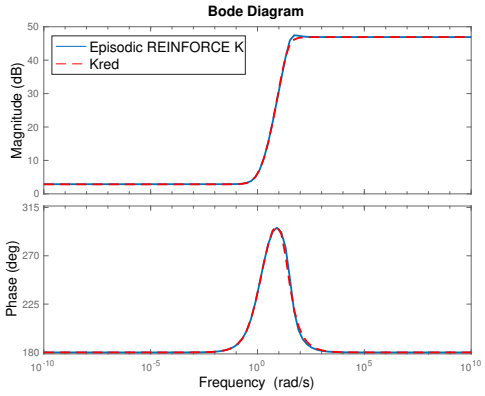
SSE of the baseline LQG controller is 0.035. The cost obtained using the solver *Fminunc* is 0.030 and using Episodic REINFORCE is 0.032, which corresponds to a 13.15%, and 6.97% optimization ratios, respectively.

The fact that the parameter ϕ is 3-dimensional will influence the number of base functions in the Youla parameter $Q(s)$ in (2.1). Hence, the output controller will be of order 6, which means we can try to reduce its order using the technique in section 5.1.1. We will use this function to reduce the order of both controllers to 2. Figure 5.17 shows the Bode diagrams of both reduced controllers, which proves

that the frequency response is equivalent with 6 states removed.



(a) Bode diagram of the controller obtained with *Fminunc* compared to the reduced order controller.



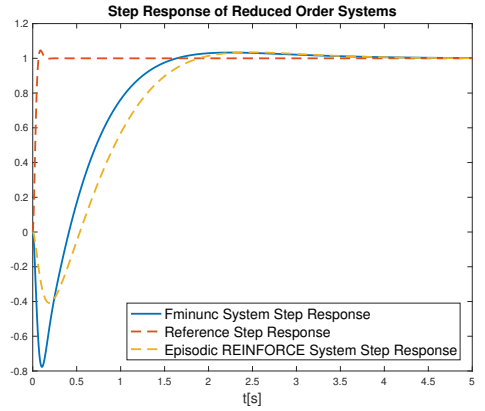
(b) Bode diagram of the learning algorithm controller compared to the reduced order controller.

Figure 5.17: Bode diagrams of the reduced order controllers.

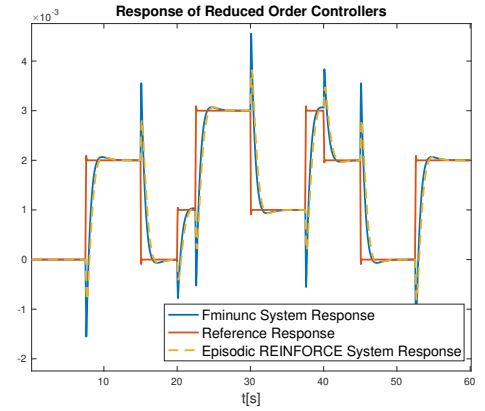
The new controllers obtained from *Fminunc* and the learning algorithm are

$$K(s) = \frac{-951.9s^2 - 4075s - 3353}{s^2 + 46.52s + 2020}, \quad K_{learning}(s) = \frac{-221.4s^2 - 1046s - 849.8}{s^2 + 39.89s + 612}. \tag{5.4}$$

The step responses of the reduced order closed-systems compared to the reference are illustrated in figure 5.18(a).



(a) Step responses of the reduced closed-loop systems compared to the step response of the reference system.



(b) Response of the two reduced controllers compared to the reference transfer function.

Figure 5.18: Results using the reduced controllers.

Algorithm 5.1: Episodic REINFORCE Optimization Algorithm for the Rocket Model

$P \longleftarrow$ rocket_system()

$\frac{s^0}{r^0} \longleftarrow$ controller_design(P)

Inputs: Maximum number of iterations (N), initial vector of parameters ϕ (ϕ^0), goal value of cost, time vector, input signal, angle reference (θ_{goal}), α, learning parameter (η)

Outputs: Final vector of parameters ϕ (v_ϕ), response of the controlled system (y_f), cost function (J), the minimum cost (J_{min}), minimum cost iteration (k_{min}), minimum value of optimization parameter (ϕ_{min}), the Youla parameter (Q), the YK controller (K), vector of learning rates (v_η)

Define: vector of ϕ (v_ϕ), vector of learning rates (v_η), adjustable parameters for adaptive η

for $k = 1$ to N **do**

  for $i = 1$ to length$(v_\phi(k,:))$ **do**

    Compute equation (2.1) to obtain matrix $Q(s)$:

    $Q(s) \longleftarrow Q(s) + v_\phi(k,i) * \left(\frac{\alpha}{s+\alpha}\right)^{(i-1)}$

  Compute controller in (2.5) using $Q(s)$

  Run: *Simulink* of the rocket model

  Output: Attitude angle θ

  Compute closed-loop transfer function in (2.8) using $Q(s)$

  Compute cost function as shown in (2.2):
  $J(k) \longleftarrow sum((\theta_{goal} - \theta).^2)$

  if $J <$ goal value of cost **then**

    `// Optimization completed`

    break

  Perturb the parameter ϕ
  $\phi^p \longleftarrow randn(1, length(v_\phi(1,:)))$

  for $i = 1$ to length$(v_\phi(k,:))$ **do**

    Compute equation (2.1) to obtain the perturbed matrix $Q_p(s)$:

    $Q_p(s) \longleftarrow Q_p(s) + (v_\phi(k,i) + \phi^{pi}) * \left(\frac{\alpha}{s+\alpha}\right)^{(i-1)}$

  Compute controller in (2.5) using $Q_p(s)$

  Run: *Simulink* of the rocket model

  Output: Attitude angle θ_p

  Compute closed-loop transfer function in (2.8) using $Q_p(s)$

  Compute cost function of the perturbed response as shown in (2.2):
  $J_p(k) \longleftarrow sum((\theta_{goal} - \theta_p).^2)$

  Adjust the learning parameter η with respect to the cost:

  Compute percentage increase:
  $cost\_var \longleftarrow \frac{J(k)-J(k-1)}{J(k-1)} * 100$

Algorithm 5.1: Episodic REINFORCE Optimization Algorithm for the Rocket Model (continued)

for k = 1 to N **do**

  if $cost\_var >= 100$ **then**

    `// cost increased by an order of magnitude`

    Decrease the learning parameter η

    $flag \longleftarrow 1$

    $rand\_factor \longleftarrow rand()$

  if $rand\_factor >= 0.99$ **and** $flag = 1$ **then**

    $flag \longleftarrow 0$

    `// 1% of the times the algorithm is "blind" to cost increase`

  if $k > (N - N * 0.05)$ **then**

    `// The learning parameter reduces with the step when there are 5% iterations left`

    $v_\eta(k) \longleftarrow v_\eta(k)/(N - (N - (N * 0.05))$

  if $\eta <$ minimum value of η **then**

    $\eta \longleftarrow$ minimum value of η

  Detect the maximum and minimum values of cost (J) and save the correspondent value of parameter (ϕ)

  Normalize the step gain:

  $eta\_norm \longleftarrow v_\eta(k)/J(k)$

  if $flag = 0$ **then**

    Update the parameter as in (3.4):

    $v_\phi(k+1,:) \longleftarrow v_\phi(k,:) - (eta\_norm) * (J_p - J(k)) * \phi^p;$

  if $flag = 1$ **then**

    `// Next` ϕ `and current` J `go back to the previous value`

    $v_\phi(k+1,:) \longleftarrow v_\phi(k-1,:)$

    $J(k) \longleftarrow J(k-1)$

  Run: Conjugate Gradient Method every 3 iterations

  $grad\_counter \longleftarrow grad\_counter + 1$

  if $grad\_counter = 3$ **then**

    Update the parameter:

    $v_\phi(k+1,:) \longleftarrow v_\phi(k,:) + (v_\phi(k,:) - v_\phi(k-2,:))$

    $grad\_counter \longleftarrow 0$

Save: file containing the learning data

6

Conclusion

Contents

6.1 Conclusions

This thesis extends the results in [1] by applying the YP to more complex and unstable systems, and using RL for feedback design systems when instability can disrupt the learning process, such as in real-world systems. The YP guaranteed the stability and convexity needed to apply learning to the unstable rocket model.

The decision to add an initial stabilizing LQG controller to the Youla controller (2.5) was of utmost significance for complex systems. The LQG ensures stability and robustness during the initial stages of control before the Youla controller, and both of these elements combined increase the total reliability of the control system.

The YP can lead to additional complexity and increase the order of the control problem, and RL algorithms typically struggle with high-dimensional action spaces. Since the number of possible actions increases exponentially with dimensionality, RL algorithms, especially stochastic optimization algorithms, often struggle to find an optimal policy in high-dimensional action spaces, which results in increased convergence time.

The RL algorithm used represents a general class of stochastic algorithms called Episodic REINFORCE. The lack of a general theory of convergence that applies to this class of algorithms, as well as their apparent propensity for convergence to local minima, are the main disadvantages of REINFORCE algorithms.

The MATLAB unconstrained optimization problem solver *Fminunc* was used to validate the results obtained with the learning algorithm, since when both techniques can find the same solution it confirms the reliability and validity of the reinforcement learning algorithm in solving the optimization problem. When the REINFORCE algorithm does not converge to the same global minimum as the solver *Fminunc* it is important to choose a better random initialization of the parameters ϕ or, in some cases, use a richer input signal that provides both systems with more data for the optimization. In some circumstances, increasing the number of REINFORCE algorithm iterations or the complexity of the Youla parameter Q by increasing the parameter's ϕ dimensionality until the order of the controller becomes unreasonable may be able to make the optimization problem feasible.

Applying episodic REINFORCE to reinforcement learning tasks requires a thorough understanding of the distinction between straightforward optimization problems, for which the algorithm can quickly find efficient policies, and difficult optimization problems, which are challenging and call for advanced techniques.

It was found that when the relationship between different policies and their cost function value is smooth, *i.e.* does not present significant irregularities or local minima, it is easier for the REINFORCE algorithm to converge to the global minimum. Furthermore, when the dimension of the optimization problem is smaller the action space is also smaller, allowing for a more efficient search for the desired

optima.

The rewards obtained from the environment may have high variance or contain significant amounts of noise, which makes it harder for this class of REINFORCE algorithms to estimate the quality of the actions and obtain stable, reliable updates, making the convergence slower.

Stochastic algorithms, such as Episodic REINFORCE, are sensitive to the initialization and random noise factors during the optimization. Hence, it was found that testing different random initializations with the *Fminunc* function prior to the REINFORCE optimization helped speed the process of finding a suitable initialization.

A method discovered through research to speed the convergence and avoid getting stuck in local minima consisted of finding a suitable learning rate and alternating between the REINFORCE update and a traditional approach, such as the conjugate gradient method update. The use of a variation of the conjugate gradient method, which is used to find the minimum of quadratic functions, resulted in an 80% reduction of the convergence time in certain cases, since it helped the REINFORCE algorithm, which relies on the randomness of each episode, to have a more efficient search within the optimization environment.

Overall, the results of combining the YK parameterization with a learning algorithm were successful to solve the proposed attitude control problem of landing an RLV, using a well-defined model and cost function, as well as adding an initial stabilizing controller for robustness. It is encouraging to apply the research of the YK parameterization presented in the literature to real-world circumstances because of its reliability. However, the REINFORCE algorithm has to be strengthened to allow for confident learning applications to hardware.

6.2 Future Work

Although some encouraging results have been found, there is still area for improvement and plenty of potential for further, in-depth investigation. Some suggestions for future research are:

- Accounting for the fuel burn, derived in equation (4.6), that causes mass variation in the rocket model control system, is essential for enhancing performance and stability. This also makes the model more applicable to real-world situations, and safe.

- Some aerodynamic forces such as drag, *i.e.* the resistance encountered by the rocket as it moves through the atmosphere, were ignored. Hence, future research would include a more complex mathematical model for rocket landing.

- Currently, the rocket has an attitude controller and an altitude controller to ensure maintenance of the nominal trajectory and a vertical landing. However, future developments may include the simul-

taneous implementation of multiple controllers to improve the performance of the rocket. Specifically, additional attitude controls may be used to meet the specific requirements before and after the "switch" time t_{ON} on the *bang-bang* thrust profile, when the rocket's thrust is activated. This approach would provide more precise and efficient control over the attitude of the rocket throughout its flight profile, and prevent heavy oscillations on the gimbal angle φ, for the first seconds.

- Expanding the research focus from SISO control systems to MISO control systems. By considering multiple input variables and their effects on the system output, this extension allows for a more comprehensive analysis of complex control systems, leading to more robustness, and improved adaptation to real-world scenarios.
- The learning process requires a cost function that is more clearly defined, with specific penalties and rewards for conditions that are significant to the optimization issue. Appropriate weights must also be chosen to appropriately balance the respective importance of these costs.
- Further investigate the learning algorithm Episodic REINFORCE to make it more reliable for more complex, high-order systems, less dependent on initial conditions, and test different techniques to improve convergence.

Bibliography

[1] J. Roberts, I. Manchester, and R. Tedrake, "Feedback controller parameterizations for reinforcement learning," in *2011 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, 05 2011, pp. 310 – 317.

[2] J. P. Hespanha, "Linear systems theory," Princeton Press, 2009.

[3] NASA, "Gimbaled thrust." [Online]. Available: https://www.grc.nasa.gov/WWW/k-12/rocket/gimbaled.html

[4] R. Ferrante, "A robust control approach for rocket landing," in *M.S. thesis, School of Informatics University of Edinburgh*, 2017, p. 78. [Online]. Available: https://project-archive.inf.ed.ac.uk/msc/20172139/msc_proj.pdf

[5] R. Sutton and A. Barto, "Reinforcement learning, vol.2," 2018.

[6] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning. machine learning, vol. 8, no. 3–4," 05 1992, p. 229–256.

[7] J. B. D. Youla and H. Jabr, "Modern wiener–hopf design of optimal controllers part i: The single-input-output case," 02 1976, pp. 3–13.

[8] D. Youla, H. Jabr, and J. Bongiorno, "Modern wiener-hopf design of optimal controllers–part ii: The multivariable case," *IEEE Transactions on Automatic Control*, vol. 21, no. 3, pp. 319–338, 1976.

[9] V. Kučera, "Stability of discrete linear feedback systems," 1975, p. 573–578.

[10] V. Kučera, "A method to teach the parameterization of all stabilizing controllers," 18th IFAC World Congress. Italy, Milan, 2011.

[11] I. M. T.-T. Tay and J. B. Moore, "High performance control," Boston: Birkha¨user, 1989.

[12] S. M. Dale, W.N., "Stabilizability and existence of system representations for discrete-time time-varying systems. siam j. control optimization, 31," 1993, pp. 1538–1557.

[13] J. M. C. Desoer, Ruey-Wen Liu and R. Saeks, “Feedback system design: The fractional representation approach to analysis and synthesis,” in *IEEE Transactions on Automatic Control, vol. 25, no. 3,*, 06 1980, pp. 399–412.

[14] A. Quadrat, “On a generalization of the youla-kuera parametrization. part i: the fractional ideal approach to siso systems,” *Syst. Control. Lett.*, vol. 50, pp. 135–148, 2003.

[15] J. Hammer, “Nonlinear system stabilization and coprimeness,” *Int. Journal of Control*, vol. 44, pp. 1349–1381, 1985.

[16] A. Paice and J. Moore, “On the youla-kucera parametrization for nonlinear systems,” *Systems Control Letters*, vol. 14, no. 2, pp. 121–129, 1990.

[17] B. D. ANDERSON, “From youla–kucera to identification, adaptive and nonlinear control,” *Automatica*, vol. 34, no. 12, pp. 1485–1506, 1998.

[18] I. Mahtout, F. Navas, V. Milanes, and F. Nashashibi, “Advances in youla-kucera parametrization: A review,” *Annual Reviews in Control*, vol. 49, pp. 81–94, 2020.

[19] D. Wang and X. Chen, “A tutorial on loop-shaping control methodologies for precision positioning systems,” *Advances in Mechanical Engineering*, vol. 9, no. 12, p. 1687814017742824, 2017.

[20] S. P. Boyd and C. H. Barratt, “Linear controller design : limits of performance,” in *Englewood Cliffs, N.J.: Prentice Hall*, 1991.

[21] S. Boyd, C. Baratt, and S. Norman, “Linear controller design: limits of performance via convex optimization,” *Proceedings of the IEEE*, vol. 78, no. 3, pp. 529–574, 1990.

[22] L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: A survey,” *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[23] J. Kober, J. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” *The International Journal of Robotics Research*, vol. 32, pp. 1238–1274, 09 2013.

[24] Q. Gan, F. Wu, and J. Zhao, “A survey of research on stability guarantee of reinforcement learning automatic control problem,” in *2021 6th International Conference on Control, Robotics and Cybernetics (CRC)*, 2021, pp. 240–249.

[25] S. R. Friedrich and M. Buss, “A robust stability approach to robot reinforcement learning based on a parameterization of stabilizing controllers,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3365–3372.

[26] J. P. Hespanha and A. Morse, "Switching between stabilizing controllers," *Automatica*, vol. 38, no. 11, pp. 1905–1917, 2002.

[27] G. Zames, "Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms, and approximate inverses," *IEEE Transactions on Automatic Control*, vol. 26, no. 2, pp. 301–320, 1981.

[28] A. Rantzer and A. Megretski, "A convex parameterization of robustly stabilizing controllers," *IEEE Transactions on Automatic Control*, vol. 39, no. 9, pp. 1802–1808, 1994.

[29] W.-M. Lu, "A state-space approach to parameterization of stabilizing controllers for nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1576–1588, 1995.

[30] M. Rotkowitz and S. Lall, "A characterization of convex problems in decentralized control^{a}st," *IEEE Transactions on Automatic Control*, vol. 51, no. 2, pp. 274–286, 2006.

[31] Z. Bojun, L. Zhanchao, and L. Gang, "High-precision adaptive predictive entry guidance for vertical rocket landing," *Journal of Spacecraft and Rockets*, vol. 56, pp. 1–7, 09 2019.

[32] B. Wie, W. Du, and M. Whorton, "Analysis and design of launch vehicle flight control systems," *AIAA Guidance, Navigation and Control Conference and Exhibit*, 08 2008.

[33] S. K. Kumar Bysani, A. Karpur, and N. Arun, "Vertical landing rockets," *TMAL02 Expert Conference*, vol. 8, no. 4, pp. 33–34, Jan. 2020. [Online]. Available: https://conference.ep.liu.se/index.php/TMAL02/article/view/555

[34] L. Blackmore, "Autonomous precision landing of space rockets," vol. 46, pp. 15–20, 01 2016.

[35] M. Sagliano, T. Tsukamoto, S. Ishimoto, S. Farì, E. Dumont, D. Seelbinder, M. Schlotterer, A. Heidecker, J. A. Macés-Hernández, and S. Woicke, "Robust control for reusable rockets via structured h ∞ synthesis," 06 2021.

[36] K. Liu, F. Liu, S. Wang, and Y. Li, "Finite-time spacecraft's soft landing on asteroids using pd and nonsingular terminal sliding mode control," *Mathematical Problems in Engineering*, vol. 2015, pp. 1–10, 01 2015.

[37] J. M. Lemos, "Controlo no espaço de estados," IST Press, 2019.

[38] C. Research, "Cvx: Matlab software for disciplined convex programming." [Online]. Available: http://cvxr.com/cvx/

[39] Matlab, "Create optimization options (optimoptions)." [Online]. Available: https://www.mathworks.com/help/optim/ug/optim.problemdef.optimizationproblem.optimoptions.html

[40] Matlab, “Find minimum of unconstrained multivariable function (fminunc).” [Online]. Available: https://www.mathworks.com/help/optim/ug/fminunc.html

[41] “An introduction to the conjugate gradient method without the agonizing pain.” [Online]. Available: https://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf

[42] W. B. Powell, “Reinforcement learning and stochastic optimization: A unified framework for sequential decisions,” March 15, 2022, chapter 6.

[43] M. Liu, W. Zhang, F. Orabona, and T. Yang, “Adam$^+$: A stochastic method with adaptive variance reduction,” 11 2020.

[44] K. P. An. Packard and R. Horowitz, “Jacobian linearizations, equilibrium points,” in *Dynamic Systems and Feedback*, 2002, p. 169–200.

[45] SpaceX, “Falcon 9,” 2021. [Online]. Available: https://www.spacex.com/vehicles/falcon-9/

[46] SpaceX, “Falcon 9 user’s guide,” September 2021, p. 9.

[47] L. I. Institute, “Definition: Nominal from 14 cfr § 401.7 — lii / legal information institute.” [Online]. Available: https://www.law.cornell.edu/definitions/index.php?width=840&height=800&iframe=true&def_id=8938ea75c026a86d70d7e542f095b6f7&term_occur=999&term_src=Title:14:Chapter:III:Subchapter:A:Part:401:401.7

A

Extra Example

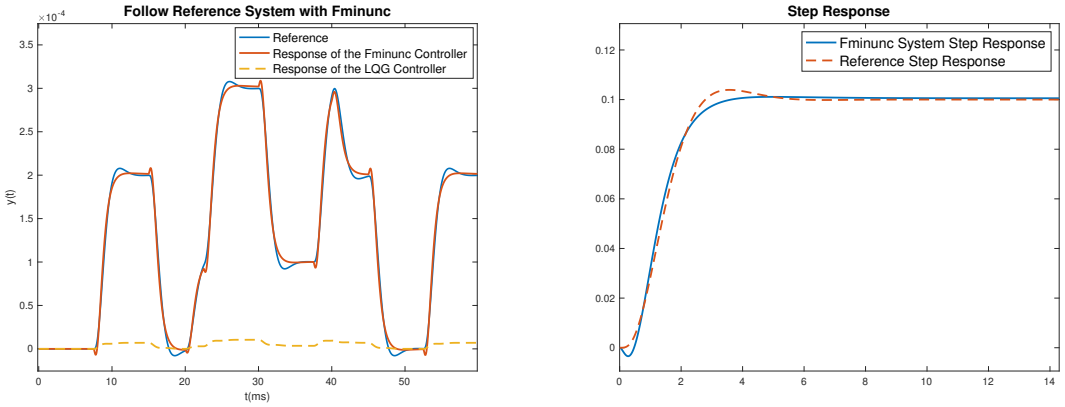
Consider the transfer function of the fourth-order closed-loop control system $T(s)$ and third-order Plant $P(s)$

$$T(s) = \frac{8}{s^4 + 16s^3 + 70s^2 + 108s + 80}, \quad P(s) = \frac{3s + 6}{s^3 + 3s^2 + 7s + 5}. \tag{A.1}$$

This example serves to demonstrate that it is possible to find all the stabilizing controllers that optimize a high-order Plant $P(s)$ to track a high-order reference $T(s)$ as well, using the optimization algorithm detailed in section 3.1 *Fminunc*.

Figure A.1(a) illustrates the controlled system's response to the impulse in comparison to the reference response when $\alpha = 1$, and the starting vector of parameters ϕ^0 has dimension 4. This demonstrates that the controller stabilizes the system and follows the reference, as seen in figure A.1(b), with a steady-state error of 1.

The initial SSE of the baseline LQG controller is, approximately, 0.0016. The cost obtained using the solver *Fminunc* is 1.93×10^{-6}, which represents a 99.87% cost optimization.



(a) Response of the resulting controller compared to the reference transfer function.

(b) Step response of the resulting controller compared to the reference.

Figure A.1: Optimization results obtained with *Fminunc* for a high-order system.