Robust Control Design and Analysis for Small Fixed-Wing Unmanned Aircraft Systems using Integral Quadratic Constraints

Mark C. Palframan

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Aerospace Engineering

Mazen Farhood, Chair
Craig A. Woolsey
Mayuresh J. Patil
Christopher J. Roy

June 21, 2016
Blacksburg, Virginia

Keywords: Unmanned Aircraft Systems, Robust Control, Worst-Case Analysis, Integral Quadratic Constraints, Uncertainty Analysis, Path-Following

Robust Control Design and Analysis for Small Fixed-Wing Unmanned Aircraft Systems using Integral Quadratic Constraints

Mark C. Palframan

The main contributions of this work are applications of robust control and analysis methods to complex engineering systems, namely, small fixed-wing unmanned aircraft systems (UAS). Multiple path-following controllers for a small fixed-wing Telemaster UAS are presented, including a linear parameter-varying (LPV) controller scheduled over path curvature. The controllers are synthesized based on a lumped path-following and UAS dynamic system, effectively combining the six degree-of-freedom aircraft dynamics with established parallel transport frame virtual vehicle dynamics. The robustness and performance of these controllers are tested in a rigorous `MATLAB` simulation environment that includes steady winds, turbulence, measurement noise, and delays. After being synthesized off-line, the controllers allow the aircraft to follow prescribed geometrically defined paths bounded by a maximum curvature. The controllers presented within are found to be robust to the disturbances and uncertainties in the simulation environment. A robust analysis framework for mathematical validation of flight control systems is also presented. The framework is specifically developed for the complete uncertainty characterization, quantification, and analysis of small fixed-wing UAS. The analytical approach presented within is based on integral quadratic constraint (IQC) analysis methods and uses linear fractional transformations (LFTs) on uncertainties to represent system models. The IQC approach can handle a wide range of uncertainties, including static and dynamic, linear time-invariant and linear time-varying perturbations. While IQC-based uncertainty analysis has a sound theoretical foundation, it has thus far mostly been applied to academic examples, and there are major challenges when it comes to applying this approach to complex engineering systems, such as UAS. The difficulty mainly lies in appropriately characterizing and quantifying the uncertainties such that the resulting uncertain model is representative of the physical system without being overly conservative, and the associated computational problem is tractable. These challenges are addressed by applying IQC-based analysis tools to analyze the robustness of the Telemaster UAS flight control system. Specifically, uncertainties are characterized and quantified based on mathematical models and flight test data obtained in house for the Telemaster platform and custom autopilot. IQC-based analysis is performed on several time-invariant $\mathcal{H}_\infty$ controllers along with various sets of uncertainties aimed at providing valuable information for use in controller analysis, controller synthesis, and comparison of multiple controllers. The

proposed framework is also transferable to other fixed-wing UAS platforms, effectively taking IQC-based analysis beyond academic examples to practical application in UAS control design and airworthiness certification. IQC-based analysis problems are traditionally solved using convex optimization techniques, which can be slow and memory intensive for large problems. An oracle for discrete-time IQC analysis problems is presented to facilitate the use of a cutting plane algorithm in lieu of convex optimization in order to solve large uncertainty analysis problems relatively quickly, and with reasonable computational effort. The oracle is reformulated to a skew-Hamiltonian/Hamiltonian eigenvalue problem in order to improve the robustness of eigenvalue calculations by eliminating unnecessary matrix multiplications and inverses. Furthermore, fast, structure exploiting eigensolvers can be employed with the skew-Hamiltonian/Hamiltonian oracle to accurately determine critical frequencies when solving IQC problems. Applicable solution algorithms utilizing the IQC oracle are briefly presented, and an example shows that these algorithms can solve large problems significantly faster than convex optimization techniques. Finally, a large complex engineering system is analyzed using the oracle and a cutting-plane algorithm. Analysis of the same system using the same computer hardware failed when employing convex optimization techniques.

This material is based upon work supported by the National Science Foundation under Grant Number CMMI-1351640, the U.S. Navy Naval Air Systems Command, and the Virginia Tech Institute for Critical Technology and Applied Science.

Acknowledgments

I would like to thank the Aerospace & Ocean Engineering Department at Virginia Tech for all the opportunities I've gotten over the last 5 years. In particular I would like to thank my advisor, Dr. Mazen Farhood. Without his guidance, patience, and support, the work presented in this dissertation would not have been possible. Dr. Farhood has taught me to employ a rigorous and thorough approach to engineering problems, and has made me a better mathematician, control designer, and engineer. I would also like to thank Dr. Craig Woolsey for his guidance and support in addition to being instrumental in the development of both the Nonlinear Systems Lab and the Kentland Experimental Aerial Systems Lab. Additionally, I would like to thank my committee members Dr. Mayuresh Patil and Dr. Christopher Roy for their suggestions and support.

I would like to thank my current and former colleagues in the Nonlinear Systems Lab whom I have collaborated with, bounced ideas off of, and worked alongside for the last 5 years. Specifically I would like to thank John Cianchetti for teaching me how to fly, Dr. Ony Arifianto and Dr. David Grymin for spearheading the Telemaster platform, Jeffrey Garnand for his assistance as a fantastic ground station operator, Dr. Artur Wolek for his mathematical suggestions and support, Deva Prakesh for all his efforts in control design and aircraft maintenance as part of our two-man flight crew, and Dr. Andrew Rogers for his camaraderie and support.

I have seen the Nonlinear Systems Lab and SPAARO aircraft go through a lot of changes over the last five years, and I hope that I have left the Nonlinear Systems Lab a better place than when I first started there back in 2011.

Last, but definitely not least, I would like to thank my wife, family, and friends for their love and support throughout the Ph.D. process. I couldn't have done it without them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Topics such as linear parameter-varying (LPV) control and integral quadratic constraint (IQC) analysis have been applied throughout the literature. In this work, LPV control and IQC-based analysis have been successfully applied to complex engineering systems. Specifically, a small fixed-wing unmanned aircraft system (UAS) is used throughout this work. Parts of this work have been previously published as conference proceedings. The work can be divided into three subtopics of UAS robust control: synthesis and testing of an LPV path-following controller [1], development of an IQC analysis framework [2], and the formulation of a robust oracle for solving large IQC problems.

1.1 Path-Following Control for UAS

One of the challenges associated with small fixed-wing UAS is operating effectively in the presence of relatively significant environmental disturbances, namely winds, gusts, and turbulence. Owing to their size, small unmanned aircraft are affected much more dramatically than traditional aircraft in the presence of what may be considered small environmental disturbances, often leading to poor performance by traditional trajectory tracking methods with time-stamped inertial position feedback. Path-following control methods, by guiding an aircraft to converge to and follow a geometric path in space specified without time parametrization, can potentially handle stronger disturbances than trajectory tracking methods [3]. In fact, [4] shows that path-following control is often able to remove performance limitations present in traditional reference tracking methods.

Path-following control has been shown to have many useful applications to UAS, involving missions related to surveillance, imaging, formation flight, and station keeping [5]-[7]. Flying various loiter patterns, for instance, is necessary to collect sparsely distributed airborne contaminants [8]. Alternatively, UAS in urban environments must maintain their position on the desired path despite the presence of significant disturbances in order to avoid collisions with the surrounding infrastructure.

Notable approaches to path-following control include waypoint guidance [9] and the use of vector fields to drive the vehicle towards the desired path [10], [11]. Among others, this work utilizes a virtual vehicle formulation [12], [13], whereby the controller strives to minimize

the error between the ownship and a fictitious vehicle constrained to the path. Specifically, this work strives to extend the formulation in [12] by incorporating it into an LPV $\mathcal{H}_\infty$ framework.

In general, the control strategy developed in this work utilizes the fact that control of the vehicle attitude can drive the vehicle towards a desired position, in this case along a geometric path in space. The designed low-level controllers need only a curvature bounded geometrically defined path as input to accurately track applicable paths at a constant speed profile in the midst of atmospheric and other disturbances. As the path-following controllers presented in this work are simply parameterized by path curvature, they can be synthesized offline. Any path bounded by curvature simply needs to be uploaded to the UAS, and it can be subsequently tracked by the onboard controller.

At the cost of increased complexity through the introduction of path-following dynamics into the overall system, a virtual vehicle approach can be considered the most flexible path-following approach in terms of the geometric paths that can be followed. In fact, by incorporating key assumptions into the approach developed in [12], the approach in this work effectively combines the path-following and vehicle dynamics in a way that the resulting system has the same number of states as the pure vehicle dynamics.

While much of the existing literature on path-following employs nonlinear control methods, such as backstepping, sliding mode, and adaptive control, e.g., see [12], [14], an $\mathcal{H}_\infty$ approach is employed in this work. Specifically, linear time-invariant (LTI) and LPV controllers are designed using the $\mathcal{H}_\infty$ norm as the performance measure. This linear framework allows

the method to take advantage of $\mathcal{H}_\infty$ tools from the vast literature on robust control. For instance, the formulation presented within is easily adaptable to formal validation techniques and the incorporation of uncertain initial conditions and other uncertainties into the synthesis process [2], [15].

In this work, we partially adopt the virtual vehicle formulation proposed by Kaminer et al. [12], where a nonlinear backstepping outer-loop controller, based on the dynamics of a parallel transport frame, prescribes pitch and yaw rates, which are then tracked by a stabilizing inner-loop controller. By incorporating key assumptions into the approach developed in [12], the approach herein effectively combines the path-following and vehicle dynamics in a way that the resulting system has the same number of states as the pure vehicle dynamics. In addition to the aforementioned benefits of $\mathcal{H}_\infty$ control, this lumped dynamics approach utilizes far fewer tuning parameters, which the authors found to be much more intuitive to use than those involved in [12].

1.2 IQC Analysis for Small Fixed-Wing UAS

An uncertainty analysis framework is presented to aid in the airworthiness certification of UAS controllers, quickly compare various controllers, and guide the design process to produce controllers which are robust against modeling inaccuracies, nonlinearities, and external disturbances. As reported by the US Department of Defense in 2002, 26% of all recorded UAS mishaps are due to flight controller issues, second only to power failures [16]. In ad-

dition to external disturbances, variations in aircraft construction, damage from field use, the addition of small payloads, and modifications to the airframe all have the potential to negatively impact the controller performance if the UAS is not remodeled to incorporate them, and the corresponding controller redesigned.

By creating a unified framework for uncertainty characterization, quantification, and analysis of fixed-wing UAS controllers, the aircraft system's robustness to disturbances, nonlinearities, and modeling inaccuracies may be quickly and inexpensively assessed. Such assessments may reduce the disparity between the relatively low manufacturing costs for small UAS and the large verification and validation costs associated with aerospace platform development [17]. The rigorous approach to system analysis presented in this work has the potential to expedite the platform validation process without relying on extensive Monte Carlo simulations, wind tunnel, and flight testing. To accomplish this, robust control techniques and integral quadratic constraint (IQC) based analysis tools are leveraged to check for robust stability and provide upper bounds on the worst case controller performance. By serving as a pre-screening tool for certification, these performance bounds can help identify the system's sensitivity to selected sets of uncertainties, which may be utilized in the controller synthesis process to improve system robustness.

First developed by Megretski and Rantzer, IQC theory provides a powerful analysis framework for simultaneous inclusion of several uncertainty types [18]. As a result, IQC-based analysis allows semidefinite programming techniques to be used to quickly obtain guaranteed upper bounds on the ℓ_2-gain performance level of an uncertain closed-loop aircraft system.

Whereas the mature μ-analysis is restricted to linear time-invariant uncertainties, the IQC approach is much more flexible and handles a wide range of uncertainties, including: static and dynamic, linear time-invariant and linear time-varying (LTV) perturbations, time delays, and sector-bounded nonlinearities [19]. These uncertainties can be easily combined and manipulated in any rational combination by representing the uncertain dynamic system as a linear fractional transformation (LFT) on uncertainties. The Linear Fractional Representation (LFR) Toolbox for `MATLAB`, for instance, could be used to formulate LFRs from uncertain linear systems [20].

To date, IQC-based analysis has been applied to several aircraft-related academic examples, typically focused on simplified longitudinal models with select modeled uncertainties [21]-[23]. For instance, the effects of individual uncertainties on a longitudinal NASA re-entry vehicle model are analyzed in [24], a longitudinal model of an aeroelastic aircraft with a dynamic input uncertainty is analyzed in [25], a two-state longitudinal ONERA fighter model is analyzed with uncertainties representing flight envelope, static parameters, and aerodynamic sub-coefficients in [26], and the effect of saturation on NASA's Generic Transport Model is analyzed in [27]. While theoretical development is still ongoing, IQC-based analysis has a strong and relatively mature theoretical basis. From an implementation perspective, IQC-based analysis methods have not previously been shown to be applicable to full 6-degree-of-freedom (DOF) aircraft models while covering all major uncertainties.

While the creation of the proposed uncertainty framework may appear simple given the available theory and semidefinite programming tools, several difficulties manifest themselves

in this problem. The contributions and challenges of this uncertainty framework are as follows:

- Uncertainties are classified for a 6 DOF UAS such that they balance the computational complexity of the resulting analysis problem with the conservativeness of the results. For instance, several related uncertainties lumped together into a fewer number of uncertainty representations would result in a more computationally manageable analysis problem, but consequently, a more conservative worst-case performance bound. On the other hand, representing the same system as many interconnected uncertainties might yield a lower performance bound, but may cause the semidefinite program to become computationally intractable, as the number of optimization variables in the Lyapunov stability matrix grows quadratically with the order of the state-space representation [21].

- The typical assumption that the aircraft center of gravity (CG) is affixed to the origin of the body-fixed reference frame is relaxed to allow the unknown CG location (restricted to a longitudinal plane) to be incorporated into the system model as an uncertainty, effectively coupling the aerodynamic forces and moments in the 6 DOF rigid body equations of motion. Considering all parametric uncertainties, including uncertain mass, moments of inertia, and CG, results in a very large LFR. An appropriate model reduction procedure is presented.

- Uncertainties representing nonlinear and unmodeled portions of the system's aerody-

namics are captured using a small number of static time-varying perturbations based on collected flight test data. Uncertainties representing controller delays, unmodeled actuator and thrust dynamics, saturations, and other nonlinearities are quantified as a set of dynamic LTI and static LTV perturbations.

- The UAS uncertainty framework is implemented for three tasks: analysis of a UAS controller with respect to several uncertainty groups, comparison of the robust performance of three designed $\mathcal{H}_\infty$ controllers, and aiding in the controller design and tuning process.

The uncertainty classification choices presented herein correspond to specific IQC-multipliers available in the literature that the authors found to best reduce conservatism while still covering the applicable uncertainty range. Additionally, the uncertainties have been chosen such that they can be easily quantified without requiring extensive additional testing.

The framework is applicable to any LTI or LPV controller, including linear quadratic regulators, PID, $\mathcal{H}_\infty$, and μ-synthesized controllers, which have been found to be effective and highly used in fixed-wing UAS flight control [1], [15], [28], [29].

1.3 A Discrete-Time IQC Oracle

The IQC framework is a valuable tool for uncertainty analysis of complex engineering systems [2]. Through application of the Kalman-Yakubovich-Popov (KYP) lemma (see Section

4.4), the frequency-dependent, infinitely constrained IQC analysis problem can be equivalently posed as a frequency-independent, finite dimensional convex optimization problem, and solved using robust semidefinite programming tools, such as `SDPT3` [18], [30]. Unfortunately, the KYP-based solution to medium and large sized IQC problems can be computationally expensive and slow to converge [31]. This has led to the development of non-KYP-based methods to solve IQC analysis problems involving frequency-gridding [21], [32], [33], or cutting plane algorithms [31], [34]. These alternative fast IQC algorithms consist of two main parts: an algorithm to generate a candidate solution, and an oracle to determine if the candidate solution satisfies the IQC inequality. By exploiting the structure of the problem, the IQC oracle quickly checks if the IQC inequality holds for its infinite set of constraints, and if not, returns violated constraints to be incorporated into the algorithm for generating new candidate solutions. Instead of relying on a Lyapunov stability matrix to represent the infinite number of frequency constraints on the system, as the KYP solution does, the IQC oracle is posed as an eigenvalue problem. The eigenvalue problem can be solved faster than its convex counterpart for medium-large problems. This allows IQC problems to be solved in a much less memory intensive fashion and, depending on the dimensions of the problem, faster.

Similar Hamiltonian eigenvalue problems are frequently used when solving for $\mathcal{H}_\infty$ norms. A bisection routine, for example, is commonly used to converge to an upper bound on the standard $\mathcal{H}_\infty$ norm by calling an oracle to check candidate norm bounds at each iteration [35], [36]. Hamiltonian-based oracles for the continuous-time IQC problem can also be found

throughout the literature [21], [31]-[34]. In this work, we present an oracle for the discrete-time IQC problem. As many control systems, such as the one in [2], are implemented in discrete-time, a discrete-time version of the IQC oracle is important for the analysis of practical engineering systems. To the authors' knowledge, the discrete-time formulation of the IQC oracle is not available in the literature. We specifically deal with real-valued dynamic systems in this work, although extensions to the complex case are possible. The IQC oracle presented in Chapter 6 is extended and manipulated to result in an eigenvalue problem applied to a skew-Hamiltonian/Hamiltonian matrix pencil. This specially structured pencil can be used in order to improve the robustness of eigenvalue calculations and allow structure preserving eigenvalue solvers to be applied. Such eigensolvers would allow the purely imaginary eigenvalues of interest to be calculated with zero error in the real part [37].

1.4 Overview and Contributions

Chapter 2 presents an overview of notation used throughout this work. The equations of motion for a fixed-wing rigid body aircraft are derived. Additionally, specific information on the aircraft platform and simulation environment used are presented. Finally, the synthesis equations for time-invariant and parameter-varying $\mathcal{H}_\infty$ control are given.

Chapter 3 presents the design of three robust path-following controllers for a fixed-wing UAS, including an LPV $\mathcal{H}_\infty$ controller. All three controllers are designed and synthesized offline, and can be used to track any geometrically defined path bounded by a maximum curvature

without requiring any additional synthesis. Specifically, the designed LPV controller provides an example of effective application of LPV control techniques to a real-world complex engineering system. The simulation results of each controller as applied to a variety of paths are also presented.

Chapter 4 provides a brief overview of IQC theory, including the IQC multipliers used in this work and an application of the Kalman-Yakubovich-Popov lemma. The presented IQC theory is referenced in the subsequent two chapters.

Chapter 5 presents an IQC-based uncertainty analysis framework for fixed-wing UAS. Appropriate uncertainty types are modeled in order to balance the conservatism and computational complexity of the resulting IQC analysis problem, and cover all the prominent uncertainties and nonlinearities. Additionally, the uncertainty framework is applied to a small fixed-wing UAS platform. Details and suggestions concerning uncertainty quantification are provided in addition to a model reduction procedure. Finally, the utility of the analysis framework is shown through analysis of a single controller, comparison of multiple controllers, and a controller tuning procedure for the modeled UAS platform.

Chapter 6 presents an oracle as an alternative method to solve discrete-time IQC problems. Two algorithms for solving discrete-time IQC problems using the oracle are briefly discussed. Implementation details and pseudocode for a cutting plane algorithm are presented. Techniques to improve the robustness of eigenvalue calculations within the oracle are applied to the discrete-time IQC oracle. Finally, the advantages of the oracle for fast solution times and application to large complex engineering systems are presented through two examples.

Finally, conclusions and areas of future work for the above topics are presented in Chapter 7.

Chapter 2

Background

This chapter presents the notation, equations, and controller synthesis algorithms that will be used in this dissertation. Section 2.1 first presents notation that will be used in this chapter and the chapters that follow. The equations of motion for a rigid-body aircraft are derived in Section 2.2, and details on the employed `MATLAB` simulation environment are presented in Section 2.3. Aerodynamic and other parameters for the Telemaster UAS platform can be found in Section 2.4. Finally, LTI and LPV formulations for $\mathcal{H}_\infty$ controller synthesis are given in Section 2.5.

2.1 Notation

The notation used is mostly standard. The set of complex vectors of dimension n, real vectors of dimension n, real-valued $n \times n$ symmetric matrices, real-valued skew-symmetric

$n \times n$ matrices, real-valued $n \times n$ upper-triangular matrices, and the set of non-negative integers are denoted by $\mathbb{C}^n$, $\mathbb{R}^n$, $\mathbb{S}^n$, $\mathbb{SK}^n$, $\mathbb{T}^n$, and $\mathbb{Z}_+$ respectively. The unit imaginary number is denoted as $j = \sqrt{-1}$. The transpose, adjoint, maximum singular value, and largest eigenvalue (when all eigenvalues are real) of a matrix X are given as X^T, X^*, $\bar{\sigma}(X)$, and $\lambda_{max}(X)$. Note that the eigenvalues of a Hermitian matrix are real. $X \preceq 0$ denotes that X is negative semidefinite. The normed space of square summable vector-valued sequences $\boldsymbol{x} = (\boldsymbol{x}(0), \boldsymbol{x}(1), \boldsymbol{x}(2), \ldots)$, with each $\boldsymbol{x}(k) \in \mathbb{R}^n$, is denoted by ℓ_2^n, and abbreviated to ℓ_2 when the dimension is evident or irrelevant to the discussion. Given $\boldsymbol{x} \in \ell_2^n$, its Fourier transform is $\hat{\boldsymbol{x}}$, and the ℓ_2 norm is defined as $\|\boldsymbol{x}\|_{\ell_2}^2 = \sum_{k=0}^{\infty} \boldsymbol{x}^T(k)\boldsymbol{x}(k) < \infty$, with k denoting time. The ℓ_2-induced norm of a bounded linear operator P mapping ℓ_2 to ℓ_2 is defined as $\|P\|_{\ell_2 \to \ell_2} = \sup_{0 \neq \boldsymbol{u} \in \ell_2} (\|P\boldsymbol{u}\|_{\ell_2} / \|\boldsymbol{u}\|_{\ell_2})$. The image and kernel of a linear map P are denoted as $\operatorname{Im} P$ and $\operatorname{Ker} P$, respectively. $\mathcal{RL}_\infty$ is the space of proper discrete-time real-rational transfer functions with no poles on the unit circle. $\mathcal{RH}_\infty \subseteq \mathcal{RL}_\infty$ contains stable functions with all poles strictly inside the unit circle. Given $G \in \mathcal{RH}_\infty$, the ℓ_2-induced norm, or $\mathcal{H}_\infty$ norm, is given by $\|G\|_\infty = \sup_{\omega \in [0, 2\pi]} \overline{\sigma}(G(e^{j\omega}))$.

Defining the symplectic matrix $\mathcal{J} \in \mathbb{SK}^m$ as

$$\mathcal{J} = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix},$$

where $m = 2n$, we call a matrix $X \in \mathbb{R}^{m \times m}$ Hamiltonian if $X\mathcal{J} = \mathcal{J}^T X^T$ and skew-Hamiltonian if $X\mathcal{J} + \mathcal{J}^T X^T = 0$. A matrix $\mathcal{U} \in \mathbb{R}^{m \times m}$ is called orthogonal if $\mathcal{U}\mathcal{U}^T =$

$\mathcal{U}^T\mathcal{U} = I_m$ and orthogonal symplectic if it is orthogonal and $\mathcal{U}\mathcal{J}\mathcal{U}^T = \mathcal{J}$. A linear matrix pencil $X - \lambda Y \in \mathbb{R}^{m\times m}$ with $\lambda \in \mathbb{C}$ is called skew-Hamiltonian/Hamiltonian if $X \in \mathbb{R}^{m\times m}$ is Hamiltonian and $Y \in \mathbb{R}^{m\times m}$ is skew-Hamiltonian [38]. We let the spectrum $\sigma(X, Y)$ represent the set of unique values $\lambda \in \mathbb{C}$ (eigenvalues) of the pencil $X - \lambda Y$ such that $\det(X - \lambda Y) = 0$. We say that the pencil $X - \lambda Y$ is regular if there exists a λ such that $\det(X - \lambda Y) \neq 0$. Finally, we define two matrix pencils, introduced in [39], which have special block structures and properties. Namely, given $A, B \in \mathbb{R}^{n\times m}$ and $C \in \mathbb{S}^m$, the D-type pencil, $X_D - \lambda Y_D$, and the C-type pencil, $X_C - \lambda Y_C$, are defined as matrix pencils of the following forms:

$$X_D - \lambda Y_D = \begin{bmatrix} 0 & A \\ -B^T & C \end{bmatrix} - \lambda \begin{bmatrix} 0 & B \\ -A^T & 0 \end{bmatrix}, \quad X_C - \lambda Y_C = \begin{bmatrix} 0 & A \\ A^T & C \end{bmatrix} - \lambda \begin{bmatrix} 0 & B \\ -B^T & 0 \end{bmatrix}.$$

We use δ and Δ, along with a unique subscript, to denote causal scalar and full block uncertainties, respectively. In addition, these uncertainties may by static LTI (δ, Δ), dynamic LTI $(\delta(z), \Delta(z))$, or static time-varying $(\delta(k), \Delta(k))$, where z is a complex number and k denotes the time instant. Dynamic LTV uncertainties are not considered herein. s The upper LFT of M and an operator Δ is formally defined as $\Delta \star M = M_{22} + M_{21}\Delta(I - M_{11}\Delta)^{-1}M_{21}$, where $M = [M_{ij}]_{i=1,2;j=1,2}$. When Δ appears with no dependence on z, it refers to a block-diagonal operator consisting of uncertainties and the term $1/z$. A realization of the system $G(z) = D + C(zI - A)^{-1}B$, the Kronecker product of an $m \times n$ matrix X and the identity

matrix I, and a skew-symmetric version of vector $\boldsymbol{x} = [a\ b\ c]^T$ will be written as

$$G = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right],\ X \otimes I = \begin{bmatrix} x_{11}I \ldots x_{1n}I \\ \vdots \quad \ddots \quad \vdots \\ x_{m1}I \ldots x_{mn}I \end{bmatrix},\ \boldsymbol{x}^{\times} = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix}.$$

Finally, $\mathrm{diag}(X, Y, Z)$ denotes a block-diagonal representation of the matrices X, Y, and Z.

2.2 Rigid-Body Aircraft Equations of Motion

A detailed derivation of the rigid-body equations of motion for a fixed wing aircraft is available in [40], among others. Two reference frames are used to define the aircraft's motion, the Earth-fixed inertial reference frame and the body-fixed reference frame, denoted $\mathcal{F}_I$ and $\mathcal{F}_b$, respectively. The inertial reference frame has its origin on the surface of the Earth and has components of ($\boldsymbol{x}_I$, $\boldsymbol{y}_I$, $\boldsymbol{z}_I$), which point to the North, East, and downwards, respectively. $\mathcal{F}_b$ has its origin affixed to the aircraft center of gravity (CG), and has components of ($\boldsymbol{x}_b$, $\boldsymbol{y}_b$, $\boldsymbol{z}_b$), which point towards the aircraft nose, towards the right wingtip, and downwards, respectively. $\mathcal{F}_b$ can be related to $\mathcal{F}_I$ by a rotation through the Euler angles, ϕ, θ, and ψ, as shown in Fig. 2.1. These angles are known as the bank angle, pitch angle, and heading angle, respectively. The Euler angles are represented in vector form as $\boldsymbol{\Lambda} = [\phi,\ \theta,\ \psi]^T$.

From Newton's second law, we have

$$\begin{aligned} \boldsymbol{F}_I + \boldsymbol{W} &= m\frac{d(\boldsymbol{V}_I)}{dt}, \\ \boldsymbol{M}_I &= \frac{d\boldsymbol{H}}{dt}, \end{aligned} \tag{2.1}$$

where $\boldsymbol{F}_I = [F_{I_x}, F_{I_y}, F_{I_z}]^T$ is the set of external forces on the aircraft expressed in $\mathcal{F}_I$, $\boldsymbol{W} = [0, 0, mg]^T$ is the weight vector in $\mathcal{F}_I$, m is the aircraft mass, $g = 9.81$ m/s^2, and $m\boldsymbol{V}_I$ is the linear momentum vector, with $\boldsymbol{V}_I = [v_x, v_y, v_z]^T$ being the velocities of the aircraft, also expressed in the inertial reference frame. Likewise, $\boldsymbol{M}_I = [M_{I_l}, M_{I_m}, M_{I_n}]^T$ is the set of moments acting on the aircraft in $\mathcal{F}_I$ and $\boldsymbol{H} = [H_x, H_y, H_z]^T$ is the vector of angular momenta. The angular momentum can be defined as

$$\boldsymbol{H} = J\boldsymbol{\Omega},$$

where $\boldsymbol{\Omega} = [p, q, r]^T$ is the vector of aircraft angular velocities, and J is the inertia tensor, defined as

$$J = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}.$$

Given the symmetry resulting from the choice of the body fixed reference frame, the inertia tensor can be simplified by assuming

$$I_{xy} = I_{yx} = I_{yz} = I_{zy} = 0.$$

Additionally, an assumption can be made specific to the aircraft discussed in this work that the I_{xz} and I_{zx} terms of the inertia tensor are negligibly small. The inerta tensor can then be rewritten as the diagonal matrix

$$J = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}.$$

Since both $\boldsymbol{F}_I$ and $\boldsymbol{M}_I$ are defined in the inertial frame, their corresponding derivatives in (2.1) are also taken with respect to $\mathcal{F}_I$. Redefining the derivatives to be taken with respect to the body-fixed reference frame yields

$$\begin{aligned} \boldsymbol{F} &= m\frac{d\boldsymbol{V}}{dt}\bigg|_b + m(\boldsymbol{\Omega} \times \boldsymbol{V}), \\ \boldsymbol{M} &= \frac{d\boldsymbol{H}}{dt}\bigg|_b + \boldsymbol{\Omega} \times \boldsymbol{H}, \end{aligned}$$

where the aircraft velocities in $\mathcal{F}_b$ are defined as $\boldsymbol{V} = [u,\, v,\, w]^T$. Expanding these equations,

the aircraft equations of motion are

$$\begin{aligned} \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} &= m \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix}, \\ \begin{bmatrix} M_l \\ M_m \\ M_n \end{bmatrix} &= \begin{bmatrix} \dot{p}I_x + qr(I_z - I_y) \\ \dot{q}I_y + rp(I_x - I_z) \\ \dot{r}I_z + pq(I_y - I_x) \end{bmatrix}. \end{aligned} \tag{2.2}$$

Equation (2.2) can easily be solved for the angular acceleration terms as $\dot{\boldsymbol{\Omega}} = J^{-1}\boldsymbol{M} - J^{-1}(\boldsymbol{\Omega} \times \boldsymbol{H})$. Expanded, this is equivalent to

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} I_x^{-1}(M_l - (I_z - I_y)qr) \\ I_y^{-1}(M_m - (I_x - I_z)pr) \\ I_z^{-1}(M_n - (I_y - I_x)pq) \end{bmatrix}. \tag{2.3}$$

In order to solve for the linear accelerations, the weight vector must be expressed in the body-fixed reference frame. A map from $\mathcal{F}_I$ to $\mathcal{F}_b$ is obtained by a series of rotations through the Euler angles, namely:

1. $\mathcal{F}_I$ is rotated about $\boldsymbol{z}_I$ through the heading angle ψ to obtain $\mathcal{F}_1 = (\boldsymbol{x}_1, \boldsymbol{y}_1, \boldsymbol{z}_1)$.

2. $\mathcal{F}_1$ is rotated about $\boldsymbol{y}_1$ through the pitch angle θ to obtain $\mathcal{F}_2 = (\boldsymbol{x}_2, \boldsymbol{y}_2, \boldsymbol{z}_2)$.

3. $\mathcal{F}_2$ is rotated about $\boldsymbol{x}_2$ through the roll angle ϕ to obtain $\mathcal{F}_b = (\boldsymbol{x}_b, \boldsymbol{y}_b, \boldsymbol{z}_b)$.

Combining the three rotation matrices, the matrix $\boldsymbol{R}_{Ib}$ is found as

$$\boldsymbol{R}_{Ib} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}. \tag{2.4}$$

Using (2.4), the body axis components of $\boldsymbol{W}$ are given as

$$\boldsymbol{W}_b = \boldsymbol{R}_{Ib}\boldsymbol{W} = mg \begin{bmatrix} -\sin\theta \\ \sin\phi\cos\theta \\ \cos\phi\cos\theta \end{bmatrix}. \tag{2.5}$$

We define the aircraft gravitational acceleration vector as $\boldsymbol{G} = m^{-1}\boldsymbol{W}_b$. Combining (2.2) and (2.5), the linear body-axis accelerations are solved for as $\dot{\boldsymbol{V}} = m^{-1}\boldsymbol{F} - \boldsymbol{\Omega} \times \boldsymbol{V} + \boldsymbol{G}$, which expands as

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} F_x/m - g\sin\theta + rv - qw \\ F_y/m + g\cos\theta\sin\phi + pw - ru \\ F_z/m + g\cos\theta\cos\phi + qu - pv \end{bmatrix}. \tag{2.6}$$

Note that in (2.3) and (2.6), the force and moment definitions in (2.2) cannot be used as they depend directly on $\boldsymbol{V}$ and $\boldsymbol{\Omega}$. Instead, these forces and moments will be estimated from other states and measurements.

The Earth-fixed dynamics are obtained through a simple rotation of the body-fixed linear

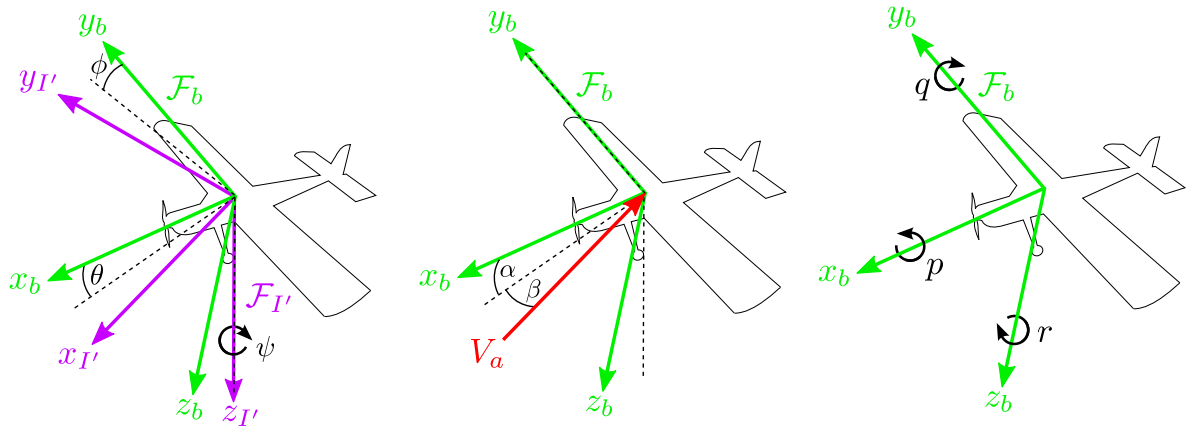

Figure 2.1: Euler angles, aerodynamic angles, and roll rates are shown with respect to the body-fixed reference frame.

---

velocities as

$$\dot{\boldsymbol{P}} = \boldsymbol{R}_{Ib}\boldsymbol{V}.$$

where $\boldsymbol{P} = [N,\ E,\ z_g]^T$ represents the North, East, and vertical CG position in $\mathcal{F}_I$.

Looking at Fig. 2.1, where $\mathcal{F}_{I'}$ is the inertial reference frame transposed to the aircraft CG, the aircraft's angular velocities can be related to its Euler angles as

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{R}_{b2}\begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{R}_{b1}\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \boldsymbol{R}_{bI}\begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix}. \tag{2.7}$$

Rewriting (2.7) as $\boldsymbol{\Omega} = \mathcal{E}(\boldsymbol{\Lambda})^{-1}\dot{\boldsymbol{\Lambda}}$, the Euler angle rates can easily be solved for as $\dot{\boldsymbol{\Lambda}} =$

$\mathcal{E}(\mathbf{\Lambda})\mathbf{\Omega}$. Expanding terms yields

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix}. \tag{2.8}$$

Given the vector of atmospheric disturbance velocities $\boldsymbol{V}_w = [u_w,\, v_w,\, w_w]^T$, the aircraft air-data measurements can be defined. These disturbances, which represent wind, turbulence, and other atmospheric effects, are defined in the body-fixed reference frame, as they are measured onboard the aircraft. The total airspeed, angle of attack, and angle of sideslip are

$$\begin{aligned} V_a &= \sqrt{(u-u_w)^2 + (v-v_w)^2 + (w-w_w)^2}, \\ \alpha &= \tan^{-1}\frac{w-w_w}{u-u_w}, \\ \beta &= \sin^{-1}\frac{v-v_w}{V_a}, \end{aligned} \tag{2.9}$$

as shown in Fig. 2.1. The linear velocity with respect to the wind is defined as $\bar{\boldsymbol{V}} = \boldsymbol{V} - \boldsymbol{V}_w$, and the total aircraft velocity as $V = \sqrt{\boldsymbol{V}^T\boldsymbol{V}}$. Finally, we denote the wind axes reference frame, shown in Fig. 2.1, as $\mathcal{F}_w$. The wind reference frame has components of $(\boldsymbol{x}_w,\, \boldsymbol{y}_w,\, \boldsymbol{z}_w)$ which are obtained using the following series of rotations:

1. $\mathcal{F}_b$ is rotated about $\boldsymbol{y}_b$ in the left-hand direction through the angle of attack α to obtain $\mathcal{F}_{w1} = (\boldsymbol{x}_{w1},\, \boldsymbol{y}_{w1},\, \boldsymbol{z}_{w1})$.

2. $\mathcal{F}_{w1}$ is rotated about $\boldsymbol{z}_{w1}$ through the side slip angle β to obtain $\mathcal{F}_w = (\boldsymbol{x}_w,\, \boldsymbol{y}_w,\, \boldsymbol{z}_w)$.

Equations 2.3 and 2.6 can easily be lumped together and rewritten as

$$\begin{bmatrix} \dot{\boldsymbol{\Omega}} \\ \dot{\boldsymbol{V}} \end{bmatrix} = \begin{bmatrix} J & 0 \\ 0 & mI_3 \end{bmatrix}^{-1} \left(\begin{bmatrix} \boldsymbol{M} \\ \boldsymbol{F} \end{bmatrix} - \begin{bmatrix} \boldsymbol{\Omega}^{\times} J & 0 \\ 0 & m\boldsymbol{\Omega}^{\times} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{bmatrix} \right) + \begin{bmatrix} 0 \\ \boldsymbol{G} \end{bmatrix}, \tag{2.10}$$

where

$$\boldsymbol{V}^{\times} = \begin{bmatrix} 0 & -w & v \\ w & 0 & -u \\ -v & u & 0 \end{bmatrix}, \quad \boldsymbol{\Omega}^{\times} = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}.$$

By relaxing the previously made assumption that $\mathcal{F}_b$ is centered at the aircraft CG, the aircraft linear and rotational velocities become coupled [41]. Defining the vector from the origin of $\mathcal{F}_b$ to the CG as $\boldsymbol{\delta}_{cg} = [\delta_x, \delta_y, \delta_z]^T$, the coupled velocities are written as

$$\begin{bmatrix} \dot{\boldsymbol{\Omega}} \\ \dot{\boldsymbol{V}} \end{bmatrix} = \begin{bmatrix} J & \boldsymbol{\delta}_{cg}^{\times} \\ -m\boldsymbol{\delta}_{cg}^{\times} & mI_3 \end{bmatrix}^{-1} \left(\begin{bmatrix} \boldsymbol{M} \\ \boldsymbol{F} + m\boldsymbol{G} \end{bmatrix} - \begin{bmatrix} J\boldsymbol{\Omega}^{\times} - m\boldsymbol{V}^{\times}\boldsymbol{\delta}_{cg}^{\times} & m\boldsymbol{\Omega}^{\times}\boldsymbol{\delta}_{cg}^{\times} \\ -m\boldsymbol{\Omega}^{\times}\boldsymbol{\delta}_{cg}^{\times} & m\boldsymbol{\Omega}^{\times} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{bmatrix} \right). \tag{2.11}$$

Once again, the rotational and translational kinematic equations are written as

$$\begin{aligned} \dot{\boldsymbol{\Lambda}} &= \mathcal{E}(\phi, \theta)\boldsymbol{\Omega}, \\ \dot{\boldsymbol{P}} &= \mathcal{R}_{Ib}(\boldsymbol{\Lambda})\boldsymbol{V}. \end{aligned} \tag{2.12}$$

Changes in the initial heading direction (Ψ_0) can have a large effect on the linearization of the equations of motion. To circumvent this, the nominal CG position will be redefined as

$\boldsymbol{P}_0 = [X,\, Y,\, h]^T$ where $X = N\cos\psi_0 + E\sin\psi_0$, $Y = -N\sin\psi_0 + E\cos\psi_0$, and $h = -z_g$. The resulting differential equations are given as

$$\dot{\boldsymbol{P}}_0 = \mathcal{R}_{Ib}(\boldsymbol{\Lambda}_0)\boldsymbol{V}I_0, \tag{2.13}$$

where $\boldsymbol{\Lambda}_0 = [\phi,\, \theta,\, \psi - \psi_0]^T$ and $I_0 = \text{diag}(1,\, 1\, -1)$.

2.3 Simulation Environment

All controllers are tested in a rigorous `MATLAB`-based simulation environment. The simulation environment involves the aircraft nonlinear flight dynamic model, along with actuator models, and features steady winds, moderate turbulence from the low altitude Dryden turbulence model (see Section 2.3.1), sensor noise, controller delays, and aerodynamic uncertainties.

Sensor noise is sampled from a Gaussian distribution with standard deviations of 0.5 rad/s for p, q, and r measurements, 2 m/s for V_a, 0.01 rad for ϕ, θ, and ψ, and 2 m for X, Y, and h, as determined from sensor specifications. The controller operates in discrete-time at a frequency of 25 Hz. Controller commands are implemented in the simulation with a delay of less than one timestep (0.04 seconds) or less. Finally, aerodynamic uncertainties are implemented as bounded additive perturbations on the aerodynamic coefficients. The aerodynamic uncertainty bounds can be found in Chapter 5.

2.3.1 Dryden Turbulence Model

The Dryden Wind Turbulence model (MIL-F-8785C) is used to generate simulated wind gusts [42]. The turbulence velocity components are calculated in the body frame with the following continuous-time transfer functions:

$$\begin{aligned} H_u(s) &= \sigma_u \sqrt{\frac{2L_u}{\pi V_a}} \cdot \frac{1}{1 + \frac{L_u}{V_a} s}, \\ H_v(s) &= \sigma_v \sqrt{\frac{L_v}{\pi V_a}} \cdot \frac{1}{\left(1 + \frac{L_v}{V_a} s\right)^2}, \\ H_w(s) &= \sigma_w \sqrt{\frac{L_w}{\pi V_a}} \cdot \frac{1}{\left(1 + \frac{L_w}{V_a} s\right)^2}, \end{aligned} \tag{2.14}$$

where V_a is the current airspeed, $L_{(.)}$ are disturbance scale factors, $\sigma_{(.)}$ are the root-mean-square disturbance intensities, and the subscripts u, v, and w refer to the aircraft body axis directions. As the Telemaster flight operations are restricted to under 400 ft above ground level (AGL), the low altitude Dryden model (valid for altitudes under 1000 ft) is utilized [43]. For low altitude, the scaling factors and intensities are given as $L_{u,v} = {}^{H}/_{h}$, $L_w = H$, $\sigma_{u,v} = {}^{u_{20}}/_{10h^{1/3}}$, and $\sigma_w = {}^{u_{20}}/_{10}$, where H is the current vehicle altitude in ft AGL, u_{20} is the average wind speed at 20 ft above ground level in kts, and $h = (0.177 + 0.000823H)^{1.2}$ ft. Values of 15 kts, 30 kts, and 45 kts are used for u_{20} to represent light, moderate, and severe turbulence, respectively. Note that the scale factors and intensities in the u and v directions are identical. The low altitude Dryden model transfer functions can now be rewritten as

$$
\begin{aligned}
H_u(s) &= \frac{u_{20}}{10h^{1/3}}\sqrt{\frac{2H}{\pi V_a h}}\cdot\frac{1}{1+\frac{H}{V_a h}s}, \\
H_v(s) &= \frac{u_{20}}{10h^{1/3}}\sqrt{\frac{H}{\pi V_a h}}\cdot\frac{1}{\left(1+\frac{H}{V_a h}s\right)^2}, \\
H_w(s) &= \frac{u_{20}}{10}\sqrt{\frac{H}{\pi V_a}}\cdot\frac{1}{\left(1+\frac{H}{V_a}s\right)^2}.
\end{aligned}
\tag{2.15}
$$

2.4 Telemaster UAS Platform

A commercially available 6 foot wingspan Telemaster radio-controlled (R/C) plane (Hobby Express) is used for modeling, simulation, and flight testing. As it is a "trainer-class" R/C plane, the Telemaster, shown in Fig. 2.2, is inherently stable. Despite its classification, the Telemaster's control surface sizing is such that the aircraft is quite agile, and can easily perform aerobatic maneuvers [44]. The airframe also exhibits a spacious fuselage, allowing a custom autopilot and several sensors to be installed. The aircraft's lifting tail configuration is designed so that the CG of the aircraft is at approximately the center chord, further back than the usual quarter-chord placement for similar platforms. This allows more of the fuselage space to be used for housing electronics and batteries without requiring additional nose ballast in order to maintain a proper CG placement. While the airframe is mostly stock, it should be noted that minor modifications have been made for the installation of sensors in both the fuselage and wings. The geometric properties of the aircraft can be found in Table 2.1. The moments of inertia displayed in Table 2.1 were determined by bifilar

Table 2.1: Telemaster Parameters

Mass (m)	3.307	kg
I_x	0.198	kg-m^2
I_y	0.305	kg-m^2
I_z	0.418	kg-m^2
Wing area (S)	0.56	m^2
Wing span (b)	1.83	m
Wing MAC* ($\bar{c}$)	0.30	m

*mean aerodynamic chord

and compound pendulum tests performed in the manner described in [44]. Previous work utilizing this aircraft can be found in [44]-[47].

Figure 2.2: Telemaster UAS. (photo by Mark Palframan)

2.4.1 Thrust Model

Assuming that the electronic speed controller provides a constant propeller RPM for a given input command δ_T, an experimentally obtained lookup table is used to map δ_T to a corresponding RPM value [44]. A Javaprop based lookup table then maps the current RPM and airspeed to the propeller thrust [48]. The thrust, T, is applied along the $\boldsymbol{x}_b$ axis. Furthermore, propeller effects, including reaction torque, P-factor, propwash, and gyroscopic precession are not modeled. Finally, it is assumed that the dynamics of the propulsion system are much faster than the aircraft dynamics, and as such no additional dynamics are included in the model. Further details on the propulsion model can be found in [44].

2.4.2 Aerodynamic Model

The aerodynamic forces and moments in (2.11) are defined in terms of aerodynamic coefficients, namely,

$$\begin{aligned} F_i(\cdot) &= \frac{1}{2} C_i(\cdot) \rho V_a^2 S, \text{ for } i = x, y, z, \\ M_j(\cdot) &= \frac{1}{2} C_j(\cdot) \rho V_a^2 S b, \text{ for } j = l, n, \\ M_m(\cdot) &= \frac{1}{2} C_m(\cdot) \rho V_a^2 S \bar{c}, \end{aligned} \tag{2.16}$$

where $C_{(\cdot)}$ denotes an aerodynamic coefficient and $\rho = 1.3302$ kg/m^2 is the air density. Common maximum likelihood system identification techniques, such as the output error and equation error methods, solve for the aerodynamic parameter values that make up the

nonlinear aerodynamic coefficients (based on the chosen aerodynamic model structure) by comparing measured force and moment time-histories to a postulated aerodynamic model [49]-[52]. The aerodynamic model structure is assumed to be

$$
\begin{aligned}
C_x &= C_{x_0} + C_{x_\alpha}\alpha + C_{x_{\delta_T}}\delta_T + C_{x_T} 2T/(\rho S V_a^2), \\
C_y &= C_{y_0} + C_{y_\beta}\beta + C_{y_{\delta_A}}\delta_A + C_{y_{\delta_R}}\delta_R + (C_{y_p}p + C_{y_r}r)b/(2V_a), \\
C_z &= C_{z_0} + C_{z_\alpha}\alpha + C_{z_{\delta_E}}\delta_E + C_{z_q}q\bar{c}/(2V_a) + C_{z_T} 2T/(\rho S V_a^2), \\
C_l &= C_{l_0} + C_{l_\beta}\beta + C_{l_{\delta_A}}\delta_A + C_{l_{\delta_R}}\delta_R + (C_{l_p}p + C_{l_r}r)b/(2V_a), \\
C_m &= C_{m_0} + C_{m_\alpha}\alpha + C_{m_{\delta_E}}\delta_E + C_{m_q}q\bar{c}/(2V_a), \\
C_n &= C_{n_0} + C_{n_\beta}\beta + C_{n_{\delta_A}}\delta_A + C_{n_{\delta_R}}\delta_R + (C_{n_p}p + C_{n_r}r)b/(2V_a),
\end{aligned}
$$

with values found in Table 2.2. Here δ_E, δ_A, and δ_R are the the elevator, aileron, and rudder deflections, respectively.

Three identical servomotors mapping the commands δ_{E_c}, δ_{A_c}, and δ_{R_c} to the control surface deflections δ_E, δ_A, and δ_R are nominally modeled as

$$G_{act}(s) = \omega_{ns}^2/(s^2 + 2\zeta_s\omega_{ns}s + \omega_{ns}^2). \tag{2.17}$$

The natural frequency and damping ratio were experimentally estimated to be $\omega_{ns} = 13.7$ rad/s and $\zeta_s = 0.67$ by measuring the servomotor frequency response in [53].

Table 2.2: Aerodynamic Parameter Values

Term	Value	Term	Value	Term	Value	Term	Value	Term	Value	Term	Value
$\mathcal{C}_{x_0}$	-0.3066	$\mathcal{C}_{y_0}$	0.0254	$\mathcal{C}_{z_0}$	-0.2103	$\mathcal{C}_{l_0}$	-0.0002	$\mathcal{C}_{m_0}$	-0.0116	$\mathcal{C}_{n_0}$	-0.0026
$\mathcal{C}_{x_\alpha}$	1.7086	$\mathcal{C}_{y_\beta}$	-0.3763	$\mathcal{C}_{z_\alpha}$	-2.3665	$\mathcal{C}_{l_\beta}$	-0.0585	$\mathcal{C}_{m_\alpha}$	-0.3529	$\mathcal{C}_{n_\beta}$	0.0258
$\mathcal{C}_{x_{\delta_T}}$	0.3005	$\mathcal{C}_{y_{\delta_A}}$	-0.1350	$\mathcal{C}_{z_{\delta_E}}$	0.6548	$\mathcal{C}_{l_{\delta_A}}$	0.1123	$\mathcal{C}_{m_{\delta_E}}$	0.5687	$\mathcal{C}_{n_{\delta_A}}$	-0.0049
$\mathcal{C}_{x_T}$	-0.2431	$\mathcal{C}_{y_{\delta_R}}$	0.1038	$\mathcal{C}_{z_T}$	-0.2628	$\mathcal{C}_{l_{\delta_R}}$	0.0052			$\mathcal{C}_{n_{\delta_R}}$	-0.0390
		$\mathcal{C}_{yp}$	0.5582	$\mathcal{C}_{zq}$	-52.6239	$\mathcal{C}_{lp}$	-0.2810	$\mathcal{C}_{mq}$	-14.262	$\mathcal{C}_{np}$	-0.0737
		$\mathcal{C}_{yr}$	0.1007			$\mathcal{C}_{lr}$	0.1663			$\mathcal{C}_{nr}$	-0.0898

2.5 $\mathcal{H}_\infty$ Control

This section presents control synthesis procedures for linear time invariant (LTI) and linear parameter varying (LPV) $\mathcal{H}_\infty$ control. The presented synthesis methods are based on [54]-[56]. Notation in this section is borrowed from [57].

2.5.1 LTI $\mathcal{H}_\infty$ Controller Synthesis

The discrete-time LTI system G with zero initial conditions is given by

$$\begin{bmatrix} \bar{\boldsymbol{x}}(k+1) \\ \bar{\boldsymbol{z}}(k) \\ \bar{\boldsymbol{y}}(k) \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & 0 \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}(k) \\ \boldsymbol{w}(k) \\ \bar{\boldsymbol{u}}(k) \end{bmatrix}, \ \bar{\boldsymbol{x}}(0) = 0, \tag{2.18}$$

where the signal $\bar{\boldsymbol{x}}(k) \in \mathbb{R}^n$ is the error between the actual and trim values of the state vector, namely $\bar{\boldsymbol{x}}(k) = \boldsymbol{x}(k) - \boldsymbol{x}_{tr}$, and $k \in \mathbb{Z}_+$ denotes the discrete time instant. Similarly, $\bar{\boldsymbol{y}}(k) = \boldsymbol{y}(k) - \boldsymbol{y}_{tr} \in \mathbb{R}^{ny}$ and $\bar{\boldsymbol{u}}(k) = \boldsymbol{u}(k) - \boldsymbol{u}_{tr} \in \mathbb{R}^{nu}$, where $\bar{\boldsymbol{y}}(k)$ and $\bar{\boldsymbol{u}}(k)$ are the errors between the measurement output and control input vectors at time instants k. The signals

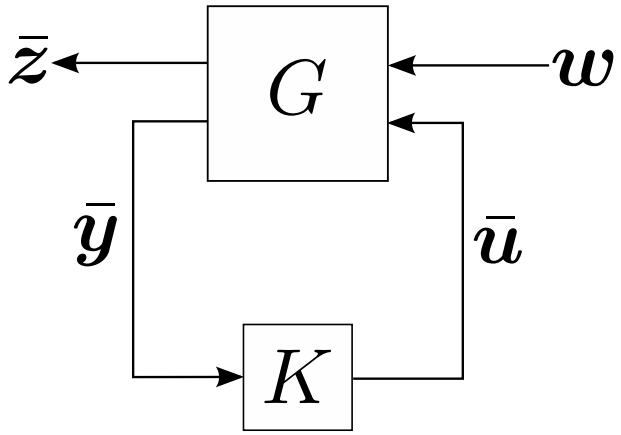

Figure 2.3: The closed-loop block diagram.

$\boldsymbol{w}(k) \in \mathbb{R}^{nw}$ and $\bar{\boldsymbol{z}}(k) \in \mathbb{R}^{nz}$ denote the exogenous disturbances and performance errors, respectively. The disturbance channel is partitioned as $\boldsymbol{w}(k) = [\boldsymbol{w}_w(k)^T,\ \boldsymbol{w}_n(k)^T]^T$, where $\boldsymbol{w}_w(k)$ represents atmospheric disturbances and $\boldsymbol{w}_n(k)$ sensor noise. The atmospheric disturbances are a summation of steady winds and gusts generated from the low altitude Dryden turbulence model, presented in Section 2.3.1. Additionally, it is assumed that disturbances are finite energy signals satisfying $\boldsymbol{w} \in \ell_2$. The control inputs $\bar{\boldsymbol{u}}(k)$ for the plant G are defined by an LTI controller, K, with the state-space representation

$$\begin{bmatrix} \boldsymbol{x}_K(k+1) \\ \bar{\boldsymbol{u}}(k) \end{bmatrix} = \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_K(k) \\ \bar{\boldsymbol{y}}(k) \end{bmatrix}, \ \boldsymbol{x}_K(0) = 0, \tag{2.19}$$

where $\boldsymbol{x}_K(k) \in \mathbb{R}^{n_K}$ is the controller state vector with zero initial conditions. Fig. 2.3 shows the feedback interconnection of G and K from (2.18) and (2.19).

Denoting the closed loop system as M and concatenating the plant and controller state

vectors as $\bar{\boldsymbol{x}}_M(k) = \begin{bmatrix} \bar{\boldsymbol{x}}(k)^T & \boldsymbol{x}_K(k)^T \end{bmatrix}^T \in \mathbb{R}^{n+n_K}$, the closed-loop system equations are

$$\begin{bmatrix} \bar{\boldsymbol{x}}_M(k+1) \\ \bar{\boldsymbol{z}}(k) \end{bmatrix} = \begin{bmatrix} A_M & B_M \\ C_M & D_M \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}_M(k) \\ \boldsymbol{w}(k) \end{bmatrix}, \; \bar{\boldsymbol{x}}_M(0) = 0, \tag{2.20}$$

and the state-space matrices A_M, B_M, C_M, and D_M are defined as

$$\left[\begin{array}{c|c} A_M & B_M \\ \hline C_M & D_M \end{array}\right] = \left[\begin{array}{cc|c} A + B_2 D_K C_2 & B_2 C_K & B_1 + B_2 D_K D_{21} \\ B_K C_2 & A_K & B_K D_{21} \\ \hline C_1 + D_{12} D_K C_2 & D_{12} C_K & D_{11} + D_{12} D_K D_{21} \end{array}\right]. \tag{2.21}$$

In this work, an admissible controller is defined in the following way:

Definition 1 (γ-admissible synthesis [57])**.** *A controller K is a γ-admissible synthesis for the plant G if the closed-loop system in Fig. 2.3 is exponentially stable and the performance inequality $\|\boldsymbol{w} \mapsto \bar{\boldsymbol{z}}\|_{\ell_2 \to \ell_2} < \gamma$ is achieved.*

The ℓ_2-gain of the input-output map is further defined as

$$\|\boldsymbol{w} \mapsto \bar{\boldsymbol{z}}\|_{\ell_2 \to \ell_2} = \sup_{\|\boldsymbol{w}\|_{\ell_2} \neq 0} \frac{\|\bar{\boldsymbol{z}}\|_{\ell_2}}{\|\boldsymbol{w}\|_{\ell_2}}.$$

Defining the following matrices:

$$\begin{aligned} \mathrm{Im}\left[\mathrm{V}_1^{\mathrm{T}}\ \mathrm{V}_2^{\mathrm{T}}\right]^{\mathrm{T}} &= \mathrm{Ker}\left[\mathrm{B}_2^{\mathrm{T}}\ \mathrm{D}_{12}^{\mathrm{T}}\right], & \left[V_1^T\ V_2^T\right]\left[V_1^T\ V_2^T\right]^T &= I, \\ \mathrm{Im}\left[\mathrm{U}_1^{\mathrm{T}}\ \mathrm{U}_2^{\mathrm{T}}\right]^{\mathrm{T}} &= \mathrm{Ker}\left[\mathrm{C}_2\ \mathrm{D}_{21}\right], & \left[U_1^T\ U_2^T\right]\left[U_1^T\ U_2^T\right]^T &= I, \end{aligned}$$

the LTI $\mathcal{H}_\infty$ controller synthesis conditions are

$$\begin{gathered} F^T R F - V_1^T R V_1 + N^T N - \gamma^2 V_2^T V_2 \prec 0, \\ \begin{bmatrix} W^T S W - U_1^T S U_1 - U_2^T U_2 & L^T \\ L & -\gamma^2 I_{n_z} \end{bmatrix} \prec 0, \\ \begin{bmatrix} R & I \\ I & S \end{bmatrix} \succeq 0, \end{gathered} \tag{2.22}$$

where

$$F = A^T V_1 + C_1^T V_2, \quad N = B_1^T V_1 + D_{11}^T V_2, \quad W = A U_1 B_1 U_2, \quad L = C_1 U_1 + D_{11} U_2.$$

To achieve optimal performance, these synthesis conditions are solved for R, S, and γ in the form of a semidefinite program (SDP), namely:

$$\text{minimize: } \gamma^2$$

$$\text{subject to: (2.22).}$$

In this work, the optimal value of γ is typically relaxed, and the synthesis conditions resolved in order to improve overall robustness. Given R, S, and γ from (2.22), one way of obtaining the γ-admissable synthesis is given next. Assuming that the coupling condition in (2.22) holds with strict matrix inequality, and so the controller will have the same state dimension as the plant, we construct a matrix X and its inverse from the synthesis solutions R and S in the following way:

$$X = \begin{bmatrix} S & -SM \\ -M^T S & I + M^T SM \end{bmatrix}, \quad X^{-1} = \begin{bmatrix} R & M \\ M^T & I \end{bmatrix}, \text{ where } M = (R - S^{-1})^{1/2}.$$

Defining the following matrices:

$$\mathcal{A} = \begin{bmatrix} A & 0 \\ 0 & 0_n \end{bmatrix}, \qquad \mathcal{B} = \begin{bmatrix} B_1 & 0 \\ 0 & 0_{n\times n_d} \end{bmatrix},$$
$$\mathcal{C}_1 = \begin{bmatrix} C_1^T & 0 \\ 0 & 0_{n\times n_z} \end{bmatrix}, \quad \mathcal{C}_2 = \begin{bmatrix} 0 & 0_{n_d\times n} \\ C_1 & 0 \end{bmatrix}, \quad \mathcal{D} = \begin{bmatrix} -I\gamma & D_{11}^T \\ D_{11} & -I\gamma \end{bmatrix},$$

the controller, K, can be found by solving the linear matrix inequality (LMI) problem

$$H + P^T JQ + Q^T J^T P \prec 0, \tag{2.23}$$

for J, where H, P, and Q are constructed as

$$H = \begin{bmatrix} -X^{-1} & \mathcal{A} & \mathcal{B} \\ \mathcal{A}^T & -X & \frac{1}{\gamma}\mathcal{C}_1 \\ \mathcal{B}^T & \frac{1}{\gamma}\mathcal{C}_2 & \frac{1}{\gamma}\mathcal{D} \end{bmatrix}, \quad P = \begin{bmatrix} 0 & I & 0_{n\times 2n+n_w} & 0 \\ B_2^T & 0 & 0 & \frac{1}{\gamma}D_{12}^T \end{bmatrix}, \quad Q = \begin{bmatrix} 0_{n\times 2n} & 0 & I & 0 & 0_{n\times n_z} \\ 0 & C_2 & 0 & D_{21} & 0 \end{bmatrix}.$$

The controller matrices in (2.19) can then be easily solved for as

$$J = \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix}.$$

2.5.2 LPV $\mathcal{H}_\infty$ Controller Synthesis

The LPV controller synthesis follows a similar procedure to the LTI case. The synthesis technique is based on an LPV system in a linear fractional transformation (LFT) form, shown in Fig. 2.4, allowing rational functions of the parameter vector $\boldsymbol{p}(k) \in \mathbb{R}^{n_p}$ to be represented. Here, $\boldsymbol{\vartheta} = \Delta(\boldsymbol{p})\boldsymbol{\varphi}$, where $\Delta(\boldsymbol{p})$ is a block diagonal matrix of parameters. The parameter-independent Lyapunov synthesis approach used is fully described in [58], a generalization of the process developed by Packard in [56].

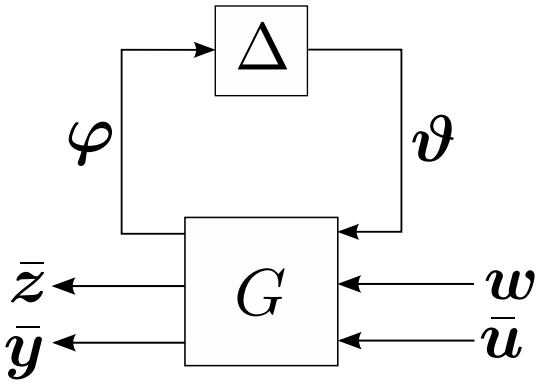

Figure 2.4: The parameter varying open loop system.

The discrete-time LPV system plant is given by

$$\begin{bmatrix} \bar{\boldsymbol{x}}(k+1) \\ \bar{\boldsymbol{z}}(k) \\ \bar{\boldsymbol{y}}(k) \end{bmatrix} = \begin{bmatrix} A(\boldsymbol{p}(k)) & B_1(\boldsymbol{p}(k)) & B_2(\boldsymbol{p}(k)) \\ C_1(\boldsymbol{p}(k)) & D_{11}(\boldsymbol{p}(k)) & D_{12}(\boldsymbol{p}(k)) \\ C_2(\boldsymbol{p}(k)) & D_{21}(\boldsymbol{p}(k)) & 0 \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}(k) \\ \boldsymbol{w}(k) \\ \bar{\boldsymbol{u}}(k) \end{bmatrix}, \ \bar{\boldsymbol{x}}(0) = 0, \tag{2.24}$$

where all system matrices may have rational dependence on the parameters in $\boldsymbol{p}$. Performing a linear fractional transformation on the parameters yields the linear fractional representation (LFR) of the LPV system:

$$\begin{bmatrix} \bar{\boldsymbol{x}}(k+1) \\ \boldsymbol{\varphi}(k) \\ \bar{\boldsymbol{z}}(k) \\ \bar{\boldsymbol{y}}(k) \end{bmatrix} = \begin{bmatrix} A_{ss} & A_{sp} & B_{1s} & B_{2s} \\ A_{ps} & A_{pp} & B_{1p} & B_{2p} \\ C_{1s} & C_{1p} & D_{11} & D_{12} \\ C_{2s} & C_{2p} & D_{21} & 0 \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}(k) \\ \boldsymbol{\vartheta}(k) \\ \boldsymbol{d}(k) \\ \bar{\boldsymbol{u}}(k) \end{bmatrix}, \tag{2.25}$$

$$\boldsymbol{\vartheta}(k) = \Delta(\boldsymbol{p}(k))\boldsymbol{\varphi}(k).$$

The LPV synthesis conditions are

$$F_s^T R_s F_s + F_p^T R_p F_p - V_{1s}^T R_s V_{1s} - V_{1p}^T R_p V_{1p} + M^T M - \gamma^2 V_2^T V_2 \prec 0,$$
$$\begin{bmatrix} \left\{ \begin{array}{c} W_s^T S_s W_s + W_p^T S_p W_p \dots \\ -U_{1s}^T S_s U_{1s} - U_{1p}^T S_p U_{1p} \dots \\ -U_2^T U_2 \end{array} \right\} & L^T \\ L & -\gamma^2 I \end{bmatrix} \prec 0, \tag{2.26}$$
$$\begin{bmatrix} R_s & I \\ I & S_s \end{bmatrix} \succeq 0, \quad \begin{bmatrix} R_p & I \\ I & S_p \end{bmatrix} \succeq 0,$$

where

$$F_s = A_{ss}^T V_{1s} + A_{ps}^T V_{1p} + C_{1s}^T V_2, \qquad F_p = A_{sp}^T V_{1s} + A_{pp}^T V_{1p} + C_{1s}^T V_2,$$
$$W_s = A_{ss} U_{1s} + A_{sp} U_{1p} + B_{1s} U_2, \qquad W_p = A_{ps} U_{1s} + A_{pp} U_{1p} + B_{1p} U_2,$$
$$M = B_{1s}^T V_{1s} + B_{1p}^T V_{1p} + D_{11}^T V_2, \qquad L = C_{1s} U_{1s} + C_{1p} U_{1p} + D_{11} U_2,$$
$$\text{Im} \begin{bmatrix} V_{1s}^T & V_{1p}^T & V_2^T \end{bmatrix}^{\text{T}} = \text{Ker} \begin{bmatrix} B_{2s}^T & B_{2p}^T & D_{12}^T \end{bmatrix}, \quad \begin{bmatrix} V_{1s}^T & V_{1p}^T & V_2^T \end{bmatrix} \begin{bmatrix} V_{1s}^T & V_{1p}^T & V_2^T \end{bmatrix}^T = I,$$
$$\text{Im} \begin{bmatrix} U_{1s}^T & U_{1p}^T & U_2^T \end{bmatrix}^{\text{T}} = \text{Ker} \begin{bmatrix} C_{2s} & C_{2p} & D_{21} \end{bmatrix}, \quad \begin{bmatrix} U_{1s}^T & U_{1p}^T & U_2^T \end{bmatrix} \begin{bmatrix} U_{1s}^T & U_{1p}^T & U_2^T \end{bmatrix}^T = I.$$

The SDP for the synthesis procedure is summarized as

$$\text{minimize: } \gamma^2$$

$$\text{subject to: (2.26).}$$

Based on the solutions for γ, R_s, S_s, R_p, and S_p, we can obtain a γ-admissible LPV controller in LFT form. One method to do so is provided next. The controller dimensions are dependent on the rank of R_s, S_s, R_p, and S_p. Assuming that the coupling conditions in (2.26) hold with strict matrix inequalities, the constructed controllers will have the same state dimensions as the plant [59]. We first construct the following matrix blocks:

$$M_s = (R_s - S_s^{-1})^{1/2},\ M_p S = (R_p - S_p^{-1})^{1/2},$$

$$X_{11} = \text{diag}\left(S_s, S_p\right),\ X_{12} = \text{diag}\left(S_s M_s, S_p M_p\right),$$

$$X_{22} = \text{diag}\left(I + M_s^T S_s M_s, I + M_p^T S_p M_p\right),$$

$$Y_{11} = \text{diag}\left(R_s, R_p\right),\ Y_{12} = \text{diag}\left(M_s, M_p\right).$$

Defining the following matrices:

$$\mathcal{A} = \begin{bmatrix} A_{ss} & A_{sp} & 0 \\ A_{ps} & A_{pp} & 0 \\ 0 & 0 & 0_{n+n_p} \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} B_{1s} & 0 \\ B_{1p} & 0 \\ 0 & 0_{n+n_p} \end{bmatrix}, \quad \mathcal{C} = \begin{bmatrix} 0 & 0 & 0_{n_d \times (n+n_p)} \\ C_{1s} & C_{1p} & 0 \end{bmatrix},$$

$$\mathcal{D} = \begin{bmatrix} -I\gamma & D_{11}^T \\ D_{11} & -I\gamma \end{bmatrix}, \quad X = \begin{bmatrix} X_{11} & -X_{12} \\ -X_{12}^T & X_{22} \end{bmatrix}, \quad Y = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{12}^T & I \end{bmatrix},$$

the controller is found by solving the LMI

$$H + P^T J Q + Q^T J^T P \prec 0, \tag{2.27}$$

where

$$P = \begin{bmatrix} 0 & 0 & I_{n+n_p} & 0_{(n+n_p)\times(2n+2n_p+n_d)} & 0 \\ B_{2s}^T & B_{2p}^T & 0 & 0 & D_{12}^T/\gamma \end{bmatrix},$$

$$Q = \begin{bmatrix} 0_{(n+n_p)\times(2n+2n_p)} & 0 & 0 & I & 0 & 0 \\ 0 & C_{2s} & C_{2p} & 0 & D_{21} & 0_{n_y \times n_z} \end{bmatrix},$$

$$H = \begin{bmatrix} -Y & \mathcal{A} & \mathcal{B} \\ \mathcal{A}^T & -X & \mathcal{C}^T/\gamma \\ \mathcal{B}^T & \mathcal{C}/\gamma & \mathcal{D}/\gamma \end{bmatrix}, \quad J = \begin{bmatrix} A_{ss}^K & A_{sp}^K & B_s^K \\ A_{ps}^K & A_{pp}^K & B_p^K \\ C_s^K & C_p^K & D^K \end{bmatrix}.$$

The resulting controller is defined by the state-space equations

$$\begin{bmatrix} \boldsymbol{x}^K(k+1) \\ \boldsymbol{\varphi}^K(k) \\ \bar{\boldsymbol{u}}(k) \end{bmatrix} = \begin{bmatrix} A_{ss}^K & A_{sp}^K & B_s^K \\ A_{ps}^K & A_{pp}^K & B_p^K \\ C_s^K & C_p^K & D^K \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^K(k) \\ \boldsymbol{\vartheta}^K(k) \\ \bar{\boldsymbol{y}}(k) \end{bmatrix}, \tag{2.28}$$

where $\boldsymbol{x}^K(0) = 0$ and $\boldsymbol{\vartheta}^K(k) = \Delta^K(\boldsymbol{p}(k))\boldsymbol{\varphi}(k)$.

The feedback interconnection of the LFR plant (2.24) and the LFR controller (2.28) is shown in Fig. 2.5. The closed-loop controller is given by the state-space equations

$$\begin{bmatrix} \boldsymbol{x}^K(k+1) \\ \bar{\boldsymbol{u}}(k) \end{bmatrix} = \begin{bmatrix} A_{cl}^K(\boldsymbol{p}(k)) & B_{cl}^K(\boldsymbol{p}(k)) \\ C_{cl}^K(\boldsymbol{p}(k)) & D_{cl}^K(\boldsymbol{p}(k)) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^K(k) \\ \bar{\boldsymbol{y}}(k) \end{bmatrix},$$

where

$$\begin{aligned} A_{cl}^K(\boldsymbol{p}) &= A_{ss}^K + A_{sp}^K\boldsymbol{p}(I - A_{pp}^K\boldsymbol{p})^{-1}A_{ps}^K, \\ B_{cl}^K(\boldsymbol{p}) &= B_s^K + A_{sp}^K\boldsymbol{p}(I - A_{pp}^K\boldsymbol{p})^{-1}B_p^K, \\ C_{cl}^K(\boldsymbol{p}) &= C_s^K + C_p^K\boldsymbol{p}(I - A_{pp}^K\boldsymbol{p})^{-1}A_{ps}^K, \\ D_{cl}^K(\boldsymbol{p}) &= D^K + C_p^K\boldsymbol{p}(I - A_{pp}^K\boldsymbol{p})^{-1}B_p^K. \end{aligned}$$

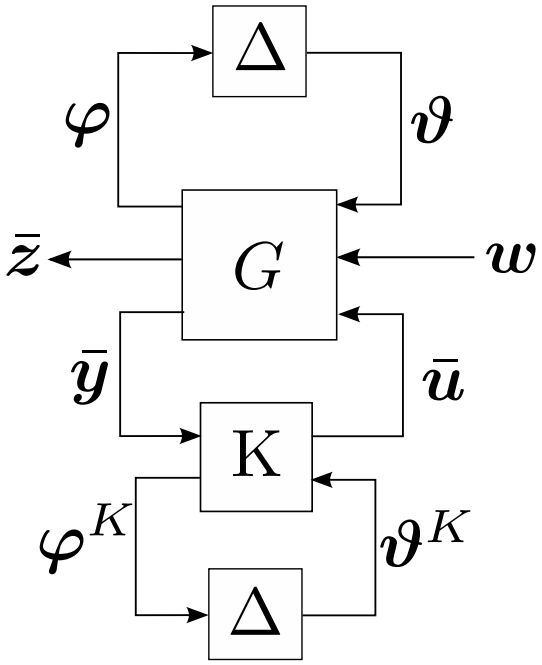

Figure 2.5: The feedback interconnection of the LFR controller.

2.5.3 Computations

SDPs in the controller synthesis process are solved in `MATLAB` 2014a using the `YALMIP` toolbox with `SDPT3` as the chosen solver [30], [60]. All computations are carried out on a Dell Precision T3500 Desktop running 64-bit Windows 7, with an Intel Xeon W3550 Quad Core processor and 6 GB of RAM.

Chapter 3

An LPV Path-Following Controller

Much of the background, definitions, and formulation of the path-following problem is borrowed from [12], with some modifications made for simplification or preference. This chapter is an extended version of [1], which is in turn based on the work found in [47]. This chapter provides a more detailed explanation of the synthesis procedure than that provided in [1] in addition to extensions to paths with a tighter radius of curvature.

The outline of this chapter is as follows. Section 3.1 presents the equations of motion of the parallel transport frame and the nonlinear backstepping controller based on these equations, as developed by [12]. Section 3.2 presents the controller synthesis procedure for three $\mathcal{H}_\infty$ controllers: a rate-tracking inner-loop controller, and LTI and LPV controllers based on the lumped UAS and path-following equations of motion. Parameter-varying trim points, linearization, discretization, and performance outputs for synthesis are all provided.

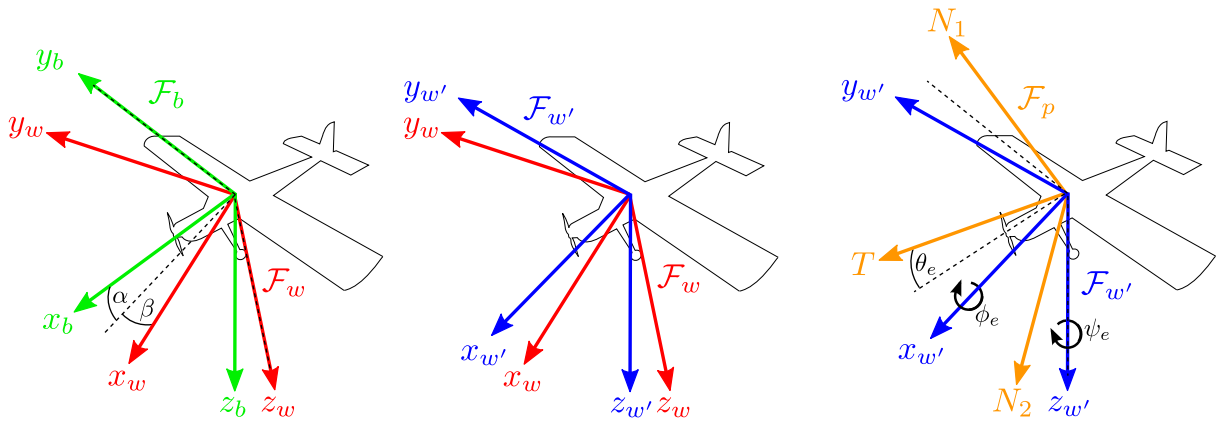

Figure 3.1: The body-fixed reference frame, $\mathcal{F}_b$, wind reference frame, $\mathcal{F}_w$, and parallel transport frame, $\mathcal{F}_p$, for an aircraft system.

Section 3.3 presents the simulation environment, paths of interest, and utilized performance metrics for evaluating path-following performance. Finally, Section 3.4 provides a summary of simulation results for the controllers presented in Section 3.2 applied to the paths in Section 3.3.

3.1 Path-Following Dynamic Equations

The path-following dynamics are based on a virtual vehicle moving along a path at some prescribed rate. At every point on the path, the virtual vehicle has an associated reference frame. Let $\boldsymbol{p}(\ell)$ represent the path to be followed in $\mathcal{F}_I$, parameterized by the path length ℓ. At each point on the path, a parallel transport frame (sometimes referred to as a rotation minimizing frame) [61], [62], denoted $\mathcal{F}_p$, is affixed to the virtual vehicle CG, as in Fig. 3.1. The three orthonormal basis vectors of $\mathcal{F}_p$, denoted $\boldsymbol{T}(\ell)$ (tangent vector), $\boldsymbol{N}_1(\ell)$ (first normal

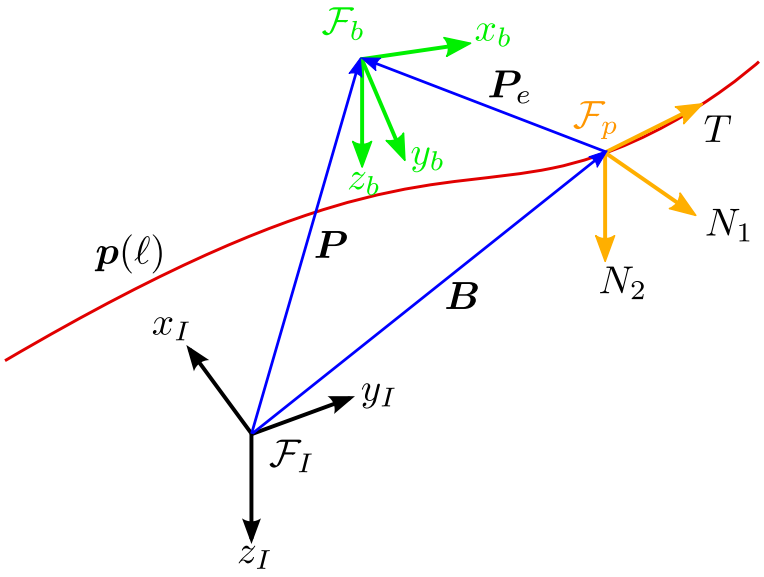

Figure 3.2: The parallel transport frame is related to the current UAS position by the error vector $\boldsymbol{P}_e$.

vector), and $\boldsymbol{N}_2(\ell)$ (second normal vector), satisfy the dynamic equation

$$\begin{bmatrix} d\boldsymbol{T}(\ell)/d\ell \\ d\boldsymbol{N}_1(\ell)/d\ell \\ d\boldsymbol{N}_2(\ell)/d\ell \end{bmatrix} = \begin{bmatrix} 0 & k_1(\ell) & k_2(\ell) \\ -k_1(\ell) & 0 & 0 \\ -k_2(\ell) & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{T}(\ell) \\ \boldsymbol{N}_1(\ell) \\ \boldsymbol{N}_2(\ell) \end{bmatrix}, \tag{3.1}$$

where $k_1(\ell)$ and $k_2(\ell)$ are parameters that vary over ℓ.

Let $\boldsymbol{P}_e = [x_e, y_e, z_e]^T$ be the vector denoting the difference between the UAS and virtual vehicle positions, expressed in the parallel transport frame $\mathcal{F}_p$, as shown in Fig. 3.2. Also, define a local UAS frame $\mathcal{F}_{w'}$ as the rotation of the UAS wind reference frame $\mathcal{F}_w$ onto the local level plane, as shown in Fig. 3.1. This frame's orientation can be described relative to $\mathcal{F}_p$ through a set of three relative error Euler angles, $\boldsymbol{\Lambda}_e = [\phi_e, \theta_e, \psi_e]^T$. Through a

small angle approximation, it is assumed that the UAS roll, pitch, and yaw rates, $\mathbf{\Omega}$, as defined in the frame $\mathcal{F}_{w'}$, are approximately equal to those in the body-fixed reference frame. After differentiation and simplification, we obtain the following equations representing the kinematic position error dynamics $\dot{\boldsymbol{P}}_e$ for the combined UAS and virtual vehicle system:

$$\begin{aligned}
\dot{x}_e &= -\dot{\ell}(1 - k_1(\ell)y_e - k_2(\ell)z_e) + V\cos\theta_e\cos\psi_e, \\
\dot{y}_e &= -\dot{\ell}k_1(\ell)x_e + V\cos\theta_e\sin\psi_e, \\
\dot{z}_e &= -\dot{\ell}k_2(\ell)x_e - V\sin\theta_e.
\end{aligned} \tag{3.2}$$

The attitude error dynamics, $\dot{\mathbf{\Lambda}}_e$, can be derived in a similar fashion using the Euler kinematic equation (2.8) as

$$\begin{aligned}
\dot{\phi}_e &= \dot{\ell}k_2(\ell)\sin\phi_e\sec\theta_e + p + r\cos\phi_e\tan\theta_e + q\sin\phi_e\tan\theta_e, \\
\dot{\theta}_e &= \dot{\ell}k_2(\ell)\cos\phi_e + q\cos\phi_e - r\sin\phi_e, \\
\dot{\psi}_e &= -\dot{\ell}(k_1(\ell) - k_2(\ell)\tan\theta_e\sin\psi_e) + q\sin\phi_e\sec\theta_e + r\cos\phi_e\sec\theta_e.
\end{aligned} \tag{3.3}$$

Together, (3.2) and (3.3) describe the path-following error of the combined UAS and virtual vehicle systems. Finally, the dynamics of the virtual vehicle are defined as

$$\dot{\ell} = K_1 x_e + V\cos\theta_e\cos\psi_e, \tag{3.4}$$

where K_1 is some positive constant.

Following the example set in [12], the error Euler angles θ_e and ψ_e are shaped using approach

angle functions to improve control performance. In a departure from the method used in [12], hyperbolic tangent functions are used as the approach angle functions for convenience. The approach angles work to ensure that the vehicle is approaching the path at all times, and provide an extra degree of freedom in the aggressiveness of the tracking behavior. The approach angles θ_δ and ψ_δ are defined as

$$\theta_\delta(z_e) = \theta_m \tanh\left(z_e/C_1\right),$$

$$\psi_\delta(y_e) = \psi_m \tanh\left(y_e/C_2\right),$$

where θ_m and ψ_m are the maximum desired approach angles, and C_1 and C_2 are scaling factors to determine the magnitude of position error corresponding to the maximum allowed approach angle. To incorporate these shaping functions, we redefine the $\mathbf{\Lambda}_e$ measurement as $[\phi_e,\, \theta_e - \theta_\delta,\, \psi_e - \psi_\delta]^T$. A sample shaping function for height error is shown in Fig. 3.3 with $C_1 = 8$ and $\theta_m = 15°$. Note that the approach angle is equivalently 0 when there is no height error, and saturates at $\pm\theta_m$ when the height error is large.

In [12], a nonlinear outer-loop control law is developed via backstepping, whereby the pitch and yaw rates, q and r, play the role of virtual control inputs. Pitch and yaw rate commands $(q_c,\, r_c)$ are generated by the nonlinear control law and then tracked by a Piccolo autopilot augmented by an $\mathcal{L}_1$ adaptive controller in an inner control loop. These commands are

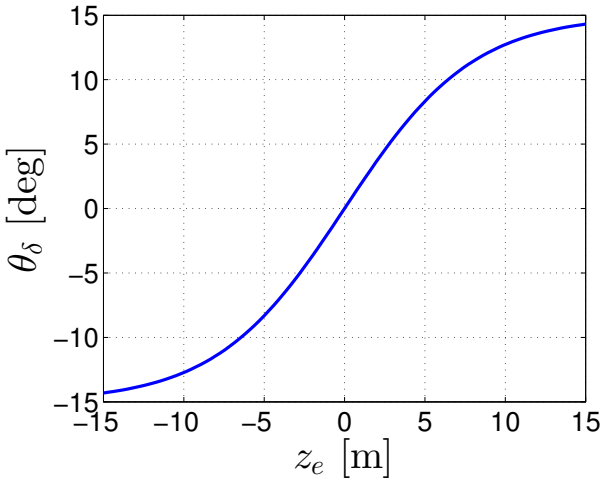

Figure 3.3: A sample approach angle shaping function guides the aircraft to the correct altitude and saturates at $\theta_\delta = 15°$.

---

defined in [12] as

$$\begin{bmatrix} q_c \\ r_c \end{bmatrix} = Q_c^{-1}(\theta_e, \phi_e) \left(\begin{bmatrix} \theta_c(\cdot) \\ \psi_c(\cdot) \end{bmatrix} - D_c(\theta_e, \psi_e, \ell) \right), \tag{3.5}$$

with the following auxiliary definitions:

$$Q_c(\theta_e, \phi_e) = \begin{bmatrix} \cos\phi_e & -\sin\phi_e \\ \frac{\sin\phi_e}{\cos\theta_e} & \frac{\cos\phi_e}{\cos\theta_e} \end{bmatrix},$$

$$D_c(\theta_e, \psi_e, \ell) = \dot{\ell} \begin{bmatrix} k_2(\ell)\cos\psi_e \\ -k_1(\ell) + k_2(\ell)\tan\theta_e \sin\psi_e \end{bmatrix},$$

$$\theta_a = \sin^{-1}\frac{z_e}{|z_e| + d_1}, \quad \psi_a = \sin^{-1}\frac{-y_e}{|y_e| + d_2},$$

$$\theta_c = -K_2(\theta_e - \theta_a) + C_3 z_e V_a \frac{\sin\theta_e - \sin\theta_a}{\theta_e - \theta_a} + \dot{\theta}_a,$$

$$\psi_c = -K_3(\psi_e - \psi_a) + C_3 y_e V_a \cos\theta_e \frac{\sin\psi_e - \sin\psi_a}{\psi_e - \psi_a} + \dot{\psi}_a,$$

where d_1, d_2, K_2, K_3 and C_3 are positive constants.

In order to develop a point of reference to compare controllers to, a similar approach to the one in [12] is taken in this work. The nonlinear control law (3.5) is used to generate pitch and yaw rate commands, q_c and r_c, which are then tracked through the disturbance channel by a standard $\mathcal{H}_\infty$ controller.

3.1.1 Planar Path-Following

For this work, the set of permissible geometric paths to be followed is restricted to 2-D paths in the $\boldsymbol{x}_I$-$\boldsymbol{y}_I$ plane. For this special case of paths, many simplifications can be made to the relevant system dynamics. The largest simplification is that the UAS elevation and bank angles can be considered identically equal to the previously defined error angles θ_e and ϕ_e. Additionally, the $k_2(\ell)$ path parameter will be identically zero for all time, allowing $\dot{\boldsymbol{P}}_e$ and $\dot{\boldsymbol{\Lambda}}_e$ to be simplified. Given $k_2(\ell) = 0$, the remaining parallel transport frame parameter, $k_1(\ell)$, can be ascribed a more physical interpretation, namely, the inverse of the current curvature of the path, $R(\ell)$, as

$$k_1(\ell) = \frac{1}{R(\ell)}. \tag{3.6}$$

A straight path segment therefore corresponds to an infinite radius of curvature and a parameter value of $k_1(\ell) = 0$. Conversely, as the radius of curvature gets smaller and the corresponding turn gets tighter, the magnitude of $k_1(\ell)$ increases.

3.2 Path-Following Controller Synthesis

The three control systems discussed in this section are synthesized based on linear plant models, G, shown in Figs. 2.3 and 2.4, which are obtained by linearizing the nonlinear equations of motion derived in Sections 2.2 and 3.1 about the trim conditions for straight and level flight or steady banked turns. For simplicity, we drop the dependence of the parameter k_1 on the path location ℓ. Three linearized models are developed for the synthesis of three corresponding controllers. An LPV model dependent on k_1 is first developed from the UAS and path-following dynamic equations, and an LPV controller synthesized. An LTI plant model is then formulated by setting $k_1 = 0$ in the dynamics of the LPV model. Finally, a second LTI model is developed based on the standard UAS equations of motion in order to synthesize an inner-loop controller to track rates provided by the outer-loop backstepping controller.

3.2.1 LPV Plant Model Formulation

The lumped path-following and UAS system is composed of equations (2.10), (3.2), and (3.3). Define the lumped UAS path-following state vector as $\boldsymbol{x} = [\boldsymbol{V}^T, \boldsymbol{\Omega}^T, \boldsymbol{P}_e^T, \boldsymbol{\Lambda}_e^T, \boldsymbol{x}_a^T]^T$, the control input as $\boldsymbol{u} = [\delta_{Ec}, \delta_{Ac}, \delta_{Rc}, \delta_T]^T$, the measurements as $\boldsymbol{y} = [p, q, r, \phi_e, \theta_e, \psi_e, V_a, \ldots$ $x_e, y_e, z_e]^T$, and the exogenous disturbances as $\boldsymbol{w} = [\boldsymbol{V}_w^T, \boldsymbol{w}_m^T]^T$, where $\boldsymbol{w}$ represents finite energy disturbances in ℓ_2, $\boldsymbol{w}_m = [m_p, m_q, m_r, m_\phi, m_\theta, m_\psi, m_{Va}, m_x, m_y, m_z]^T$ represents measurement noise, and $\boldsymbol{x}_a$ represents the actuator states. The vectors $\boldsymbol{x}(t)$, $\boldsymbol{u}(t)$, $\boldsymbol{y}(t)$, and

$\boldsymbol{w}(t)$ are real with dimensions denoted by n, n_u, n_y, and n_w, respectively.

Excluding the actuator dynamics, the 12 differential equations representing the lumped UAS and path-following systems for planar path tracking are

$$\begin{aligned}
\dot{p} &= \frac{M_l + (I_y - I_z)qr}{I_x}, \\
\dot{q} &= \frac{M_m + (I_z - I_x)pr}{I_y}, \\
\dot{r} &= \frac{M_n + (I_x - I_y)pq}{I_z}, \\
\dot{u} &= -qw + rv + \frac{F_x}{m} - g\sin\theta_e, \\
\dot{v} &= -ru + pw + \frac{F_y}{m} + g\cos\theta_e\sin\phi_e, \\
\dot{w} &= -pv + qu + \frac{F_z}{m} + g\cos\theta_e\cos\phi_e, \\
\dot{\phi}_e &= p + r\cos\phi_e\tan\theta_e + q\sin\phi_e\tan\theta_e, \\
\dot{\theta}_e &= q\cos\phi_e - r\sin\phi_e, \\
\dot{\psi}_e &= -\dot{\ell}k_1(\ell) + q\sin\phi_e\sec\theta_e + r\cos\phi_e\sec\theta_e, \\
\dot{x}_e &= -\dot{\ell}(1 - k_1(\ell)y_e) + V\cos\theta_e\cos\psi_e, \\
\dot{y}_e &= -\dot{\ell}k_1(\ell)x_e + V\cos\theta_e\sin\psi_e, \\
\dot{z}_e &= -V\sin\theta_e.
\end{aligned} \tag{3.7}$$

The differential equations of motion, performance output, and measurement output can then be written as $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{u}, k_1)$, $\boldsymbol{z} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{u})$, and $\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{w})$, respectively.

Parameter-Varying Trim Point

Assuming constant altitude flight, the required trim bank angle, $\phi_{e_{tr}}$, to maintain a steady level turn of radius R and tangential velocity V_t can be determined by relating the aircraft's lift and centripetal acceleration as

$$\tan \phi_{e_{tr}} = \frac{V_t^2}{gR}.$$

Note that since our aircraft is trimmed with no disturbances, or $\boldsymbol{V}_w = \mathbf{0}$, V_t is equivalent to the aircraft's desired airspeed, $V_{a_{tr}}$. Given our choice of bounded path curvatures, we simply employ (3.6) and a small angle assumption in order to solve for our trim bank angle as a function of k_1, namely:

$$\phi_{e_{tr}}(k_1) = \frac{k_1 V_{a_{tr}}^2}{g}. \tag{3.8}$$

The trim states are determined by using the `MATLAB` function `fmincon` to minimize the cost function $\dot{\boldsymbol{V}}^T\dot{\boldsymbol{V}} + \dot{\boldsymbol{\Omega}}^T\dot{\boldsymbol{\Omega}} + \dot{\boldsymbol{\Lambda}}^T\dot{\boldsymbol{\Lambda}} + \dot{H}^2 + (\phi - \phi_{tr}(k_1))^2 + (V_a - V_{a_{tr}})^2$ with respect to (2.10) where $V_{a_{tr}} = 13\,\mathrm{m/s}$ is the desired airspeed and $\phi_{tr}(k_1)$ is determined using (3.8).

Trim points are calculated over the range $-0.025 \leq k_1 \leq 0.025$. Trim states, measurements, and control inputs are given in Table 3.1 for straight and level flight ($k_1 = 0$), a moderate turn ($k_1 = \pm 0.0141$), and an aggressive turn ($k_1 = \pm 0.0250$). Note that the control input trims are given in terms of pulse widths. The moderate turn with $k_1 = 0.0141$ corresponds to a bank angle of approximately 14° for the chosen airspeed and a turn radius of 70.9 meters.

The aggressive turn with $k_1 = 0.0250$ corresponds to a bank angle of approximately 24.7° and a turn radius of 40 meters. A more aggressive turn radius could not be used for the Telemaster due to thrust saturation.

Table 3.1: LPV Trim Points

k_1	-0.025	-0.0141	0	0.0141	0.025	m^{-1}
p	-0.041	0.021	-0.004	-0.029	-0.048	rad/s
q	0.065	0.019	0.000	0.026	0.076	rad/s
r	-0.290	-0.152	0.026	0.204	0.341	rad/s
u	12.875	12.874	12.873	12.871	12.870	m/s
v	-0.025	-0.017	-0.006	0.005	0.013	m/s
w	1.749	1.801	1.831	1.822	1.787	m/s
V_a	13.00	13.00	13.00	13.00	13.00	m/s
ϕ/ϕ_e	-0.431	-0.243	0.000	0.243	0.431	rad
θ	0.136	0.136	0.137	0.137	0.138	rad
θ_e	-0.001	-0.001	0.000	0.001	0.001	rad
ψ/ψ_e	0.000	0.000	0.000	0.000	0.000	rad
$x_e/y_e/z_e$	0.000	0.000	0.000	0.000	0.000	m
δ_E	0.142	0.118	0.108	0.123	0.151	μs
δ_A	0.040	0.023	0.002	-0.019	-0.036	μs
δ_R	-0.032	-0.049	-0.071	-0.094	-0.111	μs
δ_T	0.622	0.592	0.577	0.590	0.618	μs

The linear and quadratic trim state fits are given by (3.9) and shown in Fig. 3.4. $u_{tr}(k_1)$, $v_{tr}(k_1)$, $p_{tr}(k_1)$, $r_{tr}(k_1)$, $\phi_{e_{tr}}(k_1)$, and $\theta_{e_{tr}}(k_1)$ are linear functions of k_1, while $w_{tr}(k_1)$ and $q_{tr}(k_1)$ are quadratic functions of k_1. As expected, the bank angle $\phi_{e_{tr}}(k_1)$ and yaw rate $r_{tr}(k_1)$ had the largest variation with k_1.

The actuator and thrust command trim fits are given by (3.9) and shown in Fig. 3.5. The lateral-directional commands $\delta_{A_{tr}}(k_1)$ and $\delta_{R_{tr}}(k_1)$ have linear fits, while the longitudinal commands $\delta_{E_{tr}}(k_1)$ and $\delta_{T_{tr}}(k_1)$ have quadratic fits.

It was found that including p_{tr}, q_{tr}, $\delta_{E_{tr}}$, and $\delta_{T_{tr}}$ as parameter-varying trims in the synthesis process did not improve controller performance, and simply increased the size of the corresponding LFR, and subsequently the computational complexity of the synthesis process. The four trim states were instead fixed at their average values.

$$
\begin{aligned}
u_{tr}(k_1) &= 0.1032k_1 + 12.8732, \\
v_{tr}(k_1) &= 0.7566k_1 - 0.0059, \\
w_{tr}(k_1) &= -101.6138k_1^2 + 0.7496k_1 + 1.8313, \\
p_{tr}(k_1) &= -0.0140k_1 - 0.0041, \\
q_{tr}(k_1) &= 225.9918k_1^2 + 0.4574k_1 - 0.0003, \\
r_{tr}(k_1) &= 12.6170k_1 + 0.0258, \\
\phi_{e_{tr}}(k_1) &= 17.2268k_1, \\
\theta_{e_{tr}}(k_1) &= 0.0485k_1, \\
\delta_{E_{tr}}(k_1) &= 61.9902k_1^2 + 0.1725k_1 + 0.0108, \\
\delta_{A_{tr}}(k_1) &= -1.5132k_1 + 0.0021, \\
\delta_{R_{tr}}(k_1) &= -1.5772k_1 - 0.0714, \\
\delta_{T_{tr}}(k_1) &= 68.4314k_1 - 0.0859k_1 + 0.5774.
\end{aligned}
\tag{3.9}
$$

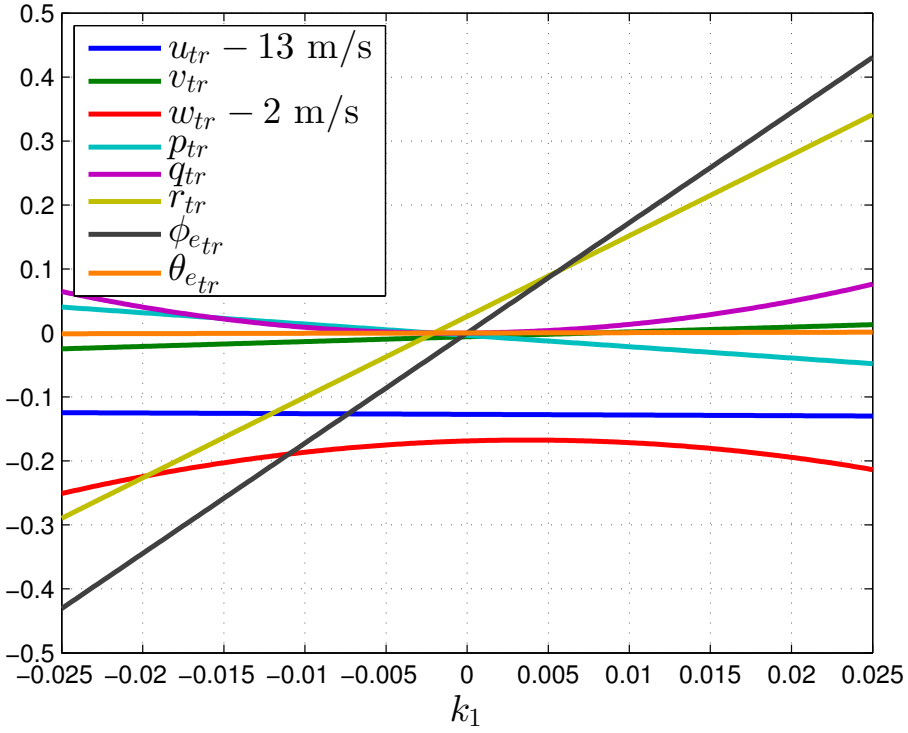

Figure 3.4: LPV Trim States

Linear Parameter-Varying Model

Linearizing $\boldsymbol{f}(\cdot)$, $\boldsymbol{g}(\cdot)$, and $\boldsymbol{h}(\cdot)$ about the parameter varying trim points $\boldsymbol{x}_{tr}(k_1)$, $\boldsymbol{u}_{tr}(k_1)$, and $\boldsymbol{w}_{tr} = \mathbf{0}$ yields the continuous-time LPV state space equations

$$\begin{bmatrix} \dot{\bar{\boldsymbol{x}}}(t) \\ \bar{\boldsymbol{z}}(t) \\ \bar{\boldsymbol{y}}(t) \end{bmatrix} = \begin{bmatrix} A^c(k_1) & B_1^{cw}(k_1) & B_2^c(k_1) \\ C_1^c(k_1) & D_{11}^{cw}(k_1) & D_{12}^c(k_1) \\ C_2^c(k_1) & D_{21}^{cw}(k_1) & 0 \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}(t) \\ \boldsymbol{w}(t) \\ \bar{\boldsymbol{u}}(t) \end{bmatrix}, \tag{3.10}$$

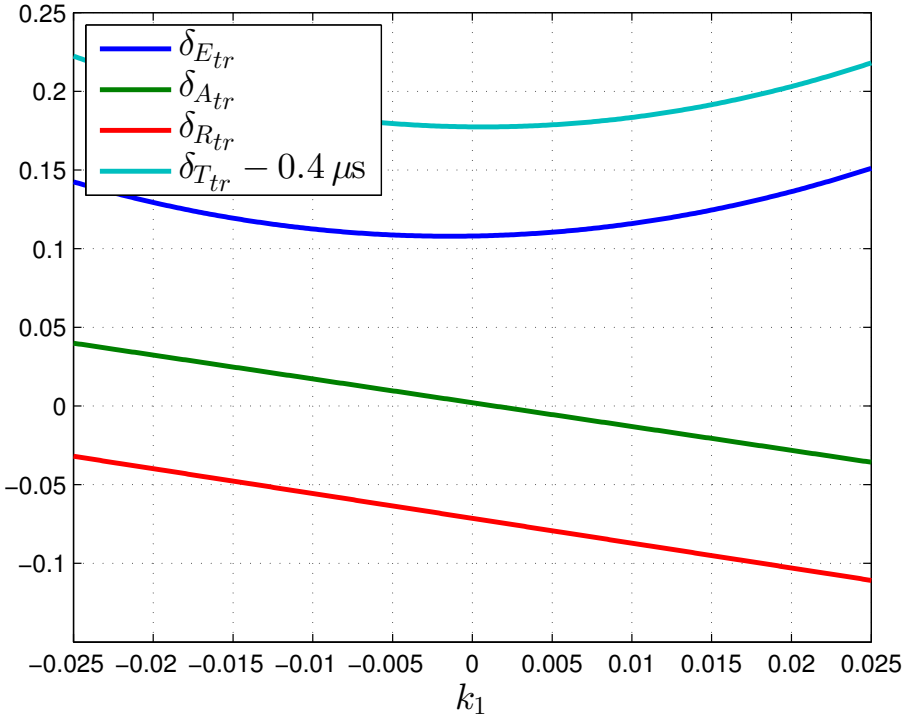

Figure 3.5: LPV Control Input Trims

where t is continuous time, $\bar{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_{tr}$, $\bar{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_{tr}$, $\bar{\boldsymbol{y}} = \boldsymbol{y} - \boldsymbol{y}_{tr}$, and $\bar{\boldsymbol{x}}(0) = \boldsymbol{0}$. The Jacobians are symbolically calculated as

$$A^c(k_1) = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))}, \quad B_1^{cw}(k_1) = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))},$$
$$B_2^c(k_1) = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))}, \quad C_1^c(k_1) = \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))},$$
$$D_{11}^{cw}(k_1) = \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{w}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))}, \quad D_{12}^c(k_1) = \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))},$$
$$C_2^c(k_1) = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))}, \quad D_{21}^{cw}(k_1) = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{w}}\right|_{(\boldsymbol{x}_{tr}(k_1),\boldsymbol{w}_{tr},\boldsymbol{u}_{tr}(k_1))}.$$

Note that $\boldsymbol{P}_e = \bar{\boldsymbol{P}}_e$ and $\boldsymbol{\Lambda}_e = \bar{\boldsymbol{\Lambda}}_e$.

As the thrust model is lookup-table based, it is linearized prior to the symbolic Jacobian

calculations using the small-perturbation method, yielding the linear model

$$T = T_{\delta_T}\delta_T + T_{Va}V_a,$$

where T_{δ_T} and T_{Va} are constants.

The matrices in (3.10) have nonlinear dependence on the varying parameter k_1 due to the existence of trigonometric functions within the equations of motion (3.7). Since $\phi_{e_{tr}}(k_1)$ is bounded as a result of k_1 being bounded, we approximate the zero centered trigonometric functions of $\phi_{e_{tr}}(k_1)$ by the low order Taylor series representations

$$\begin{aligned} \sin\phi_{e_{tr}}(k_1) &\approx \phi_{e_{tr}}(k_1), \\ \cos\phi_{e_{tr}}(k_1) &\approx 1 - \frac{1}{2}\phi_{e_{tr}}(k_1)^2. \end{aligned} \tag{3.11}$$

Similar functions are defined for $\theta_{e_{tr}}(k_1)$. Substituting these functions into the matrices in (3.10) ensures that all matrix terms are rationally dependent on k_1.

In order to achieve robust performance in the midst of disturbances, a weighting matrix is defined based on the worst-case expected atmospheric disturbances, and 3 times the expected sensor noise standard deviations as $W_w = \mathrm{diag}(3I_3, 0.5I_3, 0.01I_3, 2I_4)$. The disturbance matrices are then redefined as $B_1^c(k_1) = W_w B_1^{cw}(k_1)$, $D_{11}^c(k_1) = W_w D_{11}^{cw}(k_1)$, and $D_{21}^c(k_1) = W_w D_{21}^{cw}(k_1)$.

As the controller generates new actuator commands at 25 Hz, the model is discretized with

a sampling time of $\tau = 0.04\,\mathrm{s}$ as

$$A(k_1) = (I + \tau A^c(k_1)), \quad B_i(k_1) = \tau B_i^c(k_1), \quad C_i(k_1) = C_i^c(k_1), \quad D_{ij}(k_1) = D_{ij}^c(k_1),$$

with $i, j = 1, 2$. Euler discretization is used to maintain minimum order parameter dependence on k_1. The discrete-time system is expressed as

$$\begin{bmatrix} \bar{\boldsymbol{x}}_{k+1} \\ \bar{\boldsymbol{z}}_k \\ \bar{\boldsymbol{y}}_k \end{bmatrix} = \begin{bmatrix} A(k_1) & B_1(k_1) & B_2(k_1) \\ C_1(k_1) & D_{11}(k_1) & D_{12}(k_1) \\ C_2(k_1) & D_{21}(k_1) & 0 \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}_k \\ \boldsymbol{w}_k \\ \bar{\boldsymbol{u}}_k \end{bmatrix}, \tag{3.12}$$

with $\bar{\boldsymbol{x}}_0 = \mathbf{0}$, where $\bar{\boldsymbol{x}}_k = \bar{\boldsymbol{x}}(k\tau)$, $\bar{\boldsymbol{u}}_k = \bar{\boldsymbol{u}}(k\tau)$, and $\bar{\boldsymbol{y}}_k = \bar{\boldsymbol{y}}(k\tau)$.

Since the system has polynomial dependence on the parameter k_1, it can be equivalently represented by the LFR shown in Fig. 2.4, and defined as

$$\begin{bmatrix} \bar{\boldsymbol{x}}_{k+1} \\ \boldsymbol{\varphi}_k \\ \bar{\boldsymbol{z}}_k \\ \bar{\boldsymbol{y}}_k \end{bmatrix} = \begin{bmatrix} A_{ss} & A_{sp} & B_{1s} & B_{2s} \\ A_{ps} & A_{pp} & B_{1p} & B_{2p} \\ C_{1s} & C_{1p} & D_{11} & D_{12} \\ C_{2s} & C_{2p} & D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}_k \\ \boldsymbol{\vartheta}_k \\ \boldsymbol{w}_k \\ \bar{\boldsymbol{u}}_k \end{bmatrix}, \tag{3.13}$$

where $\boldsymbol{\vartheta}_k = k_1 \boldsymbol{\varphi}_k$, $\bar{\boldsymbol{x}}_0 = \mathbf{0}$, $\boldsymbol{\varphi}_k \in \mathbb{R}^{n_\delta}$, and $\boldsymbol{\vartheta}_k \in \mathbb{R}^{n_\delta}$.

3.2.2 LPV $\mathcal{H}_\infty$ Controller

The self-scheduled LPV controller synthesis is based on 2.5.2. The performance output for the LPV controller is chosen as

$$\bar{\boldsymbol{z}} = [0.08\bar{V}_a,\ 0.08\bar{\alpha},\ 0.08\bar{\beta},\ 0.4\phi_e,\ 2.8\theta_e,\ 0.064x_e,\ 0.16(y_e + 0.16\bar{\delta}_A + 0.12\bar{\delta}_R),$$
$$0.16(\psi_e - 0.064\bar{\delta}_R),\ 0.16(z_e + 0.04\bar{\delta}_T),\ 8\text{E-}3\bar{\delta}_E,\ 8\text{E-}4\bar{\delta}_A,\ 8\text{E-}4\bar{\delta}_R,\ 0.8\bar{\delta}_T]^T.$$

Note that the altitude error is coupled with the throttle command, δ_T, in order to guide the controller to utilize throttle to maintain airspeed as opposed to elevator. Also, the cross-track error is coupled with aileron and rudder to guide the controller to perform coordinated turns. This was found to both increase path-following performance while decreasing the likelihood of saturating the rudder deflection. The controller synthesis problem was solved in 18.3 seconds, and the optimal value of γ was found to be $\gamma_{min} = 1$, which was relaxed to $\gamma = 1.5$ to obtain satisfactory robust performance, and the synthesis problem was resolved.

3.2.3 Standard LTI $\mathcal{H}_\infty$ Controller

The LTI plant for straight and level flight is found by taking $k_1 = 0$ in (3.12). This corresponds to a straight and level trim point. The time-invariant discrete-time state space

equations are

$$\begin{bmatrix} \bar{\boldsymbol{x}}_{k+1} \\ \bar{\boldsymbol{z}}_k \\ \bar{\boldsymbol{y}}_k \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & 0 \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{x}}_k \\ \boldsymbol{w}_k \\ \bar{\boldsymbol{u}}_k \end{bmatrix}. \tag{3.14}$$

For the standard $\mathcal{H}_\infty$ controller, henceforth referred to as the LTI controller, the performance output in Fig. 2.3 is chosen as

$$\bar{\boldsymbol{z}} = [0.085\bar{V}_a,\ 8.5\text{E-}5\bar{\alpha},\ 0.0765\bar{\beta},\ 0.323\phi_e,\ 0.85\theta_e,\ 0.51\bar{\delta}_E,\ 0.17(y_e + 0.425\bar{\delta}_A - 0.03\bar{\delta}_R),$$
$$0.595(\psi_e - 0.15\bar{\delta}_R),\ 0.8\bar{\delta}_T,\ 0.0255x_e,\ 0.153(z_e + 0.0382\bar{\delta}_T),\ 8.5\text{E-}4\bar{\delta}_A,\ 8.5\text{E-}4\bar{\delta}_R]^T.$$

Synthesis for the standard $\mathcal{H}_\infty$ controller is described in detail in 2.5.1. The controller synthesis problem was solved in 7.8 seconds with $\gamma_{min} = 1.01$. The value of γ was then relaxed by 50% to 1.515 to increase robustness, and the control synthesis problem re-solved.

3.2.4 Rate-Tracking Controller

The LTI plant for the baseline rate-tracking controller is obtained in a similar manner by instead linearizing the parameter-independent UAS equations of motion

$$
\begin{aligned}
\dot{p} &= \frac{M_l + (I_y - I_z)qr}{I_x}, \\
\dot{q} &= \frac{M_m + (I_z - I_x)pr}{I_y}, \\
\dot{r} &= \frac{M_n + (I_x - I_y)pq}{I_z}, \\
\dot{u} &= -qw + rv + \frac{F_x}{m} - g\sin\theta, \\
\dot{v} &= -ru + pw + \frac{F_y}{m} + g\cos\theta\sin\phi, \\
\dot{w} &= -pv + qu + \frac{F_z}{m} + g\cos\theta\cos\phi, \\
\dot{\phi} &= p + r\cos\phi\tan\theta + q\sin\phi\tan\theta, \\
\dot{\theta} &= q\cos\phi - r\sin\phi, \\
\dot{\psi} &= q\sin\phi\sec\theta + r\cos\phi\sec\theta, \\
\dot{X} &= u\cos\theta\cos(\psi - \psi_0) + v(\sin\phi\sin\theta\cos(\psi - \psi_0) - \cos\phi\sin(\psi - \psi_0)) + \dots \\
&\quad w(\cos\phi\sin\theta\cos(\psi - \psi_0) + \sin\phi\sin(\psi - \psi_0), \\
\dot{Y} &= u\cos\theta\sin(\psi - \psi_0) + v(\sin\phi\sin\theta\sin(\psi - \psi_0) - \cos\phi\cos(\psi - \psi_0)) + \dots \\
&\quad w(\cos\phi\sin\theta\sin(\psi - \psi_0) + \sin\phi\cos(\psi - \psi_0), \\
\dot{h} &= u\sin\theta - v\sin\phi\cos\theta - w\cos\phi\cos\theta,
\end{aligned}
\tag{3.15}
$$

and actuator dynamics about the straight and level trim point in Table 3.1.

The state, measurement, and disturbance vectors are defined as $\boldsymbol{x} = [\boldsymbol{V}^T,\, \boldsymbol{\Omega}^T,\, \boldsymbol{P}^T,\, \boldsymbol{\Lambda}^T,\, \boldsymbol{x}_a^T]^T$, $\boldsymbol{w} = [\boldsymbol{V}_w^T,\, \boldsymbol{w}_m^T,\, \boldsymbol{w}_c^T]^T$, and $\boldsymbol{y} = [p,\, q - q_c,\, r - r_c,\, \phi,\, \theta,\, V_a]^T$, where $\boldsymbol{w}_m = [m_p,\, m_q,\, m_r,\, m_\phi, \dots$

$m_\theta,\ m_{V_a}]^T$, and $\boldsymbol{w}_c = [q_c,\ r_c]^T$ from (3.5). Recall that q_c and r_c are the commanded pitch and yaw rates, respectively. The weighting matrix used is $W_w = \text{diag}(3I_3, 0.5I_3, 0.01I_2, 2, I_2)$.

Similar to the example found in [63], the ideal pitch and yaw rates given by (3.5) are first passed through a second order filter with $\omega_n = 12$ and $\zeta = 0.8$ before being augmented to the disturbance vector and tracked by the standard $\mathcal{H}_\infty$ controller. The performance output is chosen as

$$\bar{\boldsymbol{z}} = [q - q_c,\ 1.52(r - r_c + 1.57\bar{\delta}_A),\ 0.02\bar{V}_a,\ 0.6\bar{\alpha},\ 0.6\bar{\beta},\ \bar{\delta}_E,\ 0.84\bar{\delta}_A,\ 0.34\bar{\delta}_R,\ 0.1\bar{\delta}_T]^T.$$

The controller synthesis problem was solved in 5.7 seconds, with $\gamma_{min} = 1$. The value of γ was relaxed by 50% to 1.5 to increase robustness, and the control synthesis problem was re-solved. This baseline controller will henceforth be referred to as the rate-tracking (RT) controller.

3.3 Simulation Environment

The rigorous `MATLAB` simulation environment presented in Section 2.3 is used to test the three controllers. The simulation environment is designed to subject the small UAS to proportionally significant atmospheric disturbances while mimicking the implementation of the controller onboard the Telemaster UAS platform. The UAS is subject to a 3 m/s steady northern wind in addition to moderate turbulent gusts from the low altitude Dryden turbu-

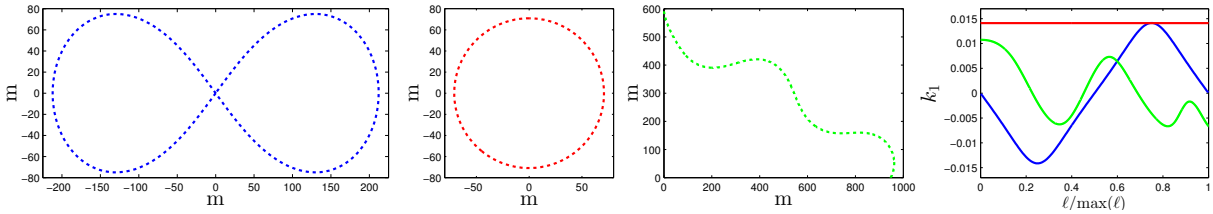

Figure 3.6: A lemniscate (blue), circular (red), and a random (green) reference path and their associated k_1 histories.

lence model presented in Section 2.3.1, and sensor noise.

A one-step time delay is also included in simulations to mimic the control system on-board the Telemaster UAS. Namely, measurements taken at the discrete time instant k are used to calculate the control input applied at time $k+1$. Recall that the UAS platform operates at 25 Hz.

3.3.1 Reference Paths

Controllers are designed and tested on various path types, shown in Fig. 3.6. The lemniscate path has a smoothly varying k_1 history and is generated by the function

$$N_{ref} = \frac{3\cos(\xi)}{k_{1_{max}}(1+\sin(\xi)^2)}, \quad E_{ref} = \frac{3\sin(\xi)\cos(\xi)}{k_{1_{max}}(1+\sin(\xi)^2)}, \tag{3.16}$$

where $k_{1_{max}}$ is a scaling parameter representing the tightest turn on the path and $H_{ref} = 0$ for all $\xi \in [\pi/2,\ 9\pi/2]$. A circular path with a constant k_1 value equal to $k_{1_{max}}$ is also used,

generated by the function

$$N_{ref} = \frac{\cos(\xi)}{k_{1max}}, \quad E_{ref} = \frac{\sin(\xi)}{k_{1max}}, \quad \xi \in [0,\ 2\pi]. \tag{3.17}$$

Finally, a 100,000 m long random path is followed. To generate the random path, random inflection points were first generated from uniform distributions across the applicable ranges of ℓ and k_1. A natural spline is then used to create a minimal overshoot, smoothly varying curve for k_1, saturating the curve at k_{1max}.

The paths shown in Fig. 3.6 correspond to $k_{1max} = 0.0141$. If followed perfectly at the desired airspeed of 13 m/s with no wind, the lemniscate path could be traversed in 74.16 s and the circle in 34.28 s. In order for the simulation results to be statistically meaningful, each controller was used to track both patterns 1000 times consecutively.

Furthermore, a set of more aggressive paths are also attempted with $k_{1max} = 0.0250$. The ideal path time for the aggressive lemniscate is 48.4 s, and for the circle, 19.3 s.

3.3.2 Performance Metrics

Several performance metrics are used in order to quantitatively compare the performance of the various controllers. The three criteria used are path error, control effort, and average path time.

The mean path error (MPE) corresponds to the average error between the UAS and the cho-

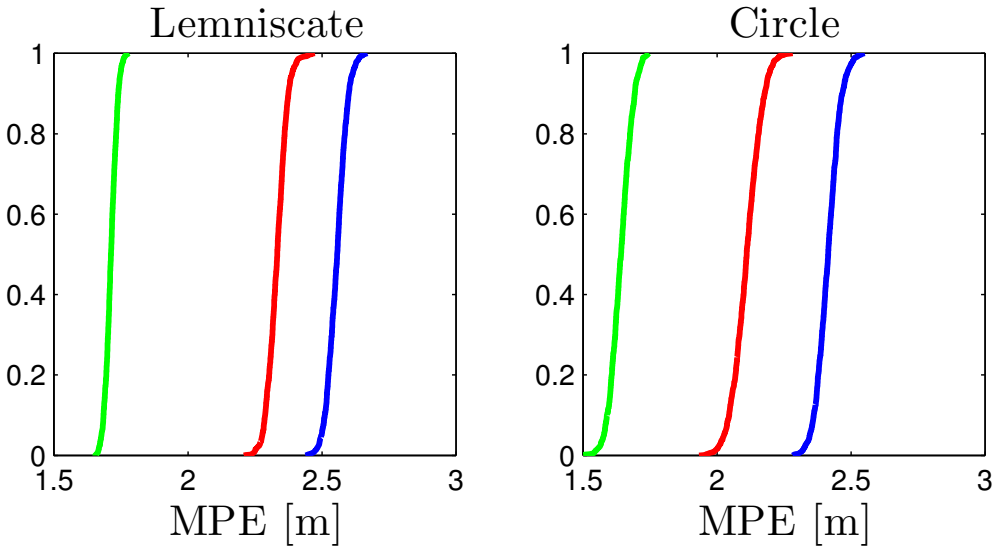

Figure 3.7: RT, LTI, and LPV MPEs over 1000 loops for $|k_1| \leq 0.0141$.

sen path. In order to accurately calculate the path error, the vector ξ is finely discretized by 10,000 points, and the resulting distance between points in $\boldsymbol{x}_I$-$\boldsymbol{y}_I$ is assumed to be negligible. The set of parameterized reference points $\boldsymbol{p} = (N, E, H)$ is defined as

$$R = \{\boldsymbol{p}(\ell_i) \text{ for i} = 1, 2, \cdots, \text{n}_\xi\}, \tag{3.18}$$

where n_ξ is the length of ℓ. Note that as $n_\xi \to \infty$, the distance between points in $\boldsymbol{x}_I$-$\boldsymbol{y}_I$ approaches 0. The minimum distance between any point $\boldsymbol{a}$ and the reference path R is then defined as

$$\text{dist}(\boldsymbol{a}, R) = \{\inf \|\boldsymbol{a} - \boldsymbol{b}\|_2 \mid \boldsymbol{b} \in R\}, \tag{3.19}$$

where $\|\boldsymbol{q}\|_2$ denotes the Euclidean norm of $\boldsymbol{q}$. Given equations (3.18) and (3.19), we define

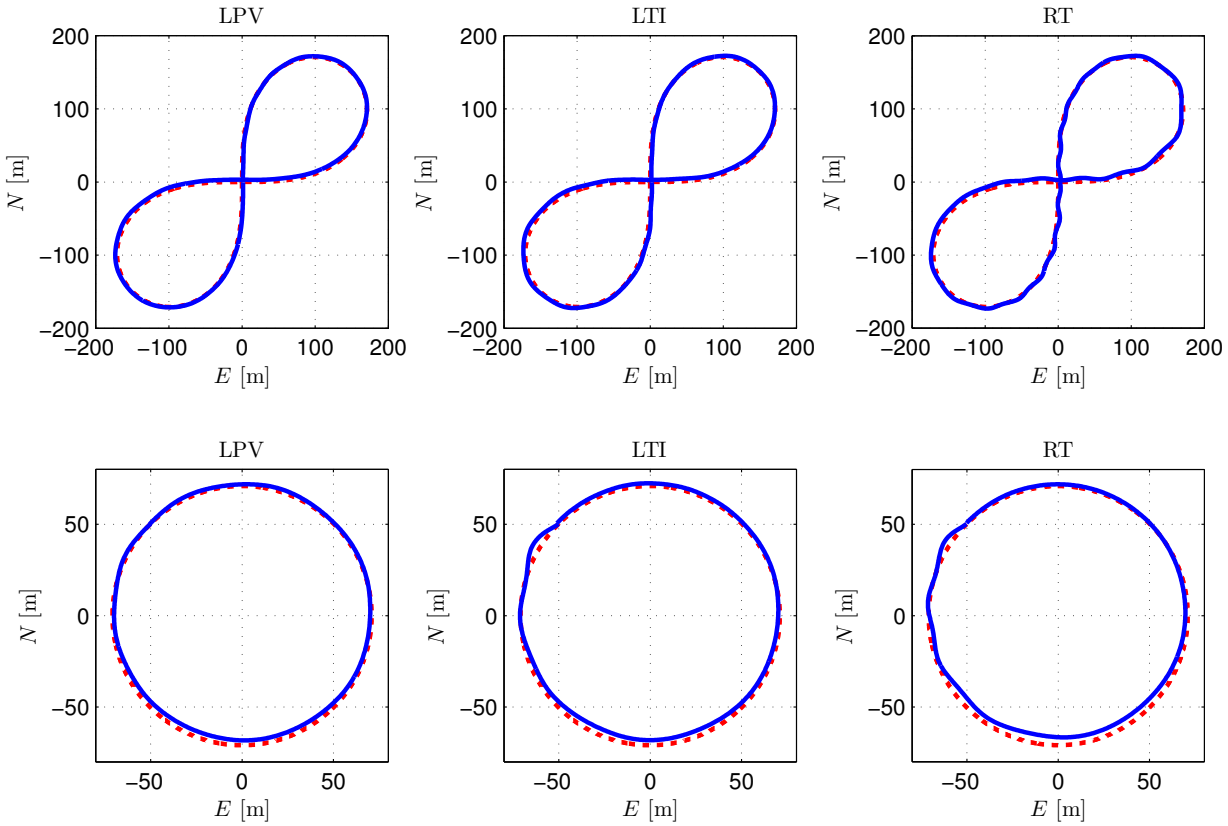

Figure 3.8: The worst case simulation runs for each controller on paths bounded as $k_{1_{max}} = 0.0141$ with a 3 m/s northerly wind.

the mean path error as

$$\text{MPE} = \frac{1}{N}\sum_{k=1}^{N} \text{dist}(\boldsymbol{P}(k), R), \tag{3.20}$$

where $\boldsymbol{P}$ is the UAS location in $\mathcal{F}_I$ as parameterized by k, and N is the total number of measurements taken in a single circuit. The UAS follows 1000 consecutive circuits for each path type and controller. An MPE value is calculated for each circuit and presented as a cumulative distribution function (CDF). The average of all 1000 MPEs is also calculated.

Table 3.2: Path-Following Performance Results

	Lemniscate			Circle			Random		
	MPE	u_r	APT	MPE	u_r	APT	MPE	u_r	APT
RT	2.56	0.245	75.84	2.41	0.156	28.95	2.87	0.253	6824
LTI	2.33	0.154	76.17	2.11	0.180	29.52	2.49	0.123	6974
LPV	1.71	0.147	72.14	1.64	0.148	28.35	1.99	0.161	6417

We define the root-mean-square control effort as

$$u_r = \sqrt{\frac{1}{N_T}\sum_{k=1}^{N_T} \bar{\boldsymbol{u}}(k)^T \bar{\boldsymbol{u}}(k)}, \tag{3.21}$$

where N_T is the total number of measurements taken over all 1000 path circuits. The control effort reflects the amount of deviation from the trim control inputs that the controller used during all circuits of a path. Finally, the average path time (APT) to complete each circuit is calculated for each controller.

3.4 Path-Following Results

A CDF of the mean path errors over 1000 circuits for the lemniscate and circular paths is presented for each of the three controllers in Fig. 3.7 for $k_{1_{max}} = 0.0141$. Additionally, a performance metric summary is given in Table 3.2.

All three controllers successfully completed all circuits for $k_{1_{max}} = 0.0141$. For the lemniscate, the RT controller had an MPE of 2.56 m with standard deviation $\sigma = 0.036$ m. The LTI controller performed slightly better with an MPE of 2.33 m and $\sigma = 0.0355$ m. While

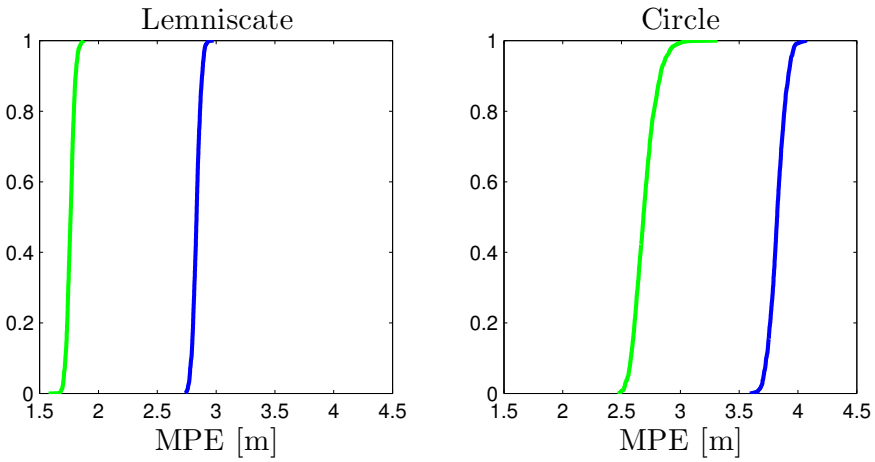

Figure 3.9: RT and LPV MPEs over 1000 loops for $|k_1| \leq 0.0250$.

the APT of the LTI controller was slightly higher than that of the RT controller, the control effort exerted by the LTI controller was a 37% improvement over the RT controller. The most consistent despite disturbances, the LPV controller completed its circuits with an MPE of 1.71 m, a 33% improvement over the RT controller, and a standard deviation of 0.022 m. u_r for the LPV and LTI controllers was comparable, but the APT for the LPV controller was the lowest of all three. Notably, both the RT and LTI controllers had difficulty maintaining an appropriate airspeed and had a tendency to slow down when disturbances were present. The LPV controller, however, was able to successfully maintain the desired airspeed in the midst of disturbances, and as a result yielded a lower APT than the LTI and RT controllers.

The circular MPE for each controller improved 4-10% over the lemniscate MPEs. The MPE for the RT controller was 2.41 m with $\sigma = 0.0454$ m. The LTI controller showed the most improvement with an MPE of 2.11 m, a 9.4% improvement over the lemniscate, and $\sigma = 0.056$ m. The relative improvement in the LTI controller is likely due to the fact that the path-

Table 3.3: Mean Path Error

	$\lvert k_1 \rvert \leq 0.0141$			$\lvert k_1 \rvert \leq 0.0250$		
	Lemniscate	Circle	Random	Lemniscate	Circle	Random
RT	2.56	2.41	2.87	2.83	3.82	2.72
LTI	2.33	2.11	2.49	-	-	-
LPV	1.71	1.64	1.99	1.76	2.70	1.64

following dynamics of the circular path are, in fact, also time-invariant. Finally, the LPV controller completed the circular paths with an MPE of 1.64 m and $\sigma = 0.0412$ m. Similar to the lemniscate, the LPV controller maintained airspeed better than the other controllers, although the difference in APT is not as pronounced with the shorter path length. Unlike the lemniscate case, the control effort of all three controllers were similar, with the LTI controller having a slightly higher control effort.

For the lemniscate and circle, the circuit that performed the worst in terms of MPE is shown for each controller in Fig 3.8. While biases are evident in the planar tracking of each controller due to the wind, the LTI and RT controllers additionally suffered from higher magnitude oscillations in the $\boldsymbol{x}_I - \boldsymbol{y}_I$ plane. Attempts to damp out these oscillations for the worst-case circuit were found to decrease the overall path-following performance for these controllers.

For the randomly generated paths, the LPV controller performed 20% better than the LTI controller and 31% better than the RT controller in terms of MPE. Similar to the lemniscate, the LTI controller had the best control effort, followed by the LPV controller.

The controllers were additionally tested with 1000 circuits on a set of paths generated using

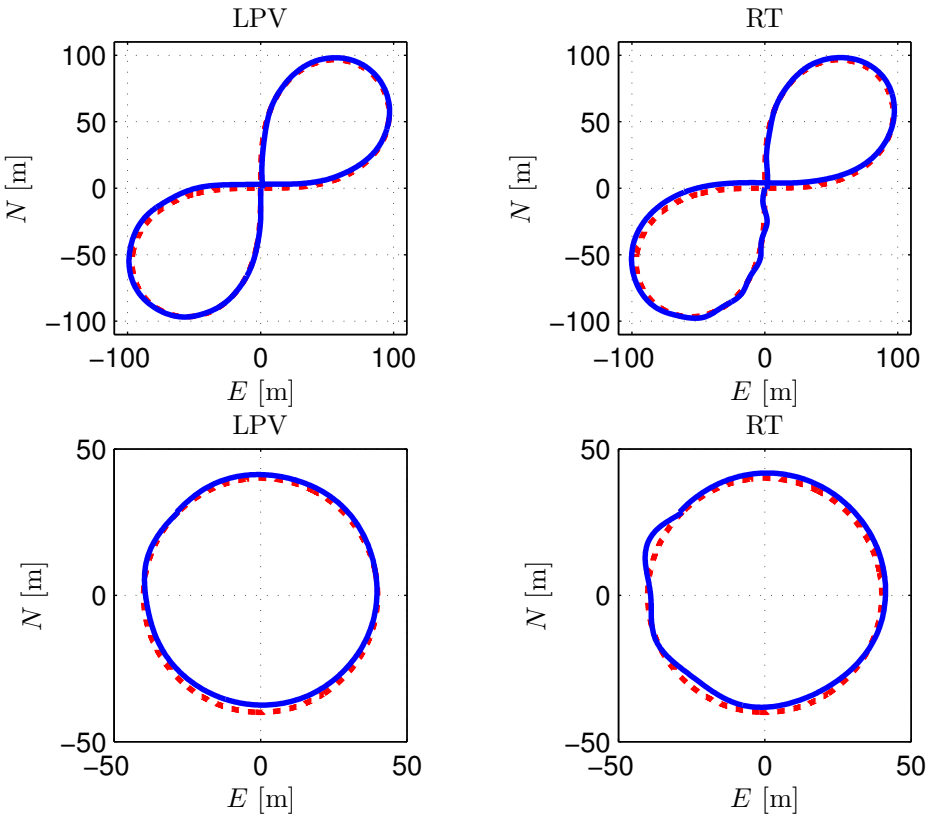

Figure 3.10: The worst case simulation runs for each controller on paths bounded as $|k_1| \leq 0.0250$ with a 3 m/s northerly wind.

$k_{1_{max}} = 0.0250$. In terms of mean path error, the RT controller performed worse on the aggressive lemniscate and circular paths compared to on those generated with $k_{1_{max}} = 0.0141$. The MPE for the aggressive lemniscate was 2.83 m, an 11% degradation from the larger lemniscate, and the MPE for the aggressive circle was 3.82, a 59% degradation. The RT controller completed the tighter turn radius random path with an MPE of 2.72 m.

The LTI controller failed to successfully follow the tighter radius paths, often saturating the rudder deflection and throttle command. The LPV controller, however, completed the lemniscate path with an MPE of 1.76, the circle with an MPE of 2.70, and the random

path with an MPE of 1.64. A CDF of the MPEs for the LPV and RT controllers is given in Fig. 3.9. While both controllers exhibited good path-following performance, the LPV controller performed better than the RT controller for the tighter set of paths in terms of MPE.

A summary of the MPEs for the six followed paths is given in Table 3.3. Additionally, the worst circuit for the RT and LPV controllers on the circle and lemniscate path satisfying $k_{1_{max}} = 0.0250$ is shown in Fig. 3.10.

Chapter 4

Integral Quadratic Constraints

This chapter presents robust stability and performance criteria for uncertain dynamic systems using IQCs. Section 4.1 presents a sufficient robust stability condition for uncertain systems. Section 4.2 extends the stability condition to include robust performance, and Section 4.3 presents the IQC multipliers that are used in this work. Finally, Section 4.4 presents a convex solution to the IQC analysis problem via the Kalman-Yakubovich-Popov (KYP) lemma.

4.1 Robust Stability using IQCs

The relevant discrete-time integral quadratic constraint based analysis theory is briefly summarized below. The interested reader can find a more detailed description of IQC theory, including continuous-time and more generalized results throughout the literature, for exam-

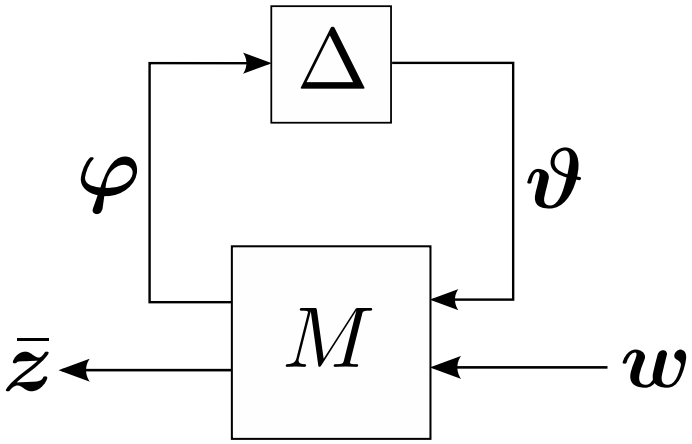

Figure 4.1: The uncertain LFR system.

ple, [18], [64], and [65].

The IQC framework was developed by Megretski and Rantzer for handling the stability and performance analysis of uncertain systems [18]. IQC analysis builds on previously developed stability principles, essentially generalizing the classical Popov multiplier approach, circle criterion, small gain, and positivity/passivity techniques [21], [66]. By transforming a set of infinitely constrained inequalities to a linear matrix inequality problem, system analysis problems can be easily solved using available computational tools [21]. However, the number of optimization variables present in the Lyapunov stability matrix grows quadratically with the order of the state-space representation, rendering high complexity problems intractable when solving IQC analysis problems via semidefinite programming [21].

Uncertain systems are modeled as an upper LFT interconnection of a stable nominal discrete-time system M and a perturbation operator Δ, as shown in Fig. 4.1. The input signal $\boldsymbol{w} \in \ell_2^{n_w}$ represents unknown finite energy signals which model noise and disturbances. The output signal $\bar{\boldsymbol{z}}$ is typically a special subset of state and control variables (or functions of

these variables) which is of particular interest. The nominal system is a causal and stable discrete-time LTI system, represented by the transfer function $M \in \mathcal{RH}_\infty$. The perturbation operator Δ is a causal, bounded operator that belongs to a set $\mathbf{\Delta}$, star-shaped with respect to the origin, and is used to capture (potentially conservatively) the effects of all possible uncertainties and nonlinearities on the nominal system.

The uncertain LFR system in Fig. 4.1 is described by the following equations:

$$\begin{bmatrix} \boldsymbol{\varphi} \\ \bar{\boldsymbol{z}} \end{bmatrix} = M \begin{bmatrix} \boldsymbol{\vartheta} \\ \boldsymbol{w} \end{bmatrix}, \ \boldsymbol{\vartheta} = \Delta(\boldsymbol{\varphi}), \ M = \left[\begin{array}{cc} M_{11} & M_{12} \\ M_{21} & M_{22} \end{array}\right]. \tag{4.1}$$

Furthermore, we can write the realization of M as

$$\left[\begin{array}{c:c} M_{11} & M_{12} \\ \hdashline M_{21} & M_{22} \end{array}\right] = \left[\begin{array}{c|c:c} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ \hdashline C_2 & D_{21} & D_{22} \end{array}\right] = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right].$$

This system is well-posed if $I - M_{11}\Delta$ has an algebraic causal inverse for all $\Delta \in \mathbf{\Delta}$. The system is robustly stable if, additionally, the inverse has a bounded ℓ_2-induced norm, i.e., for some positive scalar β, $\|(I - M_{11}\Delta)^{-1}\|_{\ell_2 \to \ell_2} < \beta$ for all $\Delta \in \mathbf{\Delta}$. In the case of a robustly stable system, all signals in Fig. 4.1 will be finite energy signals (i.e., ℓ_2 signals).

Equivalently, we can say the system is stable if it is well posed and there exists a positive

constant $\mathcal{C}$ such that

$$\sum_{k=1}^{T} \|\boldsymbol{\varphi}(k)\|_2^2 + \|\boldsymbol{\vartheta}(k)\|_2^2 + \|\bar{\boldsymbol{z}}(k)\|_2^2 \leq \mathcal{C} \sum_{k=1}^{T} \|\boldsymbol{w}(k)\|_2^2, \ \forall T \in \mathbb{Z}_+. \tag{4.2}$$

This is equivalent to saying that $\boldsymbol{x} \to 0$ as $k \to \infty$ for a given M and Δ, where k denotes the discrete time instant [67].

The signals $\boldsymbol{\varphi} \in \ell_2^{n_\varphi}$ and $\boldsymbol{\vartheta} \in \ell_2^{n_\vartheta}$ are said to satisfy the IQC defined by the so-called IQC multiplier Π, which is a self-adjoint transfer function (typically chosen in $\mathcal{RL}_\infty$), if

$$\int_{-\pi}^{\pi} \begin{bmatrix} \hat{\boldsymbol{\varphi}}(e^{j\omega}) \\ \hat{\boldsymbol{\vartheta}}(e^{j\omega}) \end{bmatrix}^* \Pi(e^{j\omega}) \begin{bmatrix} \hat{\boldsymbol{\varphi}}(e^{j\omega}) \\ \hat{\boldsymbol{\vartheta}}(e^{j\omega}) \end{bmatrix} d\omega \geq 0, \ \Pi = \begin{bmatrix} \Pi_{11} & \Pi_{12} \\ \Pi_{12}^* & \Pi_{22} \end{bmatrix}. \tag{4.3}$$

Associated with each type of uncertainty is a set of appropriate IQC multipliers, $\boldsymbol{\Pi}$, defined as the set of all self-adjoint transfer functions $\Pi \in \mathcal{RL}_\infty$ such that (4.3) holds for all $\boldsymbol{\varphi} \in \ell_2^{n_\varphi}$, $\boldsymbol{\vartheta} = \epsilon\Delta(\boldsymbol{\varphi})$, $\epsilon \in [0\ 1]$, and $\Delta \in \boldsymbol{\Delta}$. Note that the term ϵ is introduced in the preceding since, in IQC theory, the set of uncertainties is required to be star-shaped with respect to the origin such that $\epsilon\Delta \in \boldsymbol{\Delta}$ for all $\epsilon \in [0\ 1]$. Thus, any suitable multiplier $\Pi \in \boldsymbol{\Pi}$ satisfies a modified version of (4.3), where $\hat{\boldsymbol{\vartheta}}(e^{j\omega})$ is replaced with $\epsilon\hat{\boldsymbol{\vartheta}}(e^{j\omega})$, for all $\epsilon \in [0\ 1]$. We now present the main IQC stability theorem as a sufficient, but not necessary, condition for stability.

Theorem 1 (IQC Stability Theorem [18])**.** *The feedback interconnection* (4.1) *is robustly stable for all* $\Delta \in \boldsymbol{\Delta}$ *if* $\epsilon\Delta \star M$ *is well-posed for all* $\epsilon \in [0\ 1]$ *and there exists a suitable*

multiplier $\Pi \in \mathbf{\Pi}$ *such that*

$$\begin{bmatrix} M_{11}(e^{j\omega}) \\ I \end{bmatrix}^* \Pi(e^{j\omega}) \begin{bmatrix} M_{11}(e^{j\omega}) \\ I \end{bmatrix} \prec 0 \tag{4.4}$$

holds for all $\omega \in [-\pi,\ \pi]$.

Remark 1. *Given either* $\Pi_{11} \succeq 0$ *&* $\Pi_{22} \preceq 0$ *or* $\Pi_{11} \succeq 0$ *&* $\Pi_{12} = 0$*, the interconnection of* M *and* Δ *is stable for* $\epsilon\Delta$ *for all* $\epsilon \in [0\ 1]$ *if and only if it is stable for* Δ *[18].*

The multipliers chosen in our case will satisfy at least one of the two conditions in Remark 1, which will simplify the solution. Any IQC multiplier can be equivalently represented as $\Pi(z) = \Psi(z)^* S \Psi(z)$, where $\Psi(z)$ is a stable transfer function chosen in $\mathcal{RH}_\infty$, and S is real-valued and symmetric. $\Pi(z)$ can then be decomposed as:

$$\Pi(z) = \begin{bmatrix} \Psi_{11}(z) & \Psi_{12}(z) \\ \Psi_{21}(z) & \Psi_{22}(z) \end{bmatrix}^* \begin{bmatrix} S_{11} & S_{12} \\ S_{12}^T & S_{22} \end{bmatrix} \begin{bmatrix} \Psi_{11}(z) & \Psi_{12}(z) \\ \Psi_{21}(z) & \Psi_{22}(z) \end{bmatrix}, \tag{4.5}$$

where $\Psi(z)$ has the realization

$$\left[\begin{array}{c|c:c} A_\Psi & B_\Psi^1 & B_\Psi^2 \\ \hline C_\Psi & D_\Psi^1 & D_\Psi^2 \end{array}\right],$$

with A_Ψ Hurwitz. Defining

$$\left[\begin{array}{c|c} \mathcal{A} & \mathcal{B} \\ \hline \mathcal{C} & \mathcal{D} \end{array}\right] = \Psi(z) \begin{bmatrix} M_{11}(z) \\ I \end{bmatrix} = \left[\begin{array}{cc|c} A_M & 0 & B_M^1 \\ B_\Psi^1 C_M^1 & A_\Psi & B_\Psi^1 D_M^{11} + B_\Psi^2 \\ \hline D_\Psi^1 C_M^1 & C_\Psi & D_\Psi^1 D_M^{11} + D_\Psi^2 \end{array}\right], \tag{4.6}$$

(4.4) can be equivalently rewritten as

$$F(z) = \mathcal{M}(z)^* S(\gamma^2) \mathcal{M}(z) \prec 0, \tag{4.7}$$

for all $z \in \mathbb{D}$, where $\mathbb{D} = \{z | z = e^{j\omega} \text{ for all } \omega \in [-\pi,\ \pi]\}$ is the set of all points z on the complex unit circle, $S(\gamma^2) \in \mathbb{S}^{n_\mathcal{B}}$, $\Psi(z)$ is a known basis transfer function, $F(z)$ is Hermitian for all $z \in \mathbb{D}$, and the transfer matrix $\mathcal{M}(z)$ and its adjoint are given as

$$\mathcal{M}(z) = \mathcal{D} + \mathcal{C}(zI - \mathcal{A})^{-1}\mathcal{B},$$

$$\mathcal{M}(z)^* = \mathcal{D}^T + \mathcal{B}^T(I - z\mathcal{A}^T)^{-1}\mathcal{C}^T z,$$

for appropriately defined system matrices $\mathcal{A} \in \mathbb{R}^{n_\mathcal{A} \times n_\mathcal{A}}$, $\mathcal{B} \in \mathbb{R}^{n_\mathcal{A} \times n_\mathcal{B}}$, $\mathcal{C} \in \mathbb{R}^{n_\mathcal{C} \times n_\mathcal{B}}$, and $\mathcal{D} \in \mathbb{R}^{n_\mathcal{C} \times n_\mathcal{B}}$.

4.2 Robust Performance using IQCs

If we replace M_{11} by M and Π in (4.4) with $\tilde{\Pi}$, where

$$\tilde{\Pi} = \begin{bmatrix} \Pi_{11} & 0 & \Pi_{12} & 0 \\ 0 & I & 0 & 0 \\ \Pi_{12}^* & 0 & \Pi_{22} & 0 \\ 0 & 0 & 0 & -\gamma^2 I \end{bmatrix},$$

then the validity of the conditions in Theorem 1 implies that the system is robustly stable and has an ℓ_2-gain performance level γ, i.e., $\|\Delta \star M\|_{\ell_2 \to \ell_2} < \gamma$ for all $\Delta \in \mathbf{\Delta}$.

The IQC multipliers $\tilde{\Pi}$ will be re-parameterized as $\tilde{\Pi}(z) = \tilde{\Psi}(z)^* \tilde{S} \tilde{\Psi}(z)$, where

$$\tilde{S} = \begin{bmatrix} S_{11} & 0 & S_{12} & 0 \\ 0 & I & 0 & 0 \\ S_{12}^T & 0 & S_{22} & 0 \\ 0 & 0 & 0 & -\gamma^2 I \end{bmatrix}, \tilde{\Psi}(z) = \begin{bmatrix} \Psi_{11}(z) & 0 & \Psi_{12}(z) & 0 \\ 0 & I & 0 & 0 \\ \Psi_{21}(z) & 0 & \Psi_{22}(z) & 0 \\ 0 & 0 & 0 & I \end{bmatrix}.$$

The IQC performance condition can be written as

$$\tilde{F}(z) = \tilde{\mathcal{M}}(z)^* \tilde{S} \tilde{\mathcal{M}}(z) \prec 0, \tag{4.8}$$

for all $z \in \mathbb{D}$, where $\tilde{\mathcal{M}}(z) = \tilde{\mathcal{D}} + \tilde{\mathcal{C}}(zI - \tilde{\mathcal{A}})^{-1}\tilde{\mathcal{B}}$, and $\tilde{\Psi}(z)$ has the realization

$$\left[\begin{array}{c|c:c} A_{\tilde{\Psi}} & B^1_{\tilde{\Psi}} & B^2_{\tilde{\Psi}} \\ \hline C_{\tilde{\Psi}} & D^1_{\tilde{\Psi}} & D^2_{\tilde{\Psi}} \end{array}\right] = \left[\begin{array}{c|cc:cc} A_\Psi & B^1_\Psi & 0 & B^2_\Psi & 0 \\ \hline C^1_\Psi & D^{11}_\Psi & 0 & D^{12}_\Psi & 0 \\ 0 & 0 & I & 0 & 0 \\ C^2_\Psi & D^{21}_\Psi & 0 & D^{22}_\Psi & 0 \\ 0 & 0 & 0 & 0 & I \end{array}\right],$$

and $\tilde{\mathcal{A}}$, $\tilde{\mathcal{B}}$, $\tilde{\mathcal{C}}$, and $\tilde{\mathcal{D}}$ are defined as

$$\left[\begin{array}{c|c} \tilde{\mathcal{A}} & \tilde{\mathcal{B}} \\ \hline \tilde{\mathcal{C}} & \tilde{\mathcal{D}} \end{array}\right] = \tilde{\Psi}(z)\begin{bmatrix} M(z) \\ I \end{bmatrix} = \left[\begin{array}{cc|c} A_M & 0 & B_M \\ B^1_{\tilde{\Psi}}C_M & A_\Psi & B^1_{\tilde{\Psi}}D_M + B^2_{\tilde{\Psi}} \\ \hline D^1_{\tilde{\Psi}}C_M & C_{\tilde{\Psi}} & D^1_{\tilde{\Psi}}D_M + D^2_{\tilde{\Psi}} \end{array}\right].$$

When referencing the IQC inequality for robust performance, the tildes are dropped and the notation in (4.7) is used for convenience. The augmentation of the IQC multipliers and the addition of the disturbance input and performance outputs are implied when the ℓ_2-gain performance level, γ, is discussed.

4.3 IQC Multipliers

We now present the discrete-time IQC multipliers associated with the three uncertainty types utilized in this work; dynamic linear time-invariant (DLTI) uncertainties, static linear

time-invariant uncertainties (SLTI), and static linear time-varying uncertainties (SLTV).

The multipliers used for dynamic linear time-invariant uncertainties bounded as $\|\Delta\|_\infty \leq \beta$ are of the form $\Pi_1(z) = \Psi_1(z)^* S_1 \Psi_1(z)$, and those for contractive static linear time-invariant uncertainties bounded as $|\delta| \leq 1$ are of the form $\Pi_2(z) = \Psi_2(z)^* S_2 \Psi_2(z)$ [18]. Multipliers for static linear time-varying uncertainties bounded as $|\delta(k)| \leq 1$ with variation rates $\nu(k) = \delta(k+1) - \delta(k)$ bounded as $\alpha^- \leq \nu(k) \leq \alpha^+$ are of the form $\Pi_3(z) = \Psi_3(z)^* S_3 \Psi_3(z)$ [65]. The frequency dependent terms are

$$\Psi_1(z) = \begin{bmatrix} I_{n_\varphi} & 0 \\ 0 & I_{n_\vartheta} \end{bmatrix}, \ \Psi_2(z) = \begin{bmatrix} H(z) & 0 \\ 0 & H(z) \end{bmatrix}, \ \Psi_3(z) = \begin{bmatrix} H_2(z) & 0 \\ 0 & H_3(z) \end{bmatrix},$$

where $H(z)$ is a stable basis transfer function with basis length d. In this work, the basis transfer functions are defined by the realizations

$$H(z) = \left[1, (z+\lambda)^{-1}, \cdots, (z+\lambda)^{-d}\right]^T \otimes I_r = \left[\begin{array}{c|c} A_H & B_H \\ \hline C_H & D_H \end{array}\right],$$

$$H_1 = \left[\begin{array}{c|c} A_H & B_H \\ \hline I & 0 \end{array}\right], H_2 = \begin{bmatrix} H \\ H_1 \end{bmatrix}, H_3 = \left[\begin{array}{c|cc} A_H & B_H & I \\ \hline C_H & D_H & 0 \\ 0 & 0 & I \end{array}\right],$$

where λ is any pole in the open unit disc and r is the number of uncertainty repetitions in

the corresponding LFR. The frequency independent terms are given as

$$S_1 = \begin{bmatrix} \beta^2 X \otimes I_{n_\varphi} & 0 \\ 0 & -X \otimes I_{n_\vartheta} \end{bmatrix}, \; S_2 = \begin{bmatrix} X & Y \\ Y^T & -X \end{bmatrix},$$
$$S_{3_{ij}} = \begin{bmatrix} S_{4_{ij}} & 0 \\ 0 & S_{2_{ij}} \end{bmatrix}, S_4 = \begin{bmatrix} -\alpha^- \alpha^+ X & \frac{\alpha^- + \alpha^+}{2} X + Y \\ \frac{\alpha^- + \alpha^+}{2} X^T + Y^T & -X \end{bmatrix},$$

where $0 \succ X \in \mathbb{S}^q$, $-Y^T = Y \in \mathbb{R}^{q \times q}$, $q = dr$, and $i, j = 1,\ 2$. To account for the bounded variation rates in linear time-varying uncertainties, we analyze an extended uncertainty block and extended system by adding q columns of zeros to M as

$$\Delta_e = \begin{bmatrix} \delta I_r & \\ & \nu I_q H_1(z) \end{bmatrix}, M_e = \left[\begin{array}{c|cc} A_M & B_M^1 \; 0_{n \times q} & B_M^2 \\ \hline C_M & D_M^1 \; 0_{n_z \times q} & D_M^2 \end{array}\right], \tag{4.9}$$

where n is the number of states in the realization of M, $B_M = [B_M^1 \, B_M^2]$, $B_M^1 \in \mathbb{R}^{n \times n_\vartheta}$, $B_M^2 \in \mathbb{R}^{n \times n_w}$, and $D_M = [D_M^1 \, D_M^2]$ is defined with like dimensions.

4.4 The Kalman-Yakubovich-Popov Lemma

The frequency-dependent infinite-dimensional LMI (4.4) can be reformulated as a frequency independent finite-dimensional LMI problem through application of the celebrated Kalman-Yakubovich-Popov (KYP) Lemma, presented below.

Lemma 1 (Discrete-Time KYP Lemma)**.** *Given matrices $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, and*

$S \in \mathbb{S}^{n+m}$ where A has no eigenvalues on the unit circle and (A, B) is controllable, the following statements are equivalent:

i) The following holds for all frequencies $\omega \in [-\pi, \pi]$:

$$\begin{bmatrix} (e^{j\omega}I - A)^{-1}B \\ I \end{bmatrix}^* S \begin{bmatrix} (e^{j\omega}I - A)^{-1}B \\ I \end{bmatrix} \preceq 0. \tag{4.10}$$

ii) There exists a matrix $P = P^T \in \mathbb{R}^{n\times n}$ such that

$$\begin{bmatrix} A^TPA - P & A^TPB \\ B^TPA & B^TPB \end{bmatrix} + S \preceq 0. \tag{4.11}$$

Proof. The proof can be found in [68]. □

Remark 2. *The lemma also holds for strict inequalities without the requirement that (A, B) be controllable.*

The IQC inequality (4.4) can be equivalently written as

$$F(e^{j\omega}) = \begin{bmatrix} (e^{j\omega}I - \mathcal{A})^{-1}\mathcal{B} \\ I \end{bmatrix}^* \begin{bmatrix} \mathcal{Q} & \mathcal{F} \\ \mathcal{F}^T & \mathcal{R} \end{bmatrix} \begin{bmatrix} (e^{j\omega}I - \mathcal{A})^{-1}\mathcal{B} \\ I \end{bmatrix} \prec 0, \tag{4.12}$$

for all $\omega \in [-\pi,\ \pi]$. By application of the KYP lemma, if $\Pi(z) \in \mathcal{RL}_\infty$, we can rewrite the

stability condition (4.12) as the existence of a matrix $P \in \mathbb{S}^{n_\mathcal{A}}$ that satisfies the LMI

$$\begin{bmatrix} \mathcal{A}^T P \mathcal{A} - P & \mathcal{A}^T P \mathcal{B} \\ \mathcal{B}^T P \mathcal{A} & \mathcal{B}^T P \mathcal{B} \end{bmatrix} + \begin{bmatrix} \mathcal{Q} & \mathcal{F} \\ \mathcal{F}^T & \mathcal{R} \end{bmatrix} \prec 0, \tag{4.13}$$

where

$$\begin{bmatrix} \mathcal{C}^T \\ \mathcal{D}^T \end{bmatrix} S \begin{bmatrix} \mathcal{C} & \mathcal{D} \end{bmatrix} = \begin{bmatrix} \mathcal{C}^T S \mathcal{C} & \mathcal{C}^T S \mathcal{D} \\ \mathcal{D}^T S \mathcal{C} & \mathcal{D}^T S \mathcal{D} \end{bmatrix} = \begin{bmatrix} \mathcal{Q} & \mathcal{F} \\ \mathcal{F}^T & \mathcal{R} \end{bmatrix}. \tag{4.14}$$

The inclusion of the Lyapunov matrix P cancels out all frequency terms in the inequality, but adds $(n_\mathcal{A}^2 + n_\mathcal{A})/2$ decision variables to the convex problem. (4.13) thus represents a frequency-independent sufficient condition for robust stability that can be solved using semidefinite programming.

Similarly, the IQC robust performance problem has convex analysis conditions after the application of the KYP lemma. Specifically, robust stability and the minimum achievable robust performance level γ are determined by solving the following semidefinite program:

$$\begin{aligned} &\text{minimize}: \ \gamma^2 \\ &\text{subject to}: \ P \in \mathbb{S}^{n_\mathcal{A}}, \\ &\begin{bmatrix} \mathcal{A}^T P \mathcal{A} - P & \mathcal{A}^T P \mathcal{B} \\ \mathcal{B}^T P \mathcal{A} & \mathcal{B}^T P \mathcal{B} \end{bmatrix} + \begin{bmatrix} \mathcal{Q} & \mathcal{F} \\ \mathcal{F}^T & \mathcal{R} \end{bmatrix} \prec 0, \end{aligned} \tag{4.15}$$

where $\mathcal{Q}$, $\mathcal{F}$, and $\mathcal{R}$ are functions of γ.

Chapter 5

An IQC Analysis Framework for Small Fixed-Wing UAS

The following uncertainty framework, shown in Fig. 5.1, consists of interconnected uncertainties that have been judiciously picked to be as thorough as possible at covering expected uncertainties and nonlinearities while reducing the conservatism of the resulting analysis problem by leveraging different IQC multipliers. As shown in Fig. 5.2, the framework presented in this work strives to produce computationally manageable analysis problems that can be solved on a desktop computer while also resulting in meaningful analysis that is not overly conservative. Due to the modular nature of IQC-based analysis and LFRs, any uncertainties shown in Fig. 5.1 (highlighted red) can be easily modified or omitted prior to analysis. The three investigated uncertainty groups and the associated quantification methods utilized for the example analysis are described in detail in the proceeding. Specifically,

Section 5.1 presents a methodology for formal validation of UAS fight controllers, Section 5.2 presents the set of parametric uncertainties in the linear dynamic aircraft model as well as a model reduction technique for large, open-loop unstable LFRs with highly coupled static LTI uncertainties, Section 5.3 presents the representation and quantification of uncertainties capturing unmodeled dynamics and nonlinearities in the UAS aerodynamic model, Section 5.4 presents the representation and quantification of uncertainties corresponding to unmodeled actuator and thrust dynamics, saturation, and time-delays. Additionally a method for representing partial time-step time delays in discrete-time using static LTV uncertainties is presented. Finally, Section 5.5 presents analysis results using the uncertainty framework and Telemaster UAS model with LTI $\mathcal{H}_\infty$ controllers for tracking a steady trim trajectory through three examples. In the first example, the effect of the three uncertainty groups on a single controller are presented. In the second example, IQC analysis is performed to compare the robustness of three different $\mathcal{H}_\infty$ controllers. Lastly, the IQC analysis framework is used to tune an $\mathcal{H}_\infty$ controller and make it more robust to uncertainties and nonlinearities.

5.1 Algorithmic Level Certification for Control Systems

Formal validation of UAS control systems is a multi-step process in which IQC analysis can play an essential role. In order to ensure the reliability of the control system under investigation, the controller must be shown to be robust to external disturbances, unmodeled

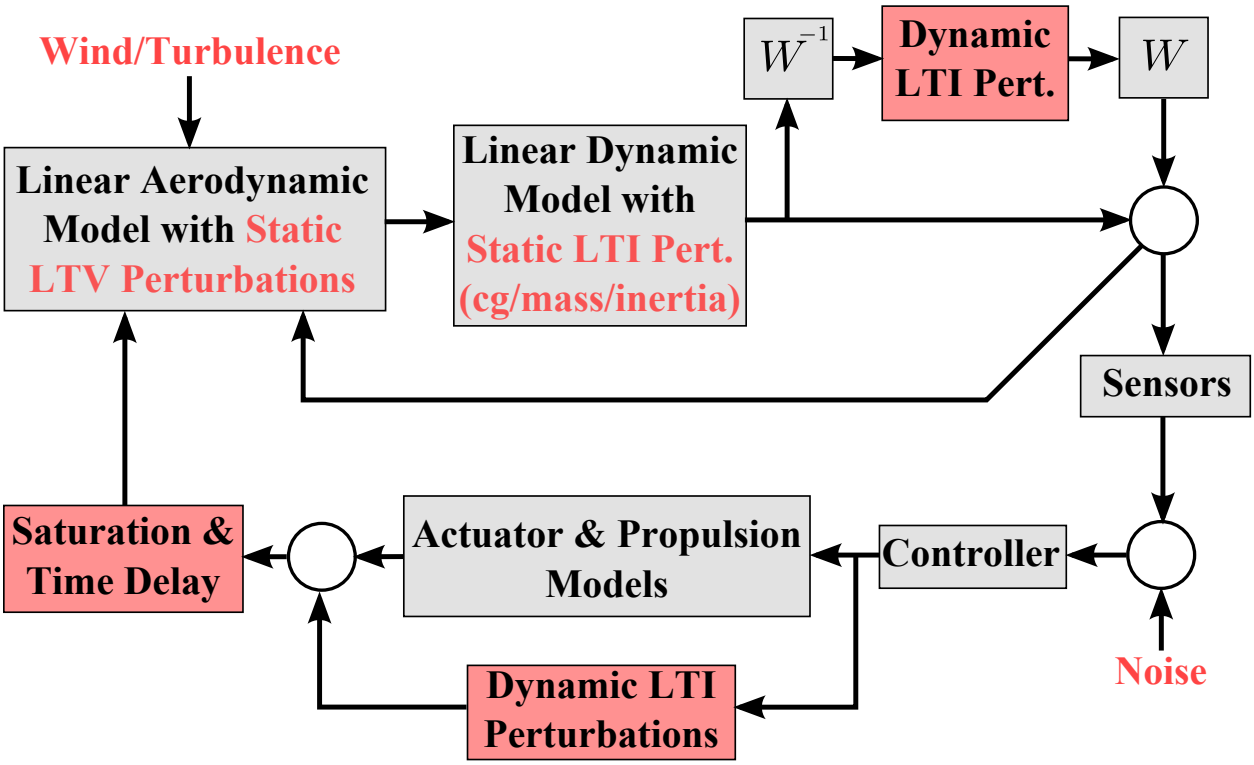

Figure 5.1: The fixed-wing UAS uncertainty framework.

dynamics, uncertainties, and nonlinearities. Fig. 5.3 shows the proposed framework for algorithmic level validation of control systems, which will be discussed next.

The UAS is first decomposed into two parts: a known reasonably accurate nonlinear model and an unknown set of uncertainties. The reasonably accurate nonlinear model is given by the differential equations (3.15) in addition to the servomotor and nonlinear thrust models. Terms such as unmodeled aerodynamics and time-delays are accounted for in the set of uncertainties related to the physical system.

The nonlinear model is next decomposed into a simplified plant model and a second set of uncertainties. The simplified plant model (2.18) is formed by linearizing and subsequently

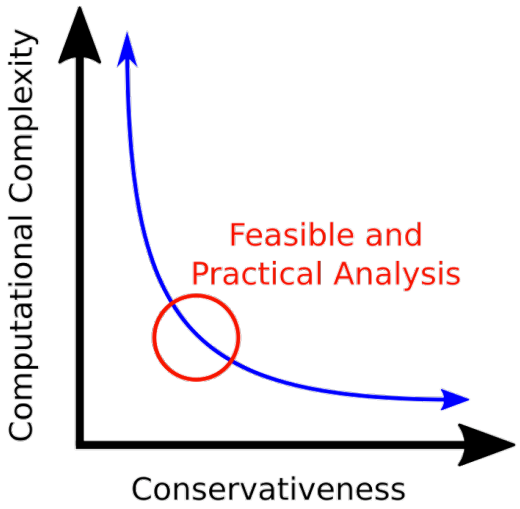

Figure 5.2: Balancing computational complexity with conservativeness.

discretizing (3.15) (with the equation for the nominal CG position replaced by (2.11)), along with the actuator dynamics. Nonlinearities ignored during the linearization process, such as saturation and nonlinear dynamics, are captured by the second set of uncertainties.

The discrete-time controller (2.5.1) is synthesized based on the simplified plant. The plant and controller combined form the nominal system M in (4.1). The set of uncertainties in the physical system and uncertainties ignored during the formation of the simplified plant are combined to form the uncertainty block, Δ, in (4.1), assuming rational dependence of the uncertain system equations on the uncertainties. The LFR $\Delta \star M$ is thus a representation of the original UAS.

The controller is next validated with respect to the chosen set of uncertainties and performance output of interest using the rigorous IQC-based analysis approach. If performance requirements are not achieved, information obtained from the resulting analysis can be used to synthesize a new controller, or even determine a better simplified plant model. This

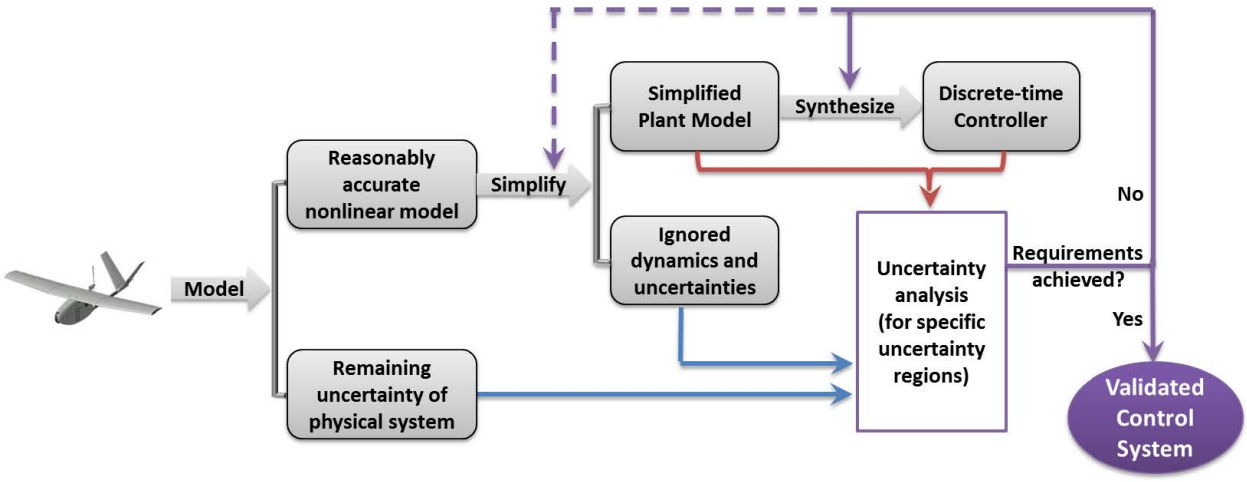

Figure 5.3: The path to a validated control system.

process can be repeated until stability and performance requirements have been achieved, at which point the control system is considered formally validated with respect to the considered types and regions of uncertainties and exogenous disturbances. The uncertainty framework discussed in this work is designed such that the uncertainty analysis portion of the control validation process is computationally manageable. Short analysis times will allow for an efficient analysis-in-the-loop controller synthesis process, resulting in controllers with guaranteed performance bounds for the modeled uncertainties and nonlinearities.

5.2 Linear Dynamic Model

The linear dynamic system equations are obtained from linearizing and discritizing (2.10), where the system inputs and outputs are $[\bar{\boldsymbol{M}}^T, \bar{\boldsymbol{F}}^T]^T$ and $\bar{\boldsymbol{x}}$, respectively. The dynamic model contains six static linear time-invariant uncertainties corresponding to uncertain mass,

Table 5.1: Parametric Uncertainties

δ	m	$+x$	z	I_x	I_y	I_z
$\sup \lvert\delta\rvert$	0.1 m	0.03 m	0.03 m	0.15 I_x	0.15 I_y	0.15 I_z

CG location, and moments of inertia, represented by $\boldsymbol{\delta}_p = [\delta_m, \delta_x, \delta_z, \delta_{I_x}, \delta_{I_y}, \delta_{I_z}]$. These parametric uncertainties cover potential errors in the initial quantification of the aircraft's physical parameters in addition to slight modifications to the airframe. The addition of a camera mounted under the nose, for example, would simultaneously affect the overall mass, CG location, and moments of inertia. If the onboard controller was deemed to be robust to perturbations in physical parameters, small airframe modifications would not necessitate a revised model and re-synthesized controller.

While the uncertain CG terms are already present in (2.10), the other parametric terms are included as additive uncertainties simply by replacing m and J by $m + \delta_m$ and $J + \text{diag}(\delta_{I_x}, \delta_{I_y}, \delta_{I_z})$, respectively. The chosen parametric uncertainty bounds can be found in Table 5.1. Due to the symmetry assumption of the Telemaster airframe, the I_{xz}, $\delta_{I_{xz}}$, and δ_y terms have been omitted, but could easily be incorporated. Since the CG of the Telemaster airframe is nominally located at the rear of the recommended CG range, the δ_x term is not centered and allows for variation towards the nose of the aircraft.

Due to the large amount of coupling induced by the uncertain CG terms in (2.10), the LFR corresponding to the linear dynamic model is very large and the corresponding IQC analysis problem was found to be computationally intractable. A common technique to reduce the model size of uncertain linear systems is through the application of balanced truncation based

methods [69], [70]. These methods can be conveniently formulated as convex optimization problems that allow the dimension of an LFR's uncertainty block to be directly reduced while simultaneously calculating guaranteed upper bounds on the worst-case model reduction error, which can be reincorporated into the uncertain system as uncertainties. The technique used herein, however, is a bit ad hoc, and was performed in order to reduce the complexity of the system LFR such that the resulting analysis problem was computationally tractable and could be solved. A more rigorous technique combining minimal realizations of LFRs and coprime factors reduction of the unstable linear dynamic block is being investigated [71].

The very large size of the LFR resulting from the parametric uncertainties was found to be prohibitive to applying balanced truncation based methods due to computational limitations. An alternative strategy has been employed here in order to instead reduce the complexity of the state-space matrix-valued function's rational dependence on the parametric uncertainties, resulting in a reduced uncertainty block size when the LFR is reformed. All uncertainties are static time-invariant, so the uncertainty space can be sampled by closing the LFR over applicable uncertainty values. This results in a nominal LTI system and allows the model reduction error to be easily assessed with a standard $\mathcal{H}_\infty$ norm.

5.2.1 Forming a System with Polynomial Dependence

Consider the LFR $\Delta_p \star M_p$, with $\Delta_p = \text{diag}(\frac{1}{z} I_{n_x}, \delta_m I_{n_{\delta m}}, \delta_x I_{n_{\delta x}}, \ldots, \delta_{I_z} I_{n_{\delta_{I_z}}})$, where $n_x = 36$ is the total number of aircraft, actuator, and controller states. Let the set of parametric

uncertainties be $\boldsymbol{\delta}_p = (\delta_m, \delta_x, \ldots, \delta_{I_z})$. The system can be equivalently expressed as $\Delta_p \star M_p = D + C\Delta_p(I - A\Delta)^{-1}B = \mathfrak{D}(\boldsymbol{\delta}_p) + \mathfrak{C}(\boldsymbol{\delta}_p)(zI - \mathfrak{A}(\boldsymbol{\delta}_p))^{-1}\mathfrak{B}(\boldsymbol{\delta}_p)$, where $\mathfrak{A}(\boldsymbol{\delta}_p)$, $\mathfrak{B}(\boldsymbol{\delta}_p)$, $\mathfrak{C}(\boldsymbol{\delta}_p)$, and $\mathfrak{D}(\boldsymbol{\delta}_p)$ are rational functions of $\boldsymbol{\delta}_p$. Our goal is to reduce the complexity of the rational dependence of $\mathfrak{A}$, $\mathfrak{B}$, $\mathfrak{C}$, and $\mathfrak{D}$ on $\boldsymbol{\delta}_p$. This is accomplished by restricting the allowable rational combinations that can be formed out of $\boldsymbol{\delta}_p$ for all four parameter-dependent matrices. The concatenated parameter-dependent matrices can be equivalently rewritten as

$$\begin{bmatrix} \mathfrak{A}(\boldsymbol{\delta}_p) & \mathfrak{B}(\boldsymbol{\delta}_p) \\ \mathfrak{C}(\boldsymbol{\delta}_p) & \mathfrak{D}(\boldsymbol{\delta}_p) \end{bmatrix} = M_0 + M_1 p_1 + M_2 p_2 + \cdots + M_q p_q,$$

where p_i represents a unique polynomial combination of parameters (with maximum degree d) from the set $\boldsymbol{p} = \{\delta_m, \delta_x, \ldots, \delta_{I_z}, 1/\delta_m, 1/\delta_x, \ldots, 1/\delta_{I_z}\}$ and $M_i \in \mathbb{R}^{(n_x+n_z)\times(n_x+n_w)}$, for $i = 0, 1, \ldots, q$. Note that polynomial combinations of uncertainties and their inverses can result in not only polynomial, but any rational combination of uncertainties.

5.2.2 Reducing the Polynomial Order

An inner-outer loop stepwise regression [72] is then employed to reduce the number and complexity of polynomial terms p_i to be used in the reduced model. The outer loop adds and removes polynomial terms, aiming to reduce q while keeping the reduction error, $\max_j \|E_j\|_\infty$ (defined below), under a pre-specified tolerance at all points j sampled from the uncertainty space in a fine grid. The reduced LFR system $\Delta_r \star M_r$ corresponds to the reduced polynomial matrices $M_{r_0} + M_{r_1} p_{r_1} + M_{r_2} p_{r_2} + \cdots + M_{r_s} p_{r_s}$, where the matrices M_{r_i}, for $i = 0, 1, \ldots, s$,

are determined in an inner stepwise regression loop designed to keep the matrices sparse so that the uncertainty block corresponding to the reformed LFR for a chosen set of polynomials, p_{r_i}, remains small. Sampling $\Delta_p \star M_p$ and $\Delta_r \star M_r$ at a point j yields a set of nominal LTI systems given by the state-space representations $(A_{M_{p_j}}, B_{M_{p_j}}, C_{M_{p_j}}, D_{M_{p_j}})$ and $(A_{M_{r_j}}, B_{M_{r_j}}, C_{M_{r_j}}, D_{M_{r_j}})$, respectively. The error system can now be defined as

$$E_j = \left[\begin{array}{cc|c} A_{M_{p_j}} & 0 & B_{M_{p_j}} \\ 0 & A_{M_{r_j}} & B_{M_{r_j}} \\ \hline C_{M_{p_j}} & -C_{M_{r_j}} & 0 \end{array}\right]. \tag{5.1}$$

5.2.3 Incorporating Model Reduction Error

The model reduction technique was applied for the performance output $\bar{\boldsymbol{z}} = \bar{\boldsymbol{P}}_0$. The parametric uncertainty block was reduced from

$$\Delta_p = \mathrm{diag}(\frac{1}{z}I_{36}, \delta_m I_{539}, \delta_x I_{702}, \delta_z I_{706}, \delta_{I_x} I_{222}, \delta_{I_y} I_{107}, \delta_{I_z} I_{222})$$

to

$$\Delta_r = \mathrm{diag}(\frac{1}{z}I_{36}, \delta_m, \delta_x I_6, \delta_z I_6, \delta_{I_x}, \delta_{I_y}, \delta_{I_z}),$$

allowing the parametric uncertainties to be coupled with the other uncertainties in Fig. 5.1 to result in a computationally manageable analysis problem. Additionally, a dynamic LTI uncertainty, $\Delta_E(z)$, is included in the linear dynamic group as an additive uncertainty to

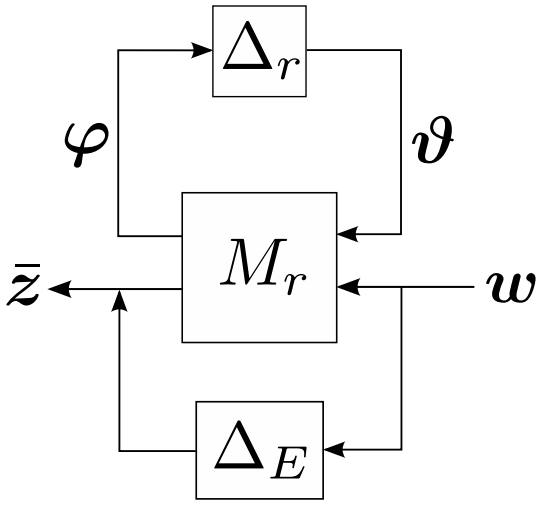

Figure 5.4: Model reduction error is incorporated into the reduced system as the dynamic LTI uncertainty Δ_E.

account for the model reduction error with dimensions corresponding to $\Delta_r \star M_r$ and bounds given by the maximum model reduction error for the system under analysis. Final analysis for the linear dynamic group is conducted on the LFR formulated as $\Delta_r \star M_r + \Delta_E(z)$, as shown in Fig. 5.4.

Finally, an additive dynamic LTI uncertainty $\Delta_N(z) \in \mathcal{RH}_\infty^{12\times 12}$ represents the effect of nonlinearities in the dynamic model ignored during the linearization process. A weighting matrix $W = \text{diag}(I_3, 3, I_2, 0.55, 0.35, 0.2, 5, 10I_2)$ was used to normalize the magnitude of the state inputs to the uncertainty, as shown in Fig. 5.1. This perturbation satisfies $\|\Delta_N\|_\infty \leq 0.01$. The weighting matrix and dynamic uncertainty are chosen by comparing closed-loop linear and nonlinear simulations. The weighting matrix reflects the relative magnitudes of the state vector in simulation. Absolute magnitudes could alternatively be used, although the resulting scalar multiplying W would be identically canceled out by its inverse in the

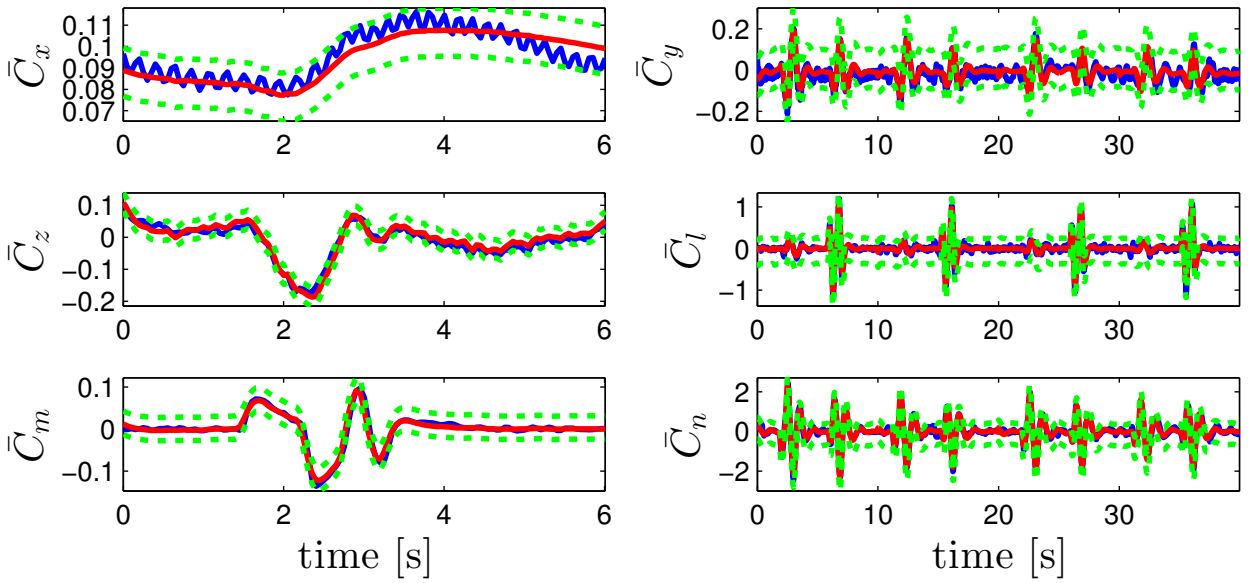

Figure 5.5: Coefficient histories obtained from the linearized aerodynamic model (red) are compared with those from accelerometer data (blue) in a validation flight test to obtain uncertainty magnitude bounds (green).

block diagram.

The linear dynamic model uncertainties (dynamic group) thus consist of 6 static LTI uncertainties, δ_m, δ_x, δ_z, δ_{I_x}, δ_{I_y}, δ_{I_z}, and 2 dynamic LTI uncertainties, $\Delta_E(z)$ and $\Delta_N(z)$.

5.3 Aerodynamic Model

Nonlinearities and unmodeled aerodynamics are represented by static linear time-varying perturbations in the uncertain system. Flight test data collected for the purposes of system identification is leveraged in order to quantify uncertainty magnitude and rate bounds to further reduce conservatism (compared to allowing arbitrarily fast variations).

Table 5.2: Aerodynamic Uncertainty Bounds $\times 100$

Term	Value	Term	Value	Term	Value	Term	Value	Term	Value	Term	Value
$\delta^+_{C_x}$	0.7223	$\delta^+_{C_y}$	1.0521	$\delta^+_{C_z}$	3.8267	$\delta^+_{C_l}$	0.5435	$\delta^+_{C_m}$	1.8174	$\delta^+_{C_n}$	0.2019
$\delta^-_{C_x}$	-0.8804	$\delta^-_{C_y}$	-1.1315	$\delta^-_{C_z}$	-3.7813	$\delta^-_{C_l}$	-0.5022	$\delta^-_{C_m}$	-1.6366	$\delta^-_{C_n}$	-0.2191
$\nu^+_{C_x}$	1.0226	$\nu^+_{C_y}$	1.2593	$\nu^+_{C_z}$	5.7708	$\nu^+_{C_l}$	0.5878	$\nu^+_{C_m}$	2.1982	$\nu^+_{C_n}$	0.2125
$\nu^-_{C_x}$	-0.9579	$\nu^-_{C_y}$	-1.5455	$\nu^-_{C_z}$	-5.0018	$\nu^-_{C_l}$	-0.4988	$\nu^-_{C_m}$	-2.5448	$\nu^-_{C_n}$	-0.3477

Recall that the aerodynamic forces and moments in (2.10) are defined in terms of aerodynamic coefficients, namely,

$$\begin{aligned} F_i(\cdot) &= \frac{1}{2} C_i(\cdot) \rho V_a^2 S, \text{ for } i = x, y, z, \\ M_j(\cdot) &= \frac{1}{2} C_j(\cdot) \rho V_a^2 S b, \text{ for } j = l, n, \\ M_m(\cdot) &= \frac{1}{2} C_m(\cdot) \rho V_a^2 S \bar{c}. \end{aligned} \tag{5.2}$$

The output error method was used to solve for the aerodynamic parameter values that make up the nonlinear aerodynamic coefficients (based on the chosen aerodynamic model structure) by comparing measured force and moment time-histories to a postulated aerodynamic model [49]-[52].

Instead of adding uncertainties to each aerodynamic sub-coefficient, six additive static time-varying magnitude and rate bounded uncertainties $\delta_{C_i}(k)$, for $i = x, y, z, l, m, n$, are used to characterize the aerodynamic uncertainties, covering sub-coefficient estimation errors, nonlinearities in the sub-coefficients, and unmodeled aerodynamics, hence resulting in a computationally manageable and thorough uncertainty representation.

Using flight test data collected to validate the aerodynamic model, shown in Fig. 5.5, the

measured coefficient histories are compared with a set of simulated coefficient histories. That is, using the same set of inputs, coefficient histories are calculated using the *linearized* aerodynamic model and compared with those obtained from the physical system in flight. Errors between the simulated and measured coefficients are calculated at each time-step. The resulting magnitude error distribution had large outliers, and the resulting bounds led to unrealistic worst-case aerodynamics. To mitigate this, the magnitude bounds on $\delta_{C_i}(k)$ were reduced using ellipsoidal peeling in order to get representative bounds that were not overly conservative [73].

Similarly, the rate bounds $\nu_{C_i}(k)$ given in Table 5.2 were calculated as the maximum derivative of the magnitude error. To incorporate the uncertainties, the aerodynamic coefficient terms $C_i(\cdot)$ in (5.2) are simply replaced by $C_i(\cdot) + \delta_{C_i}(k)$ for $i = x, y, z, l, m, n$. The uncertainty block corresponding to each aerodynamic uncertainty in the resulting LFR has a dimension of 6 since the aerodynamic uncertainties multiply V_a, a function of u, v, w, u_w, v_w, and w_w. Taking a minimal realization of the aerodynamic model LFR results in uncertainty blocks of dimension 1 for each uncertainty.

As an example of the nonlinear effect of coupling uncertainties, consider the performance output $\bar{\boldsymbol{z}} = \bar{\boldsymbol{P}}_0$ with a nominal worst-case performance value of $\|\boldsymbol{w} \mapsto \bar{\boldsymbol{z}}\|_{\ell_2 \mapsto \ell_2} < 5.93$. While the upper bound on the worst-case performance of the LFR with 6 aerodynamic uncertainties is 8.24, the upper bounds resulting from only considering $\delta_{C_x}(k)$ or $\delta_{C_z}(k)$ are 6.26 and 6.42, respectively. However, if both $\delta_{C_x}(k)$ and $\delta_{C_z}(k)$ are considered together, the resulting upper bound is 6.93, higher than any of the other individual aerodynamic

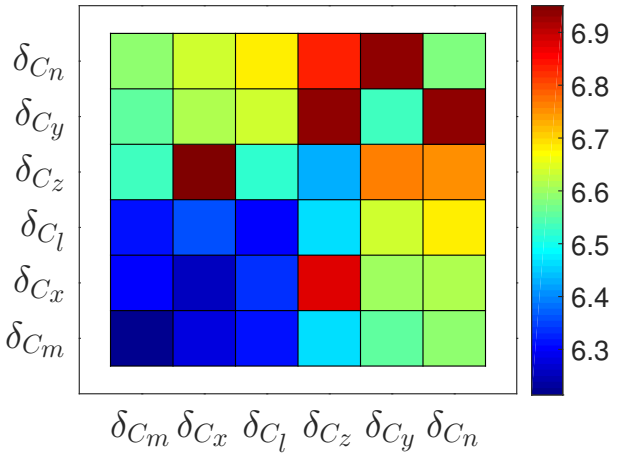

Figure 5.6: Upper bounds on $\|\boldsymbol{w} \mapsto \bar{\boldsymbol{z}}\|_{\ell_2 \mapsto \ell_2}$ considering two coupled aerodynamic uncertainties are given in the upper-left, while lower bounds are given in the lower-right.

uncertainties. Fig. 5.6 presents upper and lower bounds on $\|\boldsymbol{w} \mapsto \bar{\boldsymbol{z}}\|_{\ell_2 \mapsto \ell_2}$ for all pairs of aerodynamic uncertainties. From the figure, it is evident that $\delta_{C_z}(k)$ when coupled with $\delta_{C_x}(k)$, $\delta_{C_y}(k)$, or $\delta_{C_n}(k)$ results in a higher performance bound due to coupling. $\delta_{C_y}(k)$ coupled with $\delta_{C_n}(k)$ produces a similar effect. Coupling exercises such as this can be easily performed using any combination of uncertainties from Fig. 5.1, and help highlight the most influential uncertainties in the system ($\delta_{C_z}(k)$ and $\delta_{C_n}(k)$, in this example) as well as those that may be ignored in the analysis process to reduce computational complexity. For instance, $\delta_{C_m}(k)$ is the least influential uncertainty, and shows no major coupling effects in Fig. 5.6.

The aerodynamic uncertainties (aero group) consist of 6 static LTV uncertainties, $\delta_{C_x}(k)$, $\delta_{C_y}(k)$, $\delta_{C_z}(k)$, $\delta_{C_l}(k)$, $\delta_{C_m}(k)$, and $\delta_{C_n}(k)$.

5.4 Control Input Uncertainties and Delays

Given centered and normalized inputs, the limits on actuator deflections and thrust output can be modeled by the unit saturation operator. Fig. 5.1 depicts the actuator deflections and output thrust passing through a block diagonal saturation operator, which can be appropriately represented by the so-called Zames-Falb IQC multiplier [18], [74], [75]. Specifically, this multiplier can represent saturations as odd-monotonic and slope-restricted nonlinearities constrained by sector bounds in the range [0, 1]. While this multiplier representation appears to be an apt description of the nonlinearity, the Zames-Falb constraints fail to represent the actual unit saturation point, and as such, end up being very conservative. Additionally, as pointed out by [24], a Zames-Falb representation of saturation is only applicable to open-loop stable aircraft plants. Since 6 DOF aircraft models are not typically open-loop stable, the Zames-Falb multiplier has not been used to represent actuator and thrust saturation in this work.

Inspired by [76], saturation is instead modeled as a static time-varying uncertainty, incorporated as $\text{sat}(u) \approx u - u\delta_{\sigma u}(k)$, where the uncertainty representing saturation is bounded as $\delta_{\sigma(\cdot)}(k) \in [0,\ \sigma_{max}]$, and σ_{max} is the saturation tolerance that the system will be analyzed against (10% for the Telemaster). While still clearly conservative, this uncertainty representation can lead to some insight on the sensitivity and relative effect of saturation on stability and performance.

Several multipliers have been developed in the literature by [77], [78], to represent time-

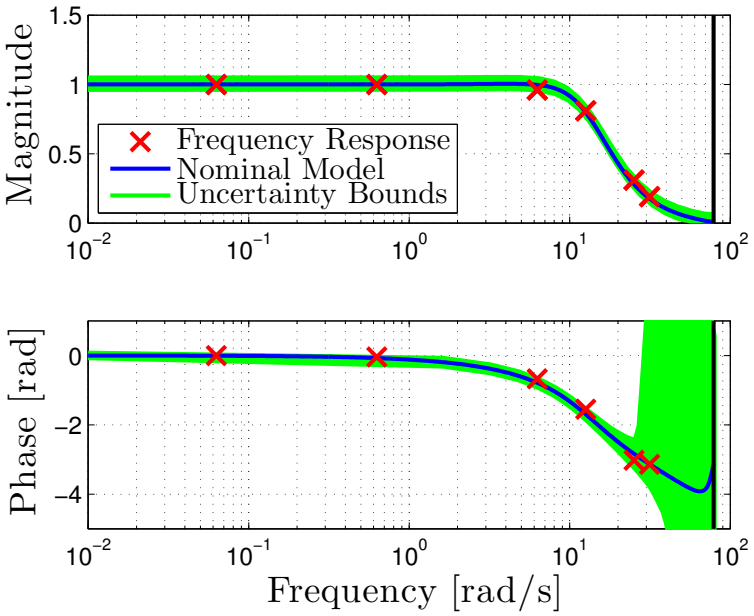

Figure 5.7: Dynamic uncertainty bounds for the actuator model.

varying time delays in discrete-time. In this work, the only considered delays are in the controller, whereby the controller receives measurements at 25 Hz, and optimal controller commands are calculated and sent to the actuators and propulsion system after a short (and time-varying) delay. Because the controller operates in discrete-time, the entire aircraft system has been discretized for analysis. In reality however, the aircraft dynamics operate in continuous-time, and the expected time delays will likely not occur in discrete intervals. In fact, the Telemaster delays are consistently less than one time step, and as such, the available multipliers would be a conservative representation of the expected delays. Time delays are therefore lumped together with the conservative and already time-varying uncertainties in place for saturation in order to reduce both conservatism and computational complexity in the analysis process.

Three identical servomotors deflect the UAS control surfaces and are nominally modeled as

$$G_{act}(s) = \omega_{ns}^2/(s^2 + 2\zeta_s\omega_{ns}s + \omega_{ns}^2). \tag{5.3}$$

The natural frequency and damping ratio were experimentally estimated to be $\omega_{ns} = 13.7$ rad/s and $\zeta_s = 0.67$ by measuring the servomotor frequency response in [53]. The same data is also used to quantify the unmodeled actuator dynamics, represented in IQC form as repeated $\mathcal{H}_\infty$ norm bounded dynamic LTI uncertainties $\delta_{act}(z)$. The uncertainties are additively combined with the dynamics as $G_{act}(z) + \delta_{act}(z)$, as shown in Fig. 5.1. An uncertainty bound is chosen such that all frequency response data points are contained within the set of possible unmodeled linear dynamics, as shown in Fig. 5.7. A bound on $\|\delta_{act}\|_\infty$ is first set as the absolute value of the maximum magnitude error between the nominal model and frequency response data points. Random transfer functions are then generated with $\mathcal{H}_\infty$ bounds less than or equal to the uncertainty bound. The uncertainty bound is relaxed slightly so that all phase data points are covered, and then relaxed slightly more to $\|\delta_{act}\|_\infty \leq 0.05$ to account for potential nonlinearities and errors in the data collection.

As the nominal thrust model is assumed to be static, a dynamic LTI uncertainty, $\|\delta_T\|_\infty \leq 0.2$, is added to the thrust model to account for unmodeled dynamics and nonlinearities in the lookup table-based thrust model from [53].

The control input uncertainties (control group) consist of 4 dynamic LTI uncertainties, $\delta_{act_E}(z)$,$\delta_{act_A}(z)$, $\delta_{act_R}(z)$, $\delta_T(z)$ and four static LTV uncertainties $\delta_{\sigma_E}(k)$, $\delta_{\sigma_A}(k)$, $\delta_{\sigma_R}(k)$,

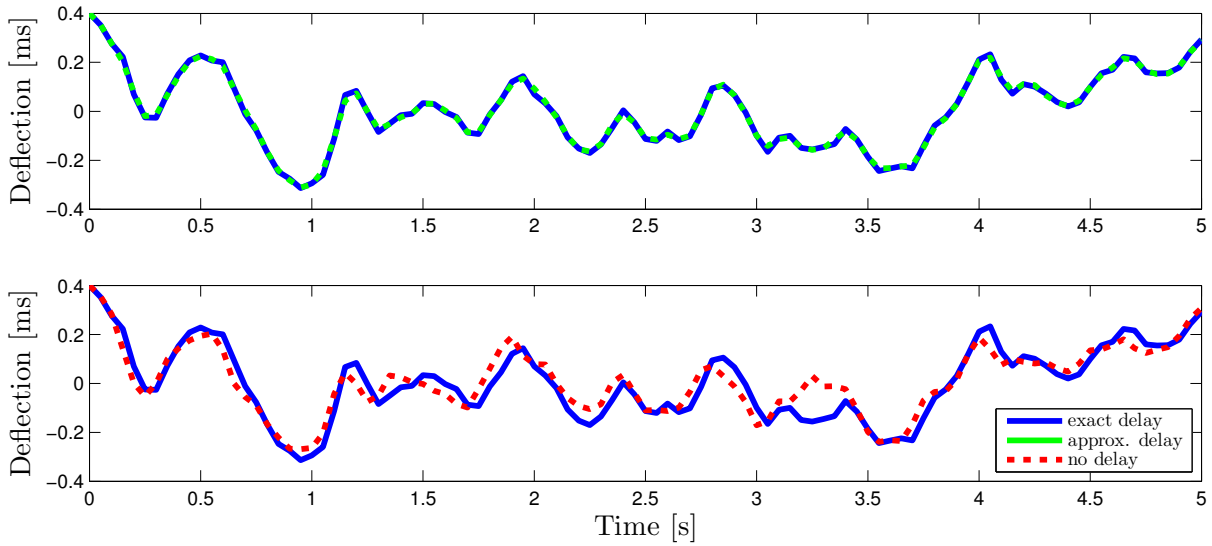

Figure 5.8: Approximating a time-varying delayed input to the servo model.

and $\delta_{\sigma_T}(k)$.

5.4.1 Time-Varying Partial Time-Step Delays

After sensor data is passed to the controller, a new set of control laws for the aircraft thrust and servomotors must be calculated, leading to a small delay in the controller commands. Depending on the processor utilized, number of calculations, and whether or not table lookups and interpolations are required, time delays may vary in magnitude and even be time varying.

Keep in mind that controller commands are implemented in discrete-time while the aircraft dynamics are actually in continuous-time. It is therefore possible that a delay may be less than one time-step if the system is modeled entirely in discrete-time.

While Kao et al. [77], [78] have developed IQC multipliers for time-varying time-step interval delays in discrete time, there is currently no IQC machinery for partial time-step delays. As a discrete-time controller is typically used with the continuous time dynamic UAS system, representing controller delays as interval time-delays may not be appropriate. To overcome this problem, it is proposed that the controller commands are passed through an uncertain transfer function which approximates the effects of a partial time-step delay and can utilize existing IQC multipliers for analysis purposes.

Time delays are approximated by passing controller commands through the transfer function:

$$G_\tau(z) = \frac{(1-\delta_\tau)z + \delta_\tau}{z}, \tag{5.4}$$

where δ_τ represents the current time-delay, normalized by the system time-step so that $\delta_\tau \in [0,\ 1]$. It can be clearly seen that a delay of $\delta_\tau = 0$ in $G_\tau(z)$ represents a unitary gain, and thus no delay, while $\delta_\tau = 1$ yields the pure integration term, $\frac{1}{z}$, and thus a full time-step delay. For intermediary values of δ_τ, the output from $G_\tau(z)$ is simply linearly interpolated using the current input and the previous input command.

As the delayed command signal will be passed through a servo model before entering the aerodynamic model for the three actuator commands, the effect of the time-varying delay is visualized in continuous time using the actuator deflection (as a normalized PWM signal in the range $[-0.4,\ 0.4]$), a time-varying zero-order hold input command, and a time-varying delay history. A time-step of $\tau = 0.04$ s, time delay variation rate of $|\nu_\tau| \leq 1$, and second-

order servo model with a natural frequency of $\omega_n = 13.7$ rad/s and a damping ration of $\zeta = 0.67$ are used for this example, where the continuous time servo model is given as

$$G_\delta(s) = \left[\begin{array}{c|c} A_\delta & B_\delta \\ \hline C_\delta & D_\delta \end{array}\right] = \left[\begin{array}{cc|c} 0 & 1 & 0 \\ -\omega_n^2 & -2\zeta\omega_n & \omega_n^2 \\ \hline 1 & 0 & 0 \end{array}\right]. \tag{5.5}$$

Where the state space representation of the discrete-time delay filter is

$$G_\tau(z) = \left[\begin{array}{c|c} A_\tau & B_\tau \\ \hline C_\tau & D_\tau \end{array}\right] = \left[\begin{array}{c|c} 0 & 1 \\ \hline \delta_\tau & 1-\delta_\tau \end{array}\right], \tag{5.6}$$

and the discrete time servo model is denoted $\hat{G}_\tau(z)$, the approximate delayed discrete time system is given as:

$$G_\tau(z)\hat{G}_\delta(z) = \left[\begin{array}{cc|c} A_\tau & B_\tau C_\delta & B_\tau D_\delta \\ 0 & A_\delta & B_\delta \\ \hline C_\tau & D_\tau C_\delta & D_\tau D_\delta \end{array}\right]. \tag{5.7}$$

The response to the continuous-time delayed actuator model $e^{-\tau\delta_\tau}G_\delta(s)$ is shown against the approximated delay discrete-time system $G_\tau(z)\hat{G}_\delta(z)$ in Figure 5.8 for identical input command and time-varying delay time histories. A comparison between $e^{-\tau\delta_\tau}G_\delta(s)$ and the discrete-time actuator without delay, $\hat{G}_\delta(z)$, is also shown. It is clear from Figure 5.8 that the delay filter $G_\tau(z)$ accurately represents the effect of a partial time-step delay in discrete-time. Additionally, given the delay variation rate $|\nu_\tau| \leq 1$, ignoring the delay effect may lead

Table 5.3: Worst-Case Performance IQC Bounds

	nominal	control	aero	dynamic	control & aero	control & dynamic	aero & dynamic	All
μ	5.9324	8.5119	7.5420	10.0785	10.5005	14.7388	13.6103	19.4337
IQC	5.9324	8.5119	8.2419	10.0785	12.0418	14.7406	15.6035	23.1363

to an inaccurate representation of the actuator deflection.

If δ_τ is a known constant, (5.4) can be incorporated into the UAS system at the cost of only 1 state per delayed channel. Likewise, if δ_τ is an unknown constant or unknown time-varying parameter, δ_τ can simply be represented using the SLTI and SLTV IQC multipliers, respectively.

5.5 Analysis Results

Various combinations of uncertainty groups are analyzed as $\epsilon\Delta \star M$ for $\epsilon \in [0\ 1]$ with performance output $\bar{\boldsymbol{z}} = \bar{\boldsymbol{P}}_0$ in Figs. 5.9 and 5.10. IQC analysis is used to determine upper bounds on $\|\epsilon\Delta\star M\|_{\ell_2\mapsto\ell_2}$, and μ-analysis (freezing M and using μ-Tools) is used to determine lower bounds [19]. These lower bounds were found to yield representative results when compared to lower bound techniques that directly account for time-varying uncertainties [79].

Four different LTI $\mathcal{H}_\infty$ controllers are analyzed. The parameter values defining the performance outputs used for synthesis of the three controllers can be found in Table 5.4. For

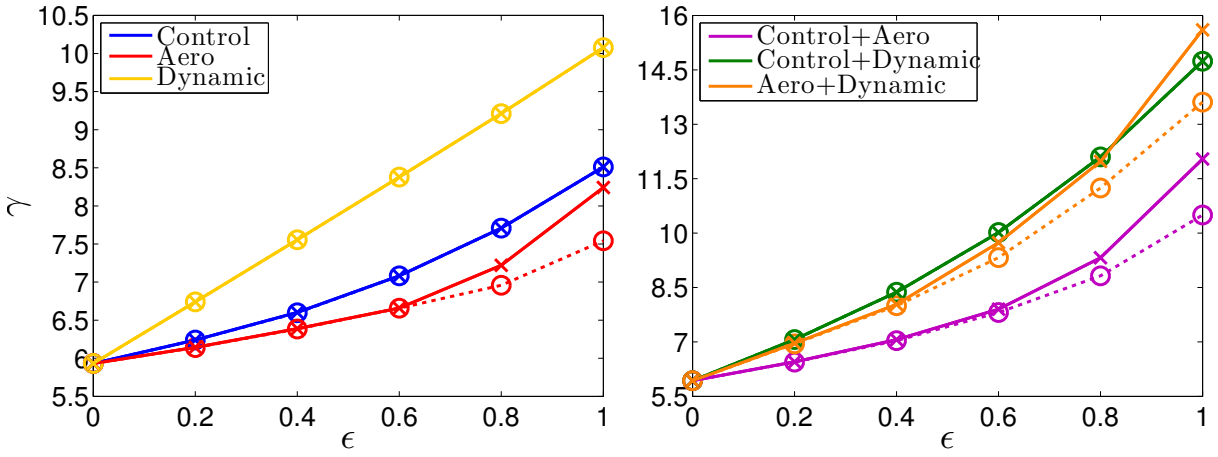

Figure 5.9: IQC upper (×) and μ lower (o) bounds on $\|\epsilon\Delta \star M\|_{\ell_2 \mapsto \ell_2}$ for individual and coupled uncertainty groups on controller 3.

synthesis, all performance outputs take the structure

$$\boldsymbol{z} = [c_1 p + c_2\delta_{A_c}, c_3 q + c_4\delta_{E_c}, c_5 r + c_6\delta_{R_c}, c_7 u + c_8\delta_T, c_9\phi, c_{10}\theta, \ldots$$

$$c_{11}h + c_{12}\delta_{E_c}, c_{13}X, c_{14}Y, c_{15}\delta_{E_c}, c_{16}\delta_{A_c}, c_{17}\delta_{R_c}, c_{18}\delta_T]^T.$$

Analysis of controller #3 subject to various groups of uncertainties is first presented. The controllers are then compared to each other using IQC analysis and validated by simulations. Finally, the tuning process from controller #2 to controller #3 is discussed. Note that controller #3 was used in the aerodynamic coupling example shown in Fig. 5.6.

5.5.1 Analysis of Controller # 3

Upper and lower bounds on the worst-case performance for various groups of uncertainties are also presented in Table 5.3. As it can be difficult to relate worst-case performance

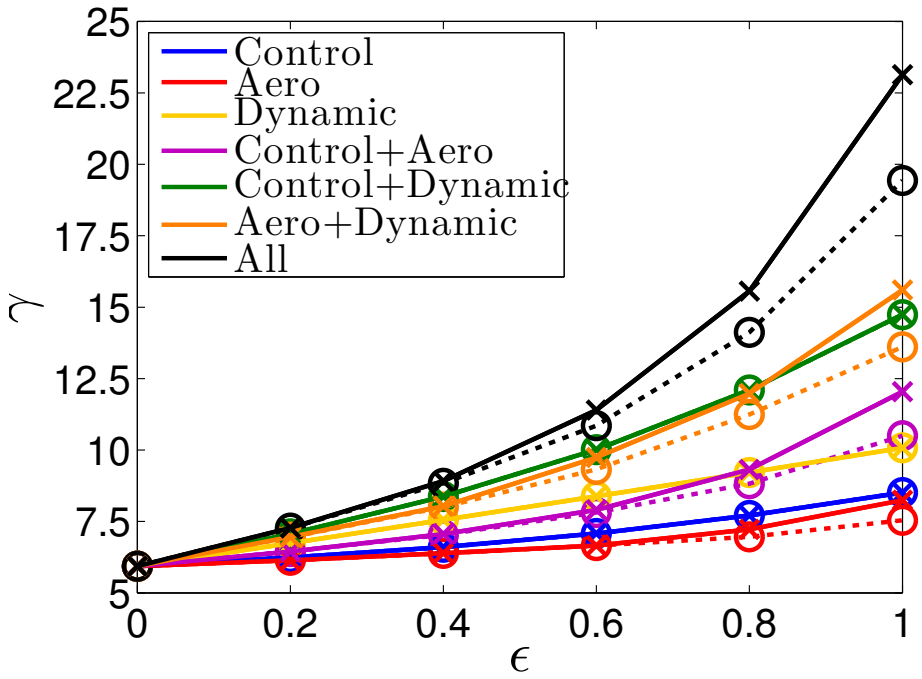

Figure 5.10: IQC upper (×) and μ lower (o) bounds on $\|\epsilon\Delta \star M\|_{\ell_2 \mapsto \ell_2}$ for all uncertainty groups on controller 3.

bounds to expected flight test results, a useful metric is the percent degradation in worst-case performance (ϵ=1) due to uncertainties, with respect to the nominal worst-case performance.

The nominal worst-case performance bound for controller #3 is 5.93 with a degradation of 43% for the control group, 39% for the aero group, and 70% for the dynamic group. There is a less than 10% difference between the upper and lower bound for the aero group. While the control group does have SLTV uncertainties, there was no noticeable difference between the IQC upper and μ lower bounds. When the control and dynamic groups are coupled together, the resulting degradation is 148%. The degradation from coupling the control & aero groups and aero & dynamic groups is 103% and 163%, respectively. All of the coupled groups resulted in more than a linear combination of performance degradations. For instance, the

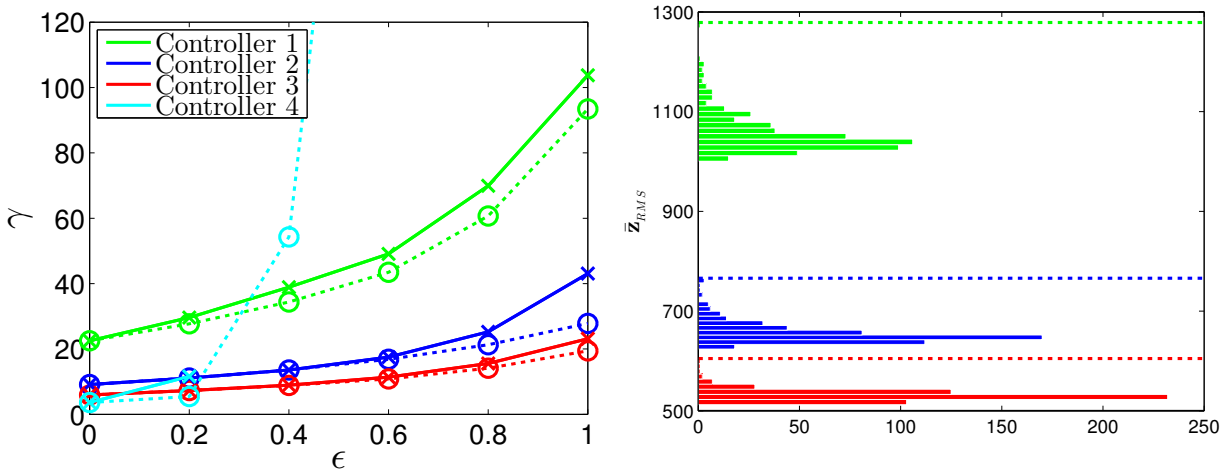

Figure 5.11: IQC upper (×) and μ lower (o) bounds for four different controllers are calculated (left). The worst-case RMS performance is represented by a bar on the corresponding simulation histograms (right).

degradation of the dynamic group (70%) and aero group (39%) added together is 109%, while the coupled analysis produced a degradation of 163%. While the control group had a higher IQC upper bound than the aero group, the aero group resulted in a higher IQC bound when coupled with the dynamic group (as opposed to the control & dynamic group). The time-varying uncertainties in the aero group resulted in a more pronounced gap between upper and lower bounds for the coupled groups with a 0.01% difference for the control & dynamic group and 15% for both the aero & dynamic and control & aero groups.

Finally, all uncertainties are analyzed together, resulting in an upper bound on worst-case performance of 23.14 and a degradation in worst-case performance between 228% and 290%. Looking at Fig. 5.10, the effect of coupling the third uncertainty group with the other two is the most dramatic, almost doubling the upper bound of the two uncertainty group case. The upper-lower bound gap was also most pronounced with all uncertainties, at 19%.

5.5.2 Comparing Controllers

The four $\mathcal{H}_\infty$ controllers in Table 5.4 are analyzed against all uncertainties. The resulting upper and lower bounds on worst-case performance are shown in Fig. 5.11. Additionally, nominal performance values and upper and lower bounds at $\epsilon = 1$ can be found in Table 5.4.

Controller #1 had the worst performance with a nominal ℓ_2-gain of 22.5 and an upper bound on worst-case performance of 103.7. This corresponds to a performance degradation of over 360% due to uncertainties and nonlinearities. Controllers #2 and #3 both performed significantly better, with nominal performance values of 9.11 and 5.93, and upper bounds on worst-case performance of 43.1 and 23.1, respectively. As mentioned in the previous section, controller #3 showed a degradation of around 290%, while the degradation for controller #2 is approximately 373%. Controller #4 had the best nominal performance value of 3.60. The lower bound on worst-case performance increased sharply as ϵ increased, indicating a tradeoff between controller performance and robustness. IQC upper bounds were unable to be found for $\epsilon \geq 0.4$ and lower bounds were unable to be found at $\epsilon = 1$.

While controllers #2 and #1 showed similar percent degradations in performance and were both analyzed with identical uncertainties, Fig. 5.11 clearly shows that controller #2 is a significant improvement over controller #1. It is possible that the worst case performance of controllers #2 and #3 are very close since the upper bound for controller #3 is 23.1 and the lower bound for controller #2 is a close 27.8. It is clear, however, that both controllers #2 and #3 are more robust to uncertainties than controller #1.

To help validate these analysis results, a large number of nonlinear simulations are performed with each controller. Simulations are conducted in the previously described `MATLAB` simulation environment which involves the nonlinear flight dynamic model, with actuator dynamics, subjected to 3.5 m/s steady winds, moderate turbulence generated by the low altitude Dryden model, time-varying time delays randomly sampled from a range of 0 to 0.005 seconds, and sensor noise. Each controller is flown for 500 simulations of 5 continuous loops, with the root-mean-square (RMS) error of the performance channel $\bar{\boldsymbol{z}} = \bar{\boldsymbol{P}}_0$ calculated for every five loops (n_τ=3500 time-steps) as $\bar{\boldsymbol{z}}_{RMS} = \sqrt{\sum_{k=1}^{n_\tau} \bar{\boldsymbol{z}}(k)^T \bar{\boldsymbol{z}}(k)}$. The simulation results are presented in a histogram in Fig. 5.11. The worst-case RMS error is marked with a horizontal bar. Controller #4 was unstable for all simulations and is therefore not shown.

While it is difficult to predict the worst-case RMS errors, comparing the two plots in Fig. 5.11 reveals similar trends between controllers. Both the IQC analysis and simulation validation show that controller #1 has by far the worst performance. Additionally, the performance of controllers #2 and #3 are much closer together than #2 and #1. Since IQC analysis can be performed much faster than simulations, it is very encouraging that worst-case controller comparison correlates with nonlinear simulations.

While controller comparisons are useful to control designers, a similar process can be used when only one controller is available. Multiple payload configurations with the same controller, for instance, can be easily compared. The relative robustness of the controller to various airframe configurations can be assessed in this manner.

Table 5.4: Controller Parameters

#	nom.	μ	IQC	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	c_{16}	c_{17}	c_{18}
1	22.5	93.4	104	0.1	0.3	0.1	0	0.1	0.6	0.12	2	0	0.3	0.01	0.3	0.01	0.01	0.3	0.3	0.6	2
2	9.11	27.8	43.1	0.1	0.3	0.1	0.4	0.1	0.8	0.07	2	0.04	0.3	0.03	0	0.03	0.03	0.3	0.3	0.8	2
3	5.93	19.4	23.1	0.1	0.4	0.1	0.4	0.1	0.6	0.07	2	0.4	0.3	0.05	0	0.04	0.04	0.4	0.4	0.6	2
4	3.60	N/A	N/A	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0

5.5.3 Controller Tuning with IQCs

Since direct comparisons between controllers have been shown to be viable in the above sections, it follows that IQC analysis could be a very useful tool in the control design process. An IQC-in-the-loop controller tuning process such as the one in Section 5.1 could bypass time-consuming simulations and allow for the tuning process to be automated using nonlinear optimization algorithms to produce controllers that are robust to uncertainties, unmodeled dynamics, and nonlinearities.

Fig. 5.12 shows three intermediate tuning steps in between controllers #2 and 3. The initial controller has a performance value of 43.08. The first iteration (2-A) reduces the performance value to 39.13 by increasing the penalty on position through coefficients c_{11}, c_{13}, and c_{17}, airpseed through c_7, and decreases the weight on roll angle with the coefficient c_9. By increasing the pitch rate and elevator coupled term, c_4, and decreasing the penalty on the height and elevator coupling term, c_{12}, the performance value is lowered to 32.08 for controller 2-B. Controller 2-C is formed by increasing the penalty on the actuator commands through c_{15}, c_{16}, c_{17}. The performance value for 2-C is 26.72. Finally, controller #3 is obtained by decreasing the airpseed penalty and h-δ_{E_c} coupling and increasing the roll angle penalty and q-δ_{E_c} coupling. The worst-case performance value of controller #3 is 23.14, a 46% reduction

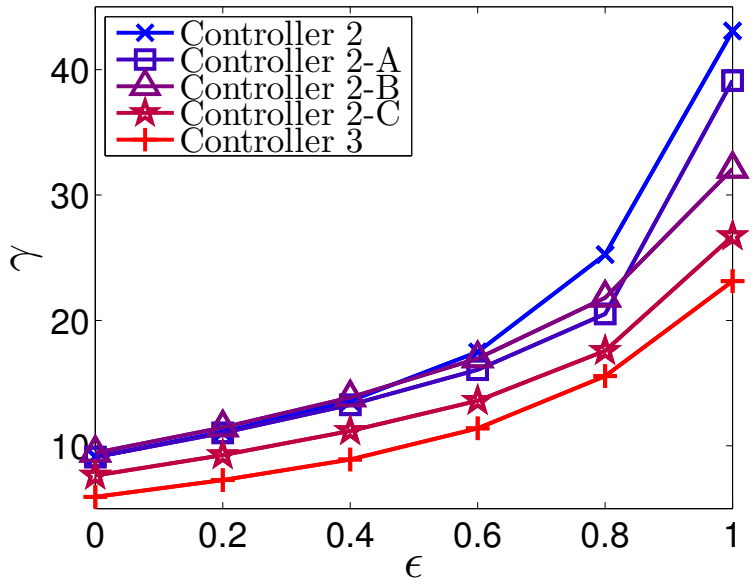

Figure 5.12: IQC upper bounds for the tuning sequence from controller 2 to controller 3.

over that of #2.

Considering the nominal performance values ($\epsilon = 0$) for the controller iterations, controller #2 has a nominal performance value of 9.106. From Controller 2 to 2-A, the nominal performance value actually increases slightly to 9.109. From 2-A to 2-B, the nominal performance continues to increase, up to 9.438. From 2-B to 2-C, the nominal performance drops to 7.636, and from 2-C to 3, it drops to 5.932. Conversely, the worst-case performance bounds monotonically decreased throughout the tuning process.

It is important to note that the changes in nominal performance value are not reflective of the changes in worst-case performance, and a simple $\mathcal{H}_\infty$ norm analysis of the nominal closed-loop system may not be indicative of a well performing controller when nonlinearities and unmodeled dynamics are taken into account.

5.5.4 Observations on IQC Analysis

IQC analysis for large complex systems remains a difficult task. Throughout the course of this work, the author has discovered that IQC analysis results for 6 DOF UAS are sensitive to a number of factors. First of all, UAS analysis problems were consistently sensitive to the chosen basis function pole. Poles located close to the unit circle tended to result in tighter upper bounds on the ℓ_2-gain performance level, γ. All analysis in this chapter used a pole of $\lambda = 0.9997$. In general, the larger the analysis problem, the more difficult it was to solve. Larger problems would potentially result in very conservative upper-bounds, unsatisfied LMIs, or errors with the chosen convex optimization solver. For example, if a minimal realization is not used for the aerodynamic uncertainties, the corresponding performance bound would be higher than if a minimal realization was analyzed, despite the fact that both systems have equivalent input-output properties. Both `SeDuMi` and `SDPT3` were used at candidate solvers, but all results in this chapter used `SDPT3`, as it was found to be more robust, and provided satisfied LMI results more of the time. One potential method to ensure the convex optimization solver satisfies the analysis problem's LMIs is to shift all constraints. If a large shift is chosen, however, the resulting upper bound on performance level may end up being conservative. Typically, a shift of 1×10^{-8} was used in this work. Another method to potentially improve upper bounds is to increase the basis function length. Increasing the basis length would occasionally result in a slightly lower performance bound when the LFR contained SLTV uncertainties. Unfortunately, increasing the basis length additionally increases the number of LFR states, and consequently the number of decision variables in

the IQC multipliers and the Lyapunov matrix. The added system states would often increase the solution time dramatically, so a short basis function was used for most analysis. Since γ^2 is optimized instead of γ in the convex solution to the IQC problem, the solver is much more sensitive to smaller γ's, and numerical discrepancies are much more likely to occur for values of γ less than one. Scaling γ, therefore, may lead to more accurate results. Additionally, it was found that normalizing uncertainties to make Δ contractive sometimes produced less conservative upper bounds.

Chapter 6

A Fast Oracle-Based Algorithm for the Discrete-Time IQC Problem

As discussed in Chapter 4, through application of the KYP lemma, the frequency-dependent infinitely constrained IQC problem can be equivalently posed as a frequency-independent finite dimensional convex optimization problem and solved using semidefinite programming tools. This chapter presents an oracle for the discrete-time IQC problem in addition to outlining a cutting plane algorithm to generate candidate solutions for the oracle to check. The chapter is organized as follows. Section 6.1 introduces orthogonal and orthogonal symplectic matrices that will be used in the cutting plane algorithm, Section 6.2 presents the discrete-time IQC oracle, Section 6.3 provides a brief overview of two non-KYP-based algorithms for solving the IQC problem, Section 6.4 provides the necessary background information and pseudocode for the implemented cutting plane algorithm, Section 6.5 describes the re-

formulation of the oracle into a more robust form, Section 6.6 describes the theory and implementation of a robust eigenvalue solver, and finally, Section 6.7 provides examples of the IQC oracle applied to large systems and complex engineering systems. In this chapter, the `MATLAB` notation $\boldsymbol{x}(i:j)$ is used to represent $[\boldsymbol{x}(i), \boldsymbol{x}(i+1), \ldots, \boldsymbol{x}(j-1), \boldsymbol{x}(j)]^T$ for convenience.

6.1 Orthogonal and Orthogonal Symplectic Matrices

Two main types of orthogonal matrices are used in order to manipulate matrix structures and annihilate certain terms via matrix multiplication, Givens rotations and Householder reflections [80], [81]. A Givens rotation in $\mathbb{R}^{m\times m}$ is given by

$$\mathscr{G}(i,j,\theta) = \begin{bmatrix} I_{i-1} & 0_{i-1\times 1} & 0_{i-1\times j-i-1} & 0_{i-1\times 1} & 0_{i-1\times m-j} \\ 0_{1\times i-1} & \cos\theta & 0_{1\times j-i-1} & \sin\theta & 0_{1\times m-j} \\ 0_{j-i-1\times i-1} & 0_{j-i-1\times 1} & I_{j-i-1} & 0_{j-i-1\times 1} & 0_{j-i-1\times m-j} \\ 0_{1\times i-1} & -\sin\theta & 0_{1\times j-i-1} & \cos\theta & 0_{1\times m-j} \\ 0_{m-j\times i-1} & 0_{m-j\times 1} & 0_{m-j\times j-i-1} & 0_{m-j\times 1} & I_{m-j} \end{bmatrix},$$

where i and j denote the locations of the $\cos\theta$ terms along the diagonal. For convenience, we define the special case Givens matrices $\mathscr{G}_o(i,\theta) = \mathscr{G}(i,i+1,\theta)$ and $\mathscr{G}_s(i,\theta) = \mathscr{G}(i,m/2+i,\theta)$. Clearly $\mathscr{G}_s$ only exists if m is even, which holds for all uses of $\mathscr{G}_s$ in this work. Note that $\mathscr{G}$ and the special case $\mathscr{G}_o$ are orthogonal, and $\mathscr{G}_s$ is orthogonal symplectic. A Givens rotation

can be used to annihilate a single term from the left or the right through proper choice of the angle $\theta \in [0, 2\pi]$.

Householder reflections $\mathscr{H} \in \mathbb{S}^m$ are used to annihilate part of a row or column of a multiplied matrix by rotating part of a vector from that matrix into its null space. We define the Householder reflection of a tall, real-valued vector $\boldsymbol{w} \in \mathbb{R}^m$ to annihilate $\boldsymbol{w}(k+1:m)$ as

$$\mathscr{H}(k, \boldsymbol{w}) = \begin{bmatrix} I_{k-1} & 0_{k-1 \times m-k+1} \\ 0_{m-k+1 \times k-1} & I_{m-k+1} - 2\frac{\boldsymbol{v}\boldsymbol{v}^T}{\boldsymbol{v}^T\boldsymbol{v}} \end{bmatrix},$$

where $\boldsymbol{v}$ is determined from $\boldsymbol{w}$ and k in Algorithm 1, taken from [81] and presented below.

Algorithm 1 Householder reflection vector

Require: $k,\ n,\ \boldsymbol{w}$

Assign $\boldsymbol{v} = \mathbf{0}_{m-k+1}$

Set $\boldsymbol{v}(1) = \boldsymbol{w}(k) + \boldsymbol{w}^T\boldsymbol{w} \operatorname{sign} \boldsymbol{w}(k)$

for $i = 2, 3, \ldots, m-k+1$ **do**

Set $\boldsymbol{v}(i) = \boldsymbol{w}(i+k-1)$

end for

Return: $\boldsymbol{v}$

Left multiplying $\boldsymbol{w}$ by the Householder reflection matrix $\mathscr{H}(k, \boldsymbol{w})$ yields

$$\mathscr{H}(k, \boldsymbol{w})\boldsymbol{w} = \begin{bmatrix} \boldsymbol{x} \\ \mathbf{0}_{m-k} \end{bmatrix}, \tag{6.1}$$

where $\boldsymbol{x} \in \mathbb{R}^k$. Householder reflections can be similarly used to annihilate rows of a matrix by being multiplied from the left. This is easily demonstrated by taking the transpose of (6.1), which yields

$$\boldsymbol{w}^T \mathscr{H}(k, \boldsymbol{w}) = \begin{bmatrix} \boldsymbol{x}^T & \mathbf{0}_{m-k}^T \end{bmatrix}.$$

6.2 A Discrete Time IQC Oracle

Recall that the IQC stability and performance inequality given in (4.7) is

$$F(z) = \mathcal{M}(z)^* S \mathcal{M}(z) \prec 0,$$

for all $z \in \mathbb{D}$, where $\mathbb{D} = \{z | z = e^{j\omega} \text{ for all } \omega \in [-\pi,\ \pi]\}$ is the set of points on the complex unit circle, and the transfer matrix $\mathcal{M}(z)$ and its adjoint are given by

$$\mathcal{M}(z) = \mathcal{D} + \mathcal{C}(zI - \mathcal{A})^{-1}\mathcal{B},$$

$$\mathcal{M}(z)^* = \mathcal{D}^T + \mathcal{B}^T(I - z\mathcal{A}^T)^{-1}\mathcal{C}^T z.$$

As it will be useful in the following sections, $F(z)$ and $F(z)^{-1}$ can be expanded as

$$F(z) = \mathcal{R} + \begin{bmatrix} \mathcal{F}^T & z\mathcal{B}^T \end{bmatrix} \begin{bmatrix} zI - \mathcal{A} & 0 \\ -\mathcal{Q} & I - z\mathcal{A}^T \end{bmatrix}^{-1} \begin{bmatrix} \mathcal{B} \\ \mathcal{F} \end{bmatrix},$$

$$F(z)^{-1} = \mathcal{R}^{-1} + \begin{bmatrix} \mathcal{R}^{-1}\mathcal{F}^T & z\mathcal{R}^{-1}\mathcal{B}^T \end{bmatrix} (H - zN)^{-1} \begin{bmatrix} -\mathcal{B}\mathcal{R}^{-1} \\ -\mathcal{F}\mathcal{R}^{-1}, \end{bmatrix}, \tag{6.2}$$

where

$$H = \begin{bmatrix} \mathcal{B}\mathcal{R}^{-1}\mathcal{F}^T - \mathcal{A} & 0 \\ \mathcal{F}\mathcal{R}^{-1}\mathcal{F}^T - \mathcal{Q} & I \end{bmatrix}, \; N = \begin{bmatrix} -I & -\mathcal{B}\mathcal{R}^{-1}\mathcal{B}^T \\ 0 & \mathcal{A}^T - \mathcal{F}\mathcal{R}^{-1}\mathcal{B}^T \end{bmatrix}. \tag{6.3}$$

Recall that $\mathcal{Q}$, $\mathcal{F}$, and $\mathcal{R}$ are functions of S, and subsequently, γ. Note also that $\mathcal{Q}$ and $\mathcal{R}$ are guaranteed to be symmetric due to their construction in (4.14). In addition, $\mathcal{R}$ is invertible for all cases of interest.

For (4.7) to hold for a candidate solution $S_0(\gamma_0^2)$, and be negative definite, $\lambda_{max}(F(z))$ must be negative for all points $z \in \mathbb{D}$. To avoid introducing a Lyapunov matrix, we use $F(z)^{-1}$ to check if $\lambda_{max}(F(z)) = 0$ for any $z \in \mathbb{D}$. Since $F(z)$ is a continuous function on $\mathbb{D}$, if $\lambda_{max}(F(z_0)) < 0$ for a point $z_0 \in \mathbb{D}$, and $F(z)^{-1}$ exists for all $z \in \mathbb{D}$, we can conclude that $S_0(\gamma_0^2)$ satisfies (4.7) and so $\|\Delta \star M\|_{\ell_2 \to \ell_2} < \gamma_0$ for all $\Delta \in \mathbf{\Delta}$. If $F(z)^{-1}$ does not exist for some $z \in \mathbb{D}$, then critical frequencies ω_i exist such that $\lambda_{max}(F(e^{j\omega_i})) = 0$.

For example, if the maximum eigenvalues of a matrix $F(z)$, for $z \in \mathbb{D}$, had the distribution shown in Fig. 6.1, the boundary condition $\lambda_{max}(F(e^{j\pi})) < 0$ would clearly hold, but there would exist several critical frequencies, corresponding to zero eigenvalues, for which $F(z)^{-1}$

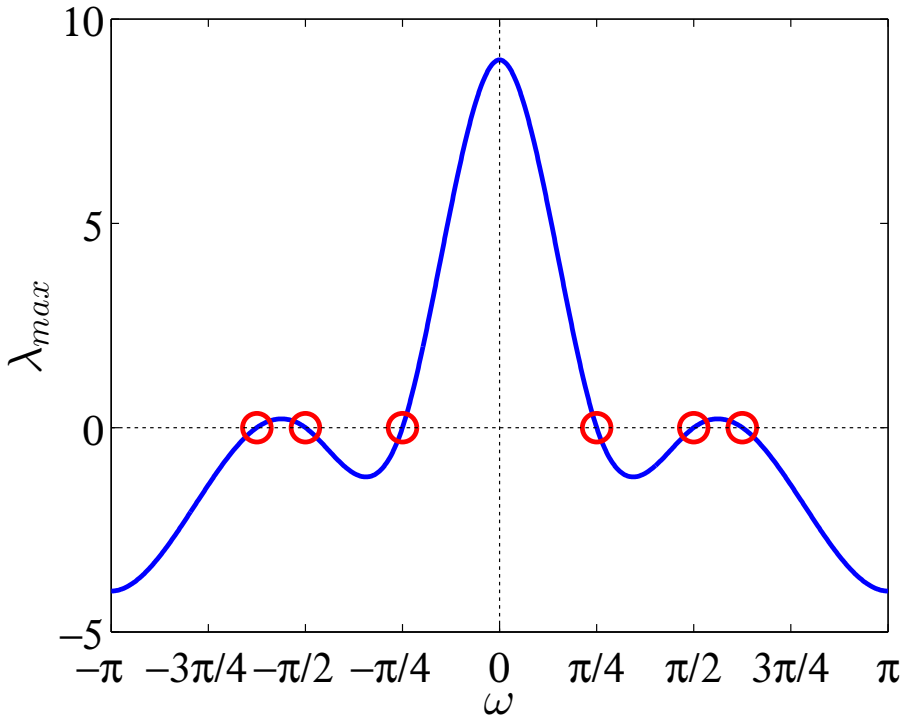

Figure 6.1: An example of critical frequencies returned by the discrete-time IQC oracle.

would not exist, and the candidate solution $S_0(\gamma_0^2)$ would not satisfy the inequality in (4.7). Furthermore, given just the critical frequencies $\pm\pi/4$, $\pm\pi/2$, and $\pm 5\pi/8$, we know that the inequality in (4.7) is violated in the ranges $[-5\pi/8,\ -\pi/2]$, $[-\pi/4,\ \pi/4]$, and $[\pi/2,\ 5\pi/8]$.

While it can determine feasibility of solutions, the oracle cannot determine optimality, and it is thus up to a non-KYP-based IQC algorithm to generate an appropriate solution to minimize γ^2, as is inherently done in the KYP solution (4.15).

We now present the main theorem for the discrete-time IQC oracle.

Theorem 2. *Given an invertible matrix $\mathcal{R} \in \mathbb{S}^{n_\mathcal{B}}$, $\mathcal{A} \in \mathbb{R}^{n_\mathcal{A} \times n_\mathcal{A}}$ with no eigenvalues in $\mathbb{D}$, and $F(-1) \prec 0$, then $F(z) \prec 0$ for all $z \in \mathbb{D}$ if and only if the regular matrix pencil $H - zN$ has no eigenvalues in $\mathbb{D}$, where H and N are defined in (6.3).*

Proof. An important difference from the continuous-time case is the requirement for the boundary condition, chosen here as $F(-1) \prec 0$. The corresponding continuous-time inequality $F_c(j\omega) \prec 0$ has a natural boundary condition at an infinite frequency, where $F_c(j\infty) \prec 0$ always holds.

We first show that $F(z)$ is negative definite for all $z \in \mathbb{D}$ if and only if it is also nonsingular and $F(-1) \prec 0$. For the "only if" direction, $F(z)$ is clearly nonsingular for all $z \in \mathbb{D}$ if $F(z) \prec 0$ for all $z \in \mathbb{D}$. We now prove the "if" direction and assume that $F(z)$ is nonsingular for all $z \in \mathbb{D}$. Since $F(-1) \prec 0$, $-1 \in \mathbb{D}$, and $F(z)$ is a continuous function on $\mathbb{D}$, $F(z) \prec 0$ must hold for all $z \in \mathbb{D}$.

Next, utilizing a similar argument to the ones used in the proofs of [35, Theorem 1] and [82, Theorem 1], we show that $F(z_0)$ is singular for some $z_0 \in \mathbb{D}$ if and only if z_0 is an eigenvalue of the pencil $H - zN$. We start by proving the "only if" direction. Since $F(z_0)$ is singular, then there exists a nonzero vector $\boldsymbol{x} \in \mathbb{C}^{n_\mathcal{B}}$ such that $F(z_0)\boldsymbol{x} = \mathbf{0}$, that is,

$$\begin{bmatrix} \mathcal{F}^T & z_0\mathcal{B}^T \end{bmatrix} \boldsymbol{y} + \mathcal{R}\boldsymbol{x} = \mathbf{0}, \quad \text{where} \quad \boldsymbol{y} = \begin{bmatrix} z_0 I - \mathcal{A} & 0 \\ -\mathcal{Q} & I - z_0\mathcal{A}^T \end{bmatrix}^{-1} \begin{bmatrix} \mathcal{B} \\ \mathcal{F} \end{bmatrix} \boldsymbol{x}.$$

Clearly, since $\boldsymbol{x}$ is nonzero, then $\boldsymbol{y}$ is also a nonzero vector. Thus, $\boldsymbol{x} = -\mathcal{R}^{-1} \begin{bmatrix} \mathcal{F}^T & z_0\mathcal{B}^T \end{bmatrix} \boldsymbol{y}$, and so,

$$\begin{bmatrix} z_0 I - \mathcal{A} & 0 \\ -\mathcal{Q} & I - z_0\mathcal{A}^T \end{bmatrix} \boldsymbol{y} = \begin{bmatrix} \mathcal{B} \\ \mathcal{F} \end{bmatrix} \boldsymbol{x} = - \begin{bmatrix} \mathcal{B} \\ \mathcal{F} \end{bmatrix} \mathcal{R}^{-1} \begin{bmatrix} \mathcal{F}^T & z_0\mathcal{B}^T \end{bmatrix} \boldsymbol{y},$$

which is equivalent to $(H - z_0 N)\boldsymbol{y} = \mathbf{0}$. Therefore, $H - z_0 N$ is singular and, hence, z_0 is an eigenvalue of $H - zN$. To prove the "if" direction, we assume z_0 is an eigenvalue of $H - zN$, and so, $(H - z_0 N)\boldsymbol{y} = \mathbf{0}$ for some nonzero vector $\boldsymbol{y}$. Defining the nonzero vector $\boldsymbol{x}$ as $\boldsymbol{x} = -\mathcal{R}^{-1}\begin{bmatrix}\mathcal{F}^T & z_0\mathcal{B}^T\end{bmatrix}\boldsymbol{y}$, we can equivalently express $(H - z_0 N)\boldsymbol{y} = \mathbf{0}$ as $F(z_0)\boldsymbol{x} = \mathbf{0}$, thus proving that $F(z_0)$ is singular. □

It directly follows that (4.7) holds if the conditions in Theorem 2 hold. Next, we briefly discuss the use of the oracle in non-KYP-based IQC algorithms.

6.3 Fast Algorithms for Solving IQC Problems

Two algorithms for solving IQC problems without the KYP lemma are briefly described, a cutting plane algorithm and a frequency gridding algorithm. Interested readers can find more details and implementation notes on the cutting plane algorithm in [31],[34], and [83], and the frequency gridding algorithm in [21] and [33]. Additionally, the following sections present a version of the cutting plane algorithm used to test the IQC oracle described in Section 6.2. Note that these algorithms are not specific to the discrete-time case and, in fact, all cited references are for continuous-time applications.

The analytic center cutting plane (ACCP) algorithm determines candidate solutions $S_0(\gamma_0^2)$ by setting the values of the decision variables in $S(\gamma^2)$ equal to the corresponding components of the analytic center of a convex set in $\mathbb{R}^{n_x}$, where n_x is equal to the number of decision variables. In each iteration, a halfspace which cuts through the current analytic center of

the convex set is added to the set of halfspaces defining it. The convex set will shrink and eventually converge around a single point. The algorithm used to test the discrete-time IQC oracle is a modified version of the one in [31], and is briefly outlined below. The algorithm is described in more detail in Section 6.4.

An initial conservative hypercube in $\mathbb{R}^{n_x}$ containing the vectorized IQC solution is defined by a set of linear inequalities expressed in matrix form as $A^T\boldsymbol{x} < \boldsymbol{b}$, where "<" corresponds to the componentwise strict inequality, $A = [I_{n_x} \ -I_{n_x}]$, and $\boldsymbol{b} = \mathbf{1}r_0$, with $\mathbf{1}$ denoting a vector with all of its components equal to 1 and r_0 the radius of the largest Euclidean ball that lies in the initial hypercube. The analytic center of this hypercube is $\boldsymbol{x}_c = \mathbf{0}$. An upper bound on γ^2 is set at infinity, and a lower bound is set by solving the following linear program:

$$\begin{aligned} &\text{minimize}: \ \boldsymbol{x}(n_x) = \gamma^2 \\ &\text{subject to}: A^T\boldsymbol{x} < \boldsymbol{b}. \end{aligned} \tag{6.4}$$

The following steps are repeated iteratively:

- The analytic center, $\boldsymbol{x}_c$, of the convex set $\{\boldsymbol{x} \in \mathbb{R}^{n_x} | \ A^T\boldsymbol{x} < \boldsymbol{b}\}$ is calculated. A candidate solution $S_0(\gamma_0^2)$ is then formed using $\boldsymbol{x}_c$.
- Constraints on the frequency-independent portion of the IQC multipliers are checked. If a violated constraint is found, A and $\boldsymbol{b}$ are appended to introduce an additional inequality defining a new halfspace.
- If all IQC multiplier constraints are satisfied, the IQC oracle is called. If no criti-

cal frequencies are found, $S_0(\gamma_0^2)$ is a feasible solution, and a constraint restricting γ^2 to be less than γ_0^2 is added by appropriately appending A and $\boldsymbol{b}$. If the oracle returns a set of critical frequencies, a violated frequency ω_0 is determined as the midpoint of one of the sections of violated frequencies, as suggested in [36]. Using (4.7), we know that $\mathcal{M}(e^{j\omega_0})^* S_0(\gamma_0^2)\mathcal{M}(e^{j\omega_0})$ is not negative definite, and has at least one positive eigenvalue. A new linear inequality constraint is determined as $\boldsymbol{x}_0^T \mathcal{M}(e^{j\omega_0})^* S(\gamma^2)\mathcal{M}(e^{j\omega_0})\boldsymbol{x}_0 \prec 0$, where $\boldsymbol{x}_0$ is the eigenvector corresponding to the largest eigenvalue of $\mathcal{M}(e^{j\omega_0})^* S_0(\gamma_0^2)\mathcal{M}(e^{j\omega_0})$.

- If the solution $S_0(\gamma_0^2)$ is feasible, the upper bound is updated to be $\boldsymbol{x}_c(n_x)$ (i.e., γ_0^2). Otherwise, the lower bound is updated by solving the linear program (6.4) with the updated set of linear inequality constraints, that is, with the appended A and $\boldsymbol{b}$.

- Finally, if the difference between the upper and lower bounds is less than a predefined tolerance, the upper bound is returned and the algorithm terminates.

Although it is not directly used in this work, the frequency gridding algorithm can also be used with the discrete-time IQC oracle, and is included for completeness. Candidate solutions for this method are determined by solving (4.7) as a convex optimization problem subject to a finite set of frequencies as opposed to all of $\mathbb{D}$. If the oracle determines that the candidate solution is not feasible for all frequencies, a violated frequency is determined in the same manner as that used in the ACCP algorithm, and is added to the finite set. This process repeats until a feasible solution is found, in which case the algorithm terminates and

returns the performance value corresponding to the last candidate solution. The frequency gridding algorithm has the disadvantage that an upper bound on γ is only found when the algorithm terminates, and thus there is no direct convergence metric.

6.4 The Analytic Center Cutting Plane Algorithm

As solving an IQC analysis problem via the KYP lemma using convex optimization techniques can be computationally expensive and slow for medium-large analysis problems, we adopt the analytic center cutting plane (ACCP) algorithm from [31]. For this algorithm, we start with a large convex set with dimensions equal to the number of decision variables in the matrix $S(\gamma^2)$ of (4.7). The analytic center of the convex set serves as a candidate solution to the IQC problem. Each iteration, halfspace constraints are added to the convex set, cutting it through its current analytic center. Each step of the algorithm, a lower bound on the performance value γ^2 can be calculated based on the current halfspaces that make up the convex set. Furthermore, if the candidate solution satisfies (4.7), an upper bound on γ^2 can also be determined. Eventually, the convex set will shrink to a very small size and the upper and lower bounds on the square of the ℓ_2-gain performance value γ will converge.

Whereas the convex optimization approach to the IQC problem utilized a Lyapunov matrix to represent the infinite number of constraints on the problem, we utilize an IQC oracle to return a single violated halfspace constraint out of the set of infinite constraints at each iteration. This halfspace is determined by means of solving an eigenvalue problem, which is

computationally inexpensive when compared to solving a large convex optimization problem. The ACCP is thus much less memory intensive than the convex optimization approach for systems with a large number of states, and examples presented in [31] show that the ACCP can solve IQC problems as fast as the convex approach for small problems, and significantly faster for large problems.

6.4.1 The Analytic Center

The center of gravity of the convex set makes the ideal candidate solution for each iteration of the algorithm, as cutting through the CG will quickly shrink the convex set with guaranteed convergence rates. However, as the CG of a convex set is computationally intensive to calculate, we will instead use the analytic center (AC) as an approximation of the CG. Given the convex set $\{\boldsymbol{x} \in \mathbb{R}^{n_x} \,|\, A^T\boldsymbol{x} \leq \boldsymbol{b}\}$ with $A \in \mathbb{R}^{n_x \times n_b}$ and $\boldsymbol{b} \in \mathbb{R}^{n_b}$, the analytic center by definition minimizes the log barrier function

$$\Phi(\boldsymbol{x}) = -\sum_{i=1}^{n_b} \log(\boldsymbol{b} - A^T\boldsymbol{x}),\ \boldsymbol{x} \in \{\boldsymbol{x} | \boldsymbol{a}_i^T\boldsymbol{x} \leq \boldsymbol{b}_i\ \forall i = 1, 2, \ldots, n_b\},$$

where $\boldsymbol{a}_i$ is the i^{th} column of A.

For our algorithm, the initial convex set P is defined as a hypercube with a radius of $\mathbf{1}_{2n_x} r_0$, where r_0 is a large number such that the optimal solution is likely to be contained within P. The hypercube has a known analytic center of $\boldsymbol{x}_c = \mathbf{0}_{n_x}$, and is defined by the halfspaces $\{\boldsymbol{x} \in \mathbb{R}^{n_x} \,|\, A^T\boldsymbol{x} \leq \boldsymbol{b}\}$, where $A = [I_{n_x} - I_{n_x}]$ and $\boldsymbol{x}$ is the vector of decision variables. In our

algorithm, $\boldsymbol{b}$ will not be directly used. Instead, we use the slack variable $\boldsymbol{s} = \boldsymbol{b} - A^T\boldsymbol{x}$. Lastly, the primal AC is initialized as $\boldsymbol{y} = \mathbf{1}\frac{1}{r_0}$, and the initial slack is defined as $\boldsymbol{s} = \boldsymbol{b} - A^T\boldsymbol{x}_c = \boldsymbol{b}$.

6.4.2 Calculating the Analytic Center

When a halfspace passing through the AC, defined by the linear inequality $\boldsymbol{a}^T\boldsymbol{x} \leq b$, is added to the convex set P, a one step procedure is used to estimate the new primal and dual analytic center values given $\boldsymbol{a}$, A, and $\boldsymbol{s}$ [84]. A value of $0 \leq \beta \leq 1$ controls the depth of the cut r into the convex set. $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{s}$, and A are updated as

$$
\begin{aligned}
r &= \sqrt{\boldsymbol{a}^T(A\,\mathrm{diag}(\boldsymbol{s})^{-2}A^T)^{-1}\boldsymbol{a}}, \\
\boldsymbol{x} &= \boldsymbol{x}_c - \beta(A\,\mathrm{diag}(\boldsymbol{s})^{-2}A^T)^{-1}\boldsymbol{a}/r, \\
\boldsymbol{y} &= \begin{bmatrix} \boldsymbol{y}^T - \boldsymbol{a}^T A^T(A^T\,\mathrm{diag}(\boldsymbol{s})^{-2}A)^{-1}\mathrm{diag}(\boldsymbol{s})^{-2}\beta/r & \beta/r \end{bmatrix}^T, \\
\boldsymbol{s} &= \begin{bmatrix} \boldsymbol{s}^T + \boldsymbol{a}^T(A^T\,\mathrm{diag}(\boldsymbol{s})^{-2}A)^{-1}A\beta/r & \beta r \end{bmatrix}^T, \\
A &= \begin{bmatrix} A & \boldsymbol{a} \end{bmatrix}.
\end{aligned}
$$

The triple ($\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{s}$) will be close to the true AC, and is guaranteed to be inside the updated set P. We then employ a primal-dual Newton procedure to iterate from our estimate inside P to the true AC [83]. Using the estimated AC, the Newton procedure will quickly converge to the true analytic center, using the convergence criterion $\|\mathrm{diag}(\boldsymbol{y})\boldsymbol{s} - \mathbf{1}\| < \epsilon$ for some

small ϵ. The triple ($\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{s}$) is iteratively updated using the following equations:

$$\boldsymbol{x} := \boldsymbol{x} - (A\,\mathrm{diag}(\boldsymbol{y})\mathrm{diag}(\boldsymbol{s})^{-1}A^T)^{-1}A\,\mathrm{diag}(\boldsymbol{s})^{-1}\mathbf{1},$$

$$\boldsymbol{y} := \boldsymbol{y} - \mathrm{diag}(\boldsymbol{s})^{-1}(I - \mathrm{diag}(\boldsymbol{y})A^T(A\,\mathrm{diag}(\boldsymbol{y})\mathrm{diag}(\boldsymbol{s})^{-1}A^T)^{-1}A\,\mathrm{diag}(\boldsymbol{s})^{-1})(\mathrm{diag}(\boldsymbol{y})\boldsymbol{s} - \mathbf{1}),$$

$$\boldsymbol{s} := \boldsymbol{s} + A^T(A\,\mathrm{diag}(\boldsymbol{y})\mathrm{diag}(\boldsymbol{s})^{-1}A^T)^{-1}A\,\mathrm{diag}(\boldsymbol{s})^{-1}\mathbf{1}.$$

As the analytic center is defined by the halfspaces that make up a convex set, and not the convex set itself, parallel halfspaces are by definition redundant and will push the analytic center away from the CG.

Note that the intercept b is not required for the above calculations, as it is specified such that the halfspace defined by the normal vector $\boldsymbol{a}$ is shifted so that the hyperplane $\{\boldsymbol{x} \mid A^T\boldsymbol{x} = \boldsymbol{b}\}$ passes through the analytic center. Since our intercept is predefined, a common occurrence in the ACCP is that the same $\boldsymbol{a}$ is returned by the oracle multiple times with decreasing intercepts. In this situation, redundant columns are not added to A so that the AC stays as close as possible to the CG.

6.4.3 Adding New Halfspaces

Specifically, the IQC oracle is used to determine frequencies ω which violate the IQC inequality such that $F(e^{j\omega}, \boldsymbol{x}_c) \not\succ 0$ for the current candidate solution $\boldsymbol{x}_c$. Recall that $F(z)$ in (4.7) is a function of $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, S, and z, and is written parameterized by z. While $\mathcal{A}$, $\mathcal{B}$,

$\mathcal{C}$, and $\mathcal{D}$ are fixed, S, which is in turn a function of γ, is not. Since the decision variables defining S (including γ) are defined by $\boldsymbol{x}$, we additionally parameterize (4.7) by $\boldsymbol{x}$ and write it as $F(z, \boldsymbol{x})$. If the oracle determines that $F(e^{j\omega}, \boldsymbol{x}_c) \prec 0$, the candidate solution $\boldsymbol{x}_c$ is feasible and satisfies the IQC inequality (4.7). The decision variable corresponding to the performance value (chosen as the last term in $\boldsymbol{x}_c$ for this work) is thus an upper bound on the worst-case ℓ_2-gain performance level of the system. The upper bound is then updated to be equal to the minimum of the current upper bound and the candidate performance value. A new halfspace is added by appending the column vector $\boldsymbol{a} = \begin{bmatrix}\mathbf{0}_{n_x-1}^T & 1\end{bmatrix}^T$ to the matrix A to restrict the upper bound on worst case performance to be lower than the current candidate value.

Given a frequency ω such that $F(e^{j\omega}, \boldsymbol{x}_c) \not\succ 0$, there exists an eigenvalue λ of $F(e^{j\omega}, \boldsymbol{x}_c)$ such that $\lambda \geq 0$. It follows that the eigenvector $\boldsymbol{u}$ corresponding to λ can be used to write $\boldsymbol{u}^T F(e^{j\omega}, \boldsymbol{x}_c)\boldsymbol{u} \geq 0$. The normal vector of the halfspace corresponding to the constraint $\boldsymbol{u}^T F(e^{j\omega}, \boldsymbol{x})\boldsymbol{u} < 0$ is determined as

$$\boldsymbol{a} = \frac{\partial \boldsymbol{u}^T F(e^{j\omega}, \boldsymbol{x})\boldsymbol{u}}{\partial \boldsymbol{x}}.$$

After $\boldsymbol{a}$ is normalized, the AC estimate and primal-dual Newton algorithm can be applied as in Section 6.4.2.

The lower bound can now be updated by minimizing the performance value subject to the current set of halfspace constraints, which can be solved as the linear program (6.4).

6.4.4 Cutting Plane Algorithm Pseudocode

We now present the pseudocode for the ACCP algorithm which was briefly discussed in Section 6.3, and expanded upon in Sections 6.4.1-6.4.3. The notation $\mathcal{O}$ refers to the IQC oracle presented in Section 6.2.

Algorithm 2 ACCP Algorithm

Require: $n_x,\ r_0,\ \epsilon_0$

Assign $A = [I_{n_x}\ \ -I_{n_x}]$ ▷ Initialize the constraint matrix

Assign $\boldsymbol{x}_c = \mathbf{0}_{n_x}$ ▷ Initialize the analytic center

Assign $\boldsymbol{y} = \mathbf{1}_{2n_x}\frac{1}{r_0}$ ▷ Initialize the primal solution

Assign $\boldsymbol{s} = \mathbf{1}_{2n_x}r_0 - A^T\boldsymbol{x}_c$ ▷ Initialize the slack values

Assign $L = \min \boldsymbol{x}(n_x)$, subject to $A^T\boldsymbol{x} \leq \boldsymbol{s} + A^T\boldsymbol{x}_c$ ▷ Initialize lower bound

Assign $U = \infty$ ▷ Initialize upper bound

$\epsilon = 1$ ▷ Initialize convergence criteria

while $\epsilon > \epsilon_0$ **do**

$\boldsymbol{\omega} = \mathcal{O}(\boldsymbol{x})$ ▷ Call IQC oracle

if $\boldsymbol{\omega} = \varnothing$ **then** ▷ $\boldsymbol{x}_c$ is a feasible solution

Set $U := \min(U,\ \boldsymbol{x}_c(n_x))$ ▷ Update upper bound

Assign $\boldsymbol{a} = \left[\mathbf{0}_{n_x-1}^T\ 1\right]^T$ ▷ Constrain performance value

else ▷ $\boldsymbol{x}_c$ is not a feasible solution

Assign $n_\omega = \text{size}(\boldsymbol{\omega})$ ▷ Number of frequency pairs

if $n_\omega = 1$ **then**

Assign $\boldsymbol{\omega}_m = 0$ ▷ Critical frequency midpoint

else

Assign $\boldsymbol{\omega}_m = \left[0 \ \frac{1}{2}(\omega(2, 3, \ldots, n_\omega) - \omega(1, 2, \ldots, n_\omega - 1))\right]$

end if

for $i = 1, 2, \ldots, n_\omega$ **do**

Assign $[\Lambda, \ \boldsymbol{U}] = \sigma\left(F(\boldsymbol{\omega}_m(i), \boldsymbol{x}_c)\right)$ ▷ Calculate eigenvalues and eigenvectors

Assign $[\lambda_{max}, j] = \max(\Lambda)$ ▷ Find largest eigenvalue and location

Assign $\lambda_\omega(i) = \lambda_{max}$

Assign $\boldsymbol{u}(:, i) = \boldsymbol{U}(:, j)$ ▷ Store eigenvector for the frequency $\boldsymbol{\omega}_m(i)$

end for

Assign $[\lambda_{max}, j] = \max(\lambda_\omega)$ ▷ Find largest eigenvalue and eigenvector

Assign $\boldsymbol{a} = \partial \boldsymbol{u}(:, j)^T F(\boldsymbol{\omega}_m(j), \boldsymbol{x}) \boldsymbol{u}(:, j) / \partial \boldsymbol{x}$ ▷ Find halfspace constraint

Set $\boldsymbol{a} := \boldsymbol{a} / \|\boldsymbol{a}\|$ ▷ Normalize halfspace

end if

Call dual-Newton algorithm to update $\boldsymbol{x}_c$, $\boldsymbol{y}$, and $\boldsymbol{s}$ given $\boldsymbol{a}$.

Set $A := [A \ \boldsymbol{a}]$

Assign $L_0 = \min \boldsymbol{x}(n_x)$, subject to $A^T \boldsymbol{x} \leq \boldsymbol{s} + A^T \boldsymbol{x}_c$ ▷ Calculate lower bound

Set $L := \max(L, \ L_0)$ ▷ Update lower bound

Set $\epsilon := \sqrt{U} - \sqrt{L}$ ▷ Update convergence criteria

end while

Assign $\gamma = \sqrt{U}$ ▷ Upper bound on worst-case performance

Return: γ

6.5 Improving Oracle Robustness

In this section, we attempt to improve the robustness of eigenvalue computations in the manner proposed for continuous-time $\mathcal{H}_\infty$ norm oracles in [82]. As a result of this process, not only are eigenvalue calculations likely to be more robust, but the resulting eigenvalue problem is specially structured such that a robust eigensolver can take advantage of symmetries in the matrix structure. In this work, we specifically deal with real-valued dynamic systems, although extensions to the complex case are possible.

Extensions to the discrete-time $\mathcal{H}_\infty$ norm case are also discussed in [82]. What follows is a complete methodology for the discrete-time IQC oracle. Besides differences in equations and structure, such as the coupling due to the $\mathcal{Q}$ matrix, we prove the discrete-time version of the matrix extension lemma, which is necessary to formulate a more robust oracle. For an $\mathcal{H}_\infty$ norm oracle, it is only required to determine feasibility for a given γ_0^2 value as the system under analysis is only a function of frequency and γ^2. The IQC oracle, on the other hand, is a function of $S(\gamma^2)$, and the oracle must return a list of the critical frequencies ω_i where $\lambda_{\max}(F(e^{j\omega_i})) = 0$ so that appropriate candidate solutions can be generated. To accommodate this, equations have been provided so that the critical frequencies can be recovered from the robust oracle solution. As will be shown later, we can use our choice of oracle boundary condition to avoid calculating infinite eigenvalues, which are a natural result of the robust oracle.

We first convert the linear matrix pencil $H - zN$ to the even dimensioned pencil $\tilde{H} - z\tilde{N}$ so that it can later be partitioned into equivalently dimensioned blocks, and certain symmetries can be maintained. If $n_{\mathcal{B}}$ is odd, we simply add a single imaginary input to the system by defining $\tilde{\mathcal{B}} = [\mathcal{B}\ \mathbf{0}]$ and $\tilde{\mathcal{D}} = [\mathcal{D}\ \mathbf{0}]$. Calculating $\tilde{\mathcal{F}}$ and $\tilde{\mathcal{R}}$ as in (4.14) with $\tilde{\mathcal{B}}$ and $\tilde{\mathcal{D}}$, we can partition $\tilde{\mathcal{B}}$, $\tilde{\mathcal{F}}$, and $\tilde{\mathcal{R}}$ into equally dimensioned parts as

$$\tilde{\mathcal{B}} = \left[\begin{array}{c:c} \mathcal{B}_1 & \mathcal{B}_2 \end{array}\right], \tilde{\mathcal{F}} = \left[\begin{array}{c:c} \mathcal{F}_1 & \mathcal{F}_2 \end{array}\right], \tilde{R} = \left[\begin{array}{c:c} \mathcal{R}_{11} & \mathcal{R}_{12} \\ \hdashline \mathcal{R}_{12}^T & \mathcal{R}_{22} \end{array}\right]. \tag{6.5}$$

The even pencil $\tilde{H} - z\tilde{N}$ is thus defined as

$$\begin{bmatrix} zI - \mathcal{A} & 0 \\ -\mathcal{Q} & I - z\mathcal{A}^T \end{bmatrix} - \begin{bmatrix} \tilde{\mathcal{B}} \\ \tilde{\mathcal{F}} \end{bmatrix} \tilde{\mathcal{R}}^{-1} \begin{bmatrix} -\tilde{\mathcal{F}}^T & -z\tilde{\mathcal{B}}^T \end{bmatrix}. \tag{6.6}$$

The added zeros in (6.6) simply add an infinite eigenvalue to those of (6.3). If (6.3) already contained an infinite eigenvalue, we have $\sigma(\tilde{H},\, \tilde{N}) = \sigma(H,\, N)$. For the more general case, we can relate their spectrums as $\sigma(\tilde{H},\, \tilde{N}) = \sigma(H,\, N) \cup \{\infty\}$.

As the matrix multiplications and inverses present in (6.3) may be ill-conditioned, we next try to reduce the number of such operations required to calculate $\sigma(H, N)$ by extending the pencil (6.6).

Lemma 2. *A matrix pencil of the form $H - zN = A - BD^{-1}C - z(E - BD^{-1}F)$, with real-valued, appropriately dimensioned matrices A, B, C, D, E, and F, invertible D, and spectrum $\sigma(H, N)$, can be extended to the pencil $H_E - zN_E$ with $\sigma(H_E, N_E) = \sigma(H, N) \cup \{\infty\}$,*

where $H_E - zN_E$ *is given by*

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} - z \begin{bmatrix} E & 0 \\ F & 0 \end{bmatrix}.$$

Proof. The proof follows a similar argument to the one used in the proof of [85, Lemma 4.3]. Given $z \in \sigma(H, N)$, there exists a nonzero vector $\boldsymbol{x}_1$ such that $(A - BD^{-1}C)\boldsymbol{x}_1 = z(E - BD^{-1}F)\boldsymbol{x}_1$, or equivalently, $(A+BD^{-1}(zF-C))\boldsymbol{x}_1 = zE\boldsymbol{x}_1$. Choosing $C\boldsymbol{x}_1+D\boldsymbol{x}_2 = zF\boldsymbol{x}_1$, the vector $\boldsymbol{x}_2$ can be solved for as $\boldsymbol{x}_2 = D^{-1}(zF - C)\boldsymbol{x}_1$. Substituting this into the original system yields $A\boldsymbol{x}_1 + B\boldsymbol{x}_2 = zE\boldsymbol{x}_1$. Since $\boldsymbol{x}_1$ is nonzero, $\boldsymbol{x}_0 = [\boldsymbol{x}_1^T \ \boldsymbol{x}_2^T]^T$ is by definition nonzero, and we have

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = z \begin{bmatrix} E & 0 \\ F & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix}. \tag{6.7}$$

Note that we have introduced a number of infinite eigenvalues in (6.7) due to the structure of N_E, and thus $\sigma(H_E, N_E) \subseteq \sigma(H, N) \cup \{\infty\}$.

Conversely, given $z \in \sigma(H_E, N_E)$, we have $A\boldsymbol{x}_1+B\boldsymbol{x}_2 = zE\boldsymbol{x}_1$ and $C\boldsymbol{x}_1+D\boldsymbol{x}_2 = zF\boldsymbol{x}_1$, with $\boldsymbol{x}_0 = [\boldsymbol{x}_1^T,\ \boldsymbol{x}_2^T]^T$ nonzero. Solving the second equation for $\boldsymbol{x}_2$ and plugging it into the first once again yields $(A + BD^{-1}(zF - C))\boldsymbol{x}_1 = zE\boldsymbol{x}_1$. It remains to show that $\boldsymbol{x}_1$ is nonzero if $\boldsymbol{x}_0$ is nonzero. Suppose $\boldsymbol{x}_1$ is the zero vector. $\boldsymbol{x}_2 = D^{-1}(zF - C)\boldsymbol{x}_1$ would therefore be zero, making $[\boldsymbol{x}_1^T,\ \boldsymbol{x}_2^T]^T$ zero. Since $[\boldsymbol{x}_1^T,\ \boldsymbol{x}_2^T]^T$ is assumed to be nonzero, this is a contradiction and $\boldsymbol{x}_1$ must be nonzero if $\boldsymbol{x}_0$ is nonzero. Therefore, we have $\sigma(H, N) \subseteq \sigma(H_E, N_E)$. In conclusion, we have proved that $\sigma(H_E, N_E) = \sigma(H, N) \cup \{\infty\}$. □

By applying Lemma 2 to (6.6), we can define the extended matrix pencil as

$$H_E - zN_E = \left[\begin{array}{cc:c} -\mathcal{A} & 0 & \tilde{\mathcal{B}} \\ -\mathcal{Q} & I & \tilde{F} \\ \hdashline -\tilde{\mathcal{F}}^T & 0 & \tilde{\mathcal{R}} \end{array}\right] - z \left[\begin{array}{cc:c} -I & 0 & 0 \\ 0 & \mathcal{A}^T & 0 \\ \hdashline 0 & \tilde{\mathcal{B}}^T & 0 \end{array}\right]. \tag{6.8}$$

Furthermore, we have the spectrum $\sigma(H_E,\, N_E) = \sigma(\tilde{H},\, \tilde{N}) \cup \{\infty\} = \sigma(H,\, N) \cup \{\infty\}$. Note that by implementing this step, we have removed all matrix inversions and multiplications from (6.3), improving the robustness of our eigenvalue calculation.

Swapping the first and second columns and taking the negative of the third row of $H_E - zN_E$ yields the D-type matrix pencil $H_D - zN_D$ [39], where $\sigma(H_D, N_D) = \sigma(H_E, N_E)$; namely,

$$H_D - zN_D = \left[\begin{array}{c:cc} 0 & -\mathcal{A} & \tilde{\mathcal{B}} \\ \hdashline I & -\mathcal{Q} & \tilde{\mathcal{F}} \\ 0 & \tilde{\mathcal{F}}^T & -\tilde{\mathcal{R}} \end{array}\right] - z \left[\begin{array}{c:cc} 0 & -I & 0 \\ \hdashline \mathcal{A}^T & 0 & 0 \\ -\tilde{\mathcal{B}}^T & 0 & 0 \end{array}\right]. \tag{6.9}$$

Since all we have done is swap rows and columns of the pencil, clearly $\sigma(H_D,\ N_D) = \sigma(H_E,\ N_E)$.

As $H_D - zN_D$ is a D-type pencil, we can say that, given $z \in \sigma(H_D,\ N_D)$, then the inverse of its complex conjugate $\bar{z}^{-1} \in \sigma(H_D,\ N_D)$, or, the pencil's eigenvalues have symmetry with respect to the unit circle.

As presented in [39], we perform a Cayley transformation, $\mathbf{c}$, followed by a "drop/add"

transformation to obtain a C-type matrix pencil $H_C - \lambda N_C$ from the D-type pencil $H_D - zN_D$ given in (6.9), where

$$H_C - \lambda N_C = \left[\begin{array}{c|cc} 0 & I-\mathcal{A} & \tilde{\mathcal{B}} \\ \hline I-\mathcal{A}^T & -\mathcal{Q} & \tilde{\mathcal{F}} \\ \tilde{\mathcal{B}}^T & \tilde{\mathcal{F}}^T & -\tilde{\mathcal{R}} \end{array}\right] - \lambda \left[\begin{array}{c|cc} 0 & -I-\mathcal{A} & \tilde{\mathcal{B}} \\ \hline I+\mathcal{A}^T & 0 & 0 \\ -\tilde{\mathcal{B}}^T & 0 & 0 \end{array}\right]. \tag{6.10}$$

The C-type matrix pencil has eigen-symmetry with respect to the imaginary axis, or, if $\lambda \in \sigma(H_C,\ N_C)$, then $-\bar{\lambda} \in \sigma(H_C,\ N_C)$. The use of "$\lambda$" instead of "$z$" in (6.13) is done merely to indicate that, roughly speaking, the composite transformation maps discrete-time eigenvalues into their continuous-time counterparts. Before giving the next result, we provide the definition of the Cayley transformation, namely, $\mathbf{c} : \mathbb{C} \cup \{\infty\} \to \mathbb{C} \cup \{\infty\}$ is defined as

$$\mathbf{c}(z) = (z-1)(z+1)^{-1},\ \mathbf{c}(-1) = \infty,\ \mathbf{c}(\infty) = 1. \tag{6.11}$$

This transformation maps all points in $\mathbb{D}$ to the imaginary axis, with -1 mapped to infinity. Specifically, the points on the unit circle, $z = e^{j\omega}$ for all $\omega \in [-\pi,\ \pi]$, are mapped as

$$\mathbf{c}(e^{j\omega}) = \frac{\cos\omega + j\sin\omega - 1}{\cos\omega + j\sin\omega + 1} = j\frac{\sin\omega}{1+\cos\omega} = j\tan\frac{\omega}{2}. \tag{6.12}$$

For convenience, we introduce the notation $\mathbf{c}(\sigma(H,N)) = \{\lambda \in \mathbb{C} \,|\, \lambda = \mathbf{c}(z) \text{ for all } z \in \sigma(H,N)\}$. The following result is based on Theorems 15 and 16 from [39].

Lemma 3. *Given $H_D - zN_D$ and $H_C - \lambda N_C$, as defined in (6.9) and (6.13), respectively,*

$H_C - \lambda N_C$ is regular if and only if $H_D - zN_D$ is regular. Furthermore, $\sigma(H_C, N_C) = \mathbf{c}(\sigma(H_D, N_D)) \cup \{\infty\}$.

Partitioning some matrices as in (6.5), the C-type pencil $H_C - \lambda N_C$ can be expressed as

$$H_C - \lambda N_C = \left[\begin{array}{c|cc} 0 & I - \mathcal{A} & \tilde{\mathcal{B}} \\ \hline I - \mathcal{A}^T & -\mathcal{Q} & \tilde{\mathcal{F}} \\ \tilde{\mathcal{B}}^T & \tilde{\mathcal{F}}^T & -\tilde{\mathcal{R}} \end{array}\right] - \lambda \left[\begin{array}{c|cc} 0 & -I - \mathcal{A} & \tilde{\mathcal{B}} \\ \hline I + \mathcal{A}^T & 0 & 0 \\ -\tilde{\mathcal{B}}^T & 0 & 0 \end{array}\right]. \tag{6.13}$$

6.5.1 Skew-Hamiltonian/Hamiltonian Pencil

Dividing the input channels of the C-Type pencil (6.13) as in (6.5) yields the identical pencil

$$H_C - \lambda N_C = \left[\begin{array}{c|ccc} 0 & I - \mathcal{A} & \mathcal{B}_1 & \mathcal{B}_2 \\ \hline I - \mathcal{A}^T & -\mathcal{Q} & \mathcal{F}_1 & \mathcal{F}_2 \\ \mathcal{B}_1^T & \mathcal{F}_1^T & -\mathcal{R}_{11} & -\mathcal{R}_{12} \\ \mathcal{B}_2^T & \mathcal{F}_2^T & -\mathcal{R}_{12}^T & -\mathcal{R}_{22} \end{array}\right] - \lambda \left[\begin{array}{c|ccc} 0 & -I - \mathcal{A} & \mathcal{B}_1 & \mathcal{B}_2 \\ \hline I + \mathcal{A}^T & 0 & 0 & 0 \\ -\mathcal{B}_1^T & 0 & 0 & 0 \\ -\mathcal{B}_2^T & 0 & 0 & 0 \end{array}\right].$$

The pencil $H_C - \lambda N_C$ can be rearranged such that both H_C and N_C are partitioned into four equivalently dimensioned blocks. Rearranging the pencil further (through multiplications by unitary matrices) results in symmetries between blocks and yields a skew-Hamiltonian/Hamiltonian matrix pencil.

Following a sequence similar to [85], we swap the 1st and 2nd row, the 2nd and 3rd row,

the 2nd and 4th row, the 2nd and 3rd column, take the negative of rows 1 and 2 and finally take the transpose of both sides to yield the skew-Hamiltonian/Hamiltonian matrix pencil $\mathcal{H} - \lambda\mathcal{N}$ defined as

$$\mathcal{H}-\lambda\mathcal{N}=\left[\begin{array}{cc:cc} \mathcal{A}-I & -\mathcal{B}_2 & 0 & \mathcal{B}_1 \\ -\mathcal{F}_1^T & \mathcal{R}_{12} & \mathcal{B}_1^T & -\mathcal{R}_{11} \\ \hdashline \mathcal{Q} & -\mathcal{F}_2 & I-\mathcal{A}^T & \mathcal{F}_1 \\ -\mathcal{F}_2^T & \mathcal{R}_{22} & \mathcal{B}_2^T & -\mathcal{R}_{12}^T \end{array}\right] - \lambda \left[\begin{array}{cc:cc} -I-\mathcal{A} & \mathcal{B}_2 & 0 & -\mathcal{B}_1 \\ 0 & 0 & \mathcal{B}_1^T & 0 \\ \hdashline 0 & 0 & -I-\mathcal{A}^T & 0 \\ 0 & 0 & \mathcal{B}_2^T & 0 \end{array}\right]. \quad (6.14)$$

The spectrum of the pencil is invariant under the applied transformations, and so, $\sigma(\mathcal{H},\mathcal{N}) = \sigma(H_C, N_C)$. Since $\sigma(H_C, N_C) = \mathbf{c}(\sigma(H_D, N_D)) \cup \{\infty\}$ and $\sigma(H_D, N_D) = \sigma(H_E, N_E) = \sigma(H, N)\cup\{\infty\}$, we can then relate the eigenvalues of the preceding skew-Hamiltonian/Hamiltonian pencil to the original pencil (6.3) as $\sigma(\mathcal{H},\mathcal{N}) = \mathbf{c}(\sigma(H, N)) \cup \{1, \infty\}$.

By setting the boundary condition in Theorem 2 to $F(-1) \prec 0$, we ensure that no unit circle eigenvalues of $H - zN$ will get mapped to ∞ by (6.11). Thus, in addition to possible infinite eigenvalues that are already in $\sigma(H, N)$, all infinite eigenvalues of $\mathcal{H} - \lambda\mathcal{N}$ can be attributed to the pencil extension and artificial input, and are not of interest to us. Only the purely imaginary eigenvalues in $\sigma(\mathcal{H},\mathcal{N})$ will correspond to critical frequencies. Using (6.12), the critical frequencies can be calculated from the purely imaginary eigenvalues λ_i (if any) as $\omega_i = 2\arctan(-j\lambda_i)$.

The skew-Hamiltonain/Hamiltonian pencil $\mathcal{H} - \lambda\mathcal{N}$ can be formed immediately given the system matrices $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$ and candidate IQC multiplier matrix $S_0(\gamma_0^2)$, and thus cor-

responds to very little computational cost. Besides improving robustness by eliminating unnecessary matrix multiplications and inverses, the skew-Hamiltonian/Hamiltonian eigenvalue problem can be solved by applying a structure exploiting eigensolver using techniques such as generalized symplectic URV (GSURV) decomposition [37]. The `SLICOT` toolbox, for example, provides a robustly implemented GSURV decomposition solver for real-valued skew-Hamiltonian/Hamiltonian matrix pencils [86]. Details on the version of the GSURV decomposition algorithm implemented for this work can be found in Section 6.6.

The GSURV decomposition algorithm takes advantage of the symmetries in $\mathcal{H} - \lambda\mathcal{N}$. Due to the skew-Hamiltonian/Hamiltonian structure of $\mathcal{H} - \lambda\mathcal{N}$, we know that $\sigma(\mathcal{H}, \mathcal{N})$ has symmetry with respect to the imaginary axis, and its eigenvalues will occur in quadruplets as $\lambda,\ \bar{\lambda},\ -\lambda,\ -\bar{\lambda}$. This can be easily shown using the definitions of Hamiltonian and skew-Hamiltonian matrices. Consider an eigenvalue λ and an eigenvector $\boldsymbol{x}$ of the pencil $\mathcal{H} - \lambda\mathcal{N}$ such that $(\mathcal{H} - \lambda\mathcal{N})\boldsymbol{x} = \mathbf{0}$. Taking the adjoint, we have

$$\mathbf{0} = \boldsymbol{x}^*(\mathcal{H}^T - \bar{\lambda}\mathcal{N}^T) = \boldsymbol{x}^*(-\mathcal{J}^T\mathcal{H}\mathcal{J} - \bar{\lambda}\mathcal{J}^T\mathcal{N}\mathcal{J}) = -\boldsymbol{x}^*\mathcal{J}^T(\mathcal{H} + \bar{\lambda}\mathcal{N})\mathcal{J}.$$

Thus $-\bar{\lambda} \in \sigma(\mathcal{H}, \mathcal{N})$ if and only if $\lambda \in \sigma(\mathcal{H}, \mathcal{N})$. Since $\mathcal{H}$ and $\mathcal{N}$ are real-valued matrices in this case, it follows that $-\lambda \in \sigma(\mathcal{H}, \mathcal{N})$ and $\bar{\lambda} \in \sigma(\mathcal{H}, \mathcal{N})$ for finite eigenvalues. Finally, we can state that

$$\sigma(\mathcal{H}, \mathcal{N}) = \sigma(-\mathcal{H}, \mathcal{N}). \tag{6.15}$$

Since any purely imaginary eigenvalues of $\mathcal{H} - \lambda\mathcal{N}$ will occur in conjugate pairs, we can

conclude that the critical frequencies in the range $\omega \in [-\pi,\ \pi]$ will be symmetric about 0.

6.5.2 Decision Variable Scaling

When implementing the ACCP algorithm, it is possible that the decision variables in S will have a very large magnitude. As such, this has the potential to negatively affect the well-posedness of the resulting eigenvalue problem. Fortunately, we can exploit the structure of the matrix pencil in order to inexpensively scale the decision variables in S.

Looking at (6.13), the only nonzero terms of the northwest and southeast blocks of H_C and N_C is the block

$$\begin{bmatrix} -\mathcal{Q} & \mathcal{F} \\ \mathcal{F}^T & -\mathcal{R} \end{bmatrix} = \begin{bmatrix} -\mathcal{C}^T \\ \mathcal{D}^T \end{bmatrix} S \begin{bmatrix} \mathcal{C} & -\mathcal{D} \end{bmatrix},$$

in the southeast part of H_C. Defining the matrix

$$\mathscr{R} = \begin{bmatrix} I\frac{\sqrt{\rho}}{\rho} & 0 \\ 0 & I\sqrt{\rho} \end{bmatrix},$$

where ρ is any positive constant and parameterizing H_C by S, we have $\mathscr{R}N_C\mathscr{R} = N_C$ and $\mathscr{R}H_C(S)\mathscr{R} = H_C(\rho S)$. We can relate the spectrums of the pencils $H_C(S) - \lambda N_C$ and $H_C(\rho S) - \lambda N_C$ as

$$\sigma(H_C(S),\, N_C) = \sigma(\mathscr{R}H_C(S)\mathscr{R},\, \mathscr{R}N_C\mathscr{R}) = \sigma(H_C(\rho S),\, N_C).$$

Thus, we can scale the matrix S by any positive constant ρ, and it follows that $\sigma(\mathcal{H}(S), \mathcal{N}) = \sigma(\mathcal{H}(\rho S), \mathcal{N})$ for the skew-Hamiltonian/Hamiltonian pencil. For this work, we will use $\rho = 1/\|S\|$, so that $\|\rho S\| = 1$.

6.6 Robust Structure Exploiting Eigenvalue Solver

As a full pseudocode for the GSURV decomposition algorithm is not readily available in the literature, it is outlined in detail here. To improve the clarity and readability of this section, the matrices in the oracle have been lumped together and simplified. The skew-Hamiltonian/Hamiltonian pencil (6.14) has the following structure:

$$\mathcal{H} - \lambda\mathcal{N} = \begin{bmatrix} \mathcal{E} & \mathcal{F} \\ \mathcal{G} & -\mathcal{E}^T \end{bmatrix} - \lambda \begin{bmatrix} \mathcal{K} & \mathcal{L} \\ 0 & \mathcal{K}^T \end{bmatrix},$$

where $\mathcal{H}, \mathcal{N} \in \mathbb{R}^{m\times m}$, $\mathcal{F}, \mathcal{G} \in \mathbb{S}^n$, and $\mathcal{L}$ is skew-symmetric $\in \mathbb{R}^{n\times n}$.

Recall that by ensuring $\omega = \pi$ is not a violated frequency, the linear matrix pencil in the oracle will have no corresponding infinite eigenvalues that were mapped from -1 to infinity. Thus, the only eigenvalues of interest will have exactly zero real part. In order to avoid false detections of purely imaginary eigenvalues, a structure exploiting eigenvalue solver for real-valued skew-Hamiltonian/Hamiltonian matrix pencils will be used to deflate the matrix subspaces and calculate eigenvalues with exactly zero real part [37].

In order to find the eigenvalues, we will need to compute a particular decomposition to put

the matrices of interest into skew-triangular and skew-Hessenberg form using orthogonal matrices $\mathcal{Q}_1$ and $\mathcal{Q}_2$ as

$$\begin{aligned} \mathcal{Q}_1^T \mathcal{N} \mathcal{J} \mathcal{Q}_1 \mathcal{J}^T &= \begin{bmatrix} N_{11} & N_{12} \\ 0 & N_{11}^T \end{bmatrix}, \\ \mathcal{Z} = \mathcal{J} \mathcal{Q}_2^T \mathcal{J}^T \mathcal{N} \mathcal{Q}_2 &= \begin{bmatrix} Z_{11} & Z_{12} \\ 0 & Z_{11}^T \end{bmatrix}, \\ \mathcal{Q}_1^T \mathcal{H} \mathcal{Q}_2 &= \begin{bmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{bmatrix}, \end{aligned} \tag{6.16}$$

where N_{11}, Z_{11}, and H_{11} are upper-triangular, and H_{22}^T is upper quasi-triangular. Theorem 3 states that such decompositions will always be possible to find.

Theorem 3. *Generalized Symplectic URV Decomposition*

Given real-valued skew-Hamiltonian and Hamiltonian matrices $\mathcal{H}$ and $\mathcal{N}$, there exists orthogonal matrices $\mathcal{Q}_1$ and $\mathcal{Q}_2$ such that the decomposition (6.16) can be formed where N_{11}, Z_{11}, and H_{11} are upper-triangular, and H_{22}^T is upper-Hessenberg.

Proof. The proof is presented by construction in Section 6.6.2. □

6.6.1 Double Sized Skew-Hamiltonian/Hamiltonian Pencil

Using (6.16) and the Hamiltonian structure of $\mathcal{H}$, we can arrive at the identity

$$\mathcal{J}\mathcal{Q}_2^T\mathcal{J}^T\mathcal{H}\mathcal{J}\mathcal{Q}_1\mathcal{J}^T = \begin{bmatrix} -H_{22}^T & H_{12}^T \\ 0 & -H_{11}^T \end{bmatrix}. \tag{6.17}$$

To start, from (6.16), we can directly obtain the relation

$$\mathcal{Q}_2^T\mathcal{H}^T\mathcal{Q}_1 = \begin{bmatrix} H_{11}^T & 0 \\ H_{12}^T & H_{22}^T \end{bmatrix}. \tag{6.18}$$

From the definition of a Hamiltonain matrix, we have $\mathcal{H}\mathcal{J} = \mathcal{J}^T\mathcal{H}^T$. We can easily solve for $\mathcal{H}^T$ using the fact that $\mathcal{J}\mathcal{J} = \mathcal{J}^T\mathcal{J}^T = -I$ as $\mathcal{H}^T = -\mathcal{J}^T\mathcal{H}\mathcal{J}$. Substituting this into (6.18) and pre and post multiplying by $\mathcal{J}$ and $\mathcal{J}^T$, respectively, yields the result in (6.17).

We next define the composition of a symmetric matrix $\mathcal{P}$ and skew-symmetric matrix $\mathcal{Y}$ as

$$\mathcal{Y} = \frac{\sqrt{2}}{2}\begin{bmatrix} I_m & I_m \\ -I_m & I_m \end{bmatrix}, \ \mathcal{P} = \begin{bmatrix} I_n & 0 & 0 & 0 \\ 0 & 0 & I_n & 0 \\ 0 & I_n & 0 & 0 \\ 0 & 0 & 0 & I_n \end{bmatrix}, \ \mathcal{X} = \mathcal{Y}\mathcal{P} = \frac{\sqrt{2}}{2}\begin{bmatrix} I_n & I_n & 0 & 0 \\ 0 & 0 & I_n & I_n \\ -I_n & I_n & 0 & 0 \\ 0 & 0 & -I_n & I_n \end{bmatrix},$$

where $\mathcal{X}$ is orthogonal.

Following [37], we define the large $2m \times 2m$ matrices

$$\tilde{\mathcal{B}}_{\mathcal{H}} = \begin{bmatrix} \mathcal{H} & 0 \\ 0 & -\mathcal{H} \end{bmatrix},$$
$$\tilde{\mathcal{B}}_{\mathcal{N}} = \begin{bmatrix} \mathcal{N} & 0 \\ 0 & \mathcal{N} \end{bmatrix}.$$

Clearly, the pencil $\tilde{\mathcal{B}}_{\mathcal{H}} - \lambda\tilde{\mathcal{B}}_{\mathcal{N}}$ has the spectrum $\sigma(\tilde{\mathcal{B}}_{\mathcal{H}}, \tilde{\mathcal{B}}_{\mathcal{N}}) = \sigma(\mathcal{H}, \mathcal{N}) \cup \sigma(-\mathcal{H}, \mathcal{N})$. From (6.15), we can conclude that $\sigma(\tilde{\mathcal{B}}_{\mathcal{H}}, \tilde{\mathcal{B}}_{\mathcal{N}}) = \sigma(\mathcal{H}, \mathcal{N})$.

Next we define the pencil $\hat{\mathcal{B}}_{\mathcal{H}} - \lambda\hat{\mathcal{B}}_{\mathcal{N}} = \mathcal{Y}^T(\tilde{\mathcal{B}}_{\mathcal{H}} - \lambda\tilde{\mathcal{B}}_{\mathcal{N}})\mathcal{Y}$, where it is clear that $\sigma(\hat{\mathcal{B}}_{\mathcal{H}}, \hat{\mathcal{B}}_{\mathcal{N}}) = \sigma(\tilde{\mathcal{B}}_{\mathcal{H}}, \tilde{\mathcal{B}}_{\mathcal{N}})$. $\hat{\mathcal{B}}_{\mathcal{H}}$ and $\hat{\mathcal{B}}_{\mathcal{N}}$ can be expanded as

$$\hat{\mathcal{B}}_{\mathcal{H}} = \mathcal{Y}^T\tilde{\mathcal{B}}_H\mathcal{Y} = \begin{bmatrix} 0 & \mathcal{H} \\ \mathcal{H} & 0 \end{bmatrix}, \tag{6.19}$$
$$\hat{\mathcal{B}}_{\mathcal{N}} = \mathcal{Y}^T\tilde{\mathcal{B}}_{\mathcal{N}}\mathcal{Y} = \tilde{\mathcal{B}}_{\mathcal{N}}.$$

Additionally, we define the pencil $\mathcal{B}_{\mathcal{H}} - \lambda\mathcal{B}_{\mathcal{N}}$ where

$$\begin{aligned}
\mathcal{B}_{\mathcal{H}} = \mathcal{X}^T \tilde{B}_{\mathcal{H}} \mathcal{X} = \mathcal{P}^T \hat{B}_{\mathcal{H}} \mathcal{P} &= \left[\begin{array}{cc:cc} 0 & \mathcal{E} & 0 & \mathcal{F} \\ \mathcal{E} & 0 & \mathcal{F} & 0 \\ \hdashline 0 & \mathcal{G} & 0 & -\mathcal{E}^T \\ \mathcal{G} & 0 & -\mathcal{E}^T & 0 \end{array}\right], \\
\mathcal{B}_{\mathcal{N}} = \mathcal{X}^T \tilde{B}_{\mathcal{N}} \mathcal{X} = \mathcal{P}^T \tilde{B}_{\mathcal{N}} \mathcal{P} &= \left[\begin{array}{cc:cc} \mathcal{K} & 0 & \mathcal{L} & 0 \\ 0 & \mathcal{K} & 0 & \mathcal{L} \\ \hdashline 0 & 0 & \mathcal{K}^T & 0 \\ 0 & 0 & 0 & \mathcal{K}^T \end{array}\right].
\end{aligned} \tag{6.20}$$

Combining (6.16), (6.17), and (6.19) immediately yields

$$\begin{aligned}
\begin{bmatrix} \mathcal{Q}_1 & 0 \\ 0 & \mathcal{J}\mathcal{Q}_2\mathcal{J}^T \end{bmatrix}^T \hat{\mathcal{B}}_{\mathcal{H}} \begin{bmatrix} \mathcal{J}\mathcal{Q}_1\mathcal{J}^T & 0 \\ 0 & \mathcal{Q}_2 \end{bmatrix} &= \left[\begin{array}{cc:cc} 0 & 0 & H_{11} & H_{12} \\ 0 & 0 & 0 & H_{22} \\ \hdashline -H_{22}^T & H_{12}^T & 0 & 0 \\ 0 & -H_{11}^T & 0 & 0 \end{array}\right], \\
\begin{bmatrix} \mathcal{Q}_1 & 0 \\ 0 & \mathcal{J}\mathcal{Q}_2\mathcal{J}^T \end{bmatrix}^T \hat{\mathcal{B}}_{\mathcal{N}} \begin{bmatrix} \mathcal{J}\mathcal{Q}_1\mathcal{J}^T & 0 \\ 0 & \mathcal{Q}_2 \end{bmatrix} &= \left[\begin{array}{cc:cc} N_{11} & N_{12} & 0 & 0 \\ 0 & N_{11}^T & 0 & 0 \\ \hdashline 0 & 0 & Z_{11} & Z_{12} \\ 0 & 0 & 0 & Z_{11}^T \end{array}\right].
\end{aligned} \tag{6.21}$$

Next, we define the double sized $2m \times 2m$ orthogonal matrix

$$\mathcal{Q}_3 = \mathcal{P}^T \begin{bmatrix} \mathcal{J}\mathcal{Q}_1\mathcal{J}^T & 0 \\ 0 & \mathcal{Q}_2 \end{bmatrix} \mathcal{P}. \tag{6.22}$$

Using the identity

$$\mathcal{J}_{2m}\mathcal{P} \begin{bmatrix} \mathcal{J}_m^T & 0 \\ 0 & I_m \end{bmatrix} = \mathcal{P}^T \begin{bmatrix} I_m & 0 \\ 0 & \mathcal{J}_m \end{bmatrix},$$

and following the example in [85], we calculate $\mathcal{J}\mathcal{Q}_3^T\mathcal{J}^T$ as

$$\begin{aligned} \mathcal{J}\mathcal{Q}_3^T\mathcal{J}^T &= \mathcal{J}\mathcal{P} \begin{bmatrix} \mathcal{J}^T\mathcal{Q}_1^T\mathcal{J} & 0 \\ 0 & \mathcal{Q}_2^T \end{bmatrix} \mathcal{P}^T\mathcal{J}^T \\ &= \mathcal{J}\mathcal{P} \begin{bmatrix} \mathcal{J}^T & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \mathcal{Q}_1^T & 0 \\ 0 & \mathcal{Q}_2^T \end{bmatrix} \begin{bmatrix} \mathcal{J} & 0 \\ 0 & I \end{bmatrix} \mathcal{P}^T\mathcal{J}^T \\ &= \mathcal{P}^T \begin{bmatrix} I & 0 \\ 0 & \mathcal{J} \end{bmatrix} \begin{bmatrix} \mathcal{Q}_1^T & 0 \\ 0 & \mathcal{Q}_2^T \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & \mathcal{J}^T \end{bmatrix} \mathcal{P} \\ &= \mathcal{P}^T \begin{bmatrix} \mathcal{Q}_1^T & 0 \\ 0 & \mathcal{J}\mathcal{Q}_2^T\mathcal{J}^T \end{bmatrix} \mathcal{P}. \end{aligned} \tag{6.23}$$

Combining (6.20), (6.21), (6.22), and (6.23), we can calculate

$$\begin{aligned}\mathcal{J}\mathcal{Q}_3^T\mathcal{J}^T(\mathcal{B}_\mathcal{H}-\lambda\mathcal{B}_\mathcal{N})\mathcal{Q}_3 &= \mathcal{P}^T\begin{bmatrix}\mathcal{Q}_1^T & 0\\ 0 & \mathcal{J}\mathcal{Q}_2^T\mathcal{J}^T\end{bmatrix}\mathcal{P}(\mathcal{B}_\mathcal{H}-\lambda\mathcal{B}_\mathcal{N})\mathcal{P}^T\begin{bmatrix}\mathcal{J}\mathcal{Q}_1\mathcal{J}^T & 0\\ 0 & \mathcal{Q}_2\end{bmatrix}\mathcal{P}\\ &= \mathcal{P}^T\begin{bmatrix}\mathcal{Q}_1^T & 0\\ 0 & \mathcal{J}\mathcal{Q}_2^T\mathcal{J}^T\end{bmatrix}(\hat{\mathcal{B}}_\mathcal{H}-\lambda\hat{\mathcal{B}}_\mathcal{N})\begin{bmatrix}\mathcal{J}\mathcal{Q}_1\mathcal{J}^T & 0\\ 0 & \mathcal{Q}_2\end{bmatrix}\mathcal{P} \qquad (6.24)\\ &= \left[\begin{array}{cc:cc} 0 & H_{11} & 0 & H_{12}\\ -H_{22}^T & 0 & H_{12}^T & 0\\ \hdashline 0 & 0 & 0 & H_{22}\\ 0 & 0 & -H_{11}^T & 0\end{array}\right] - \lambda\left[\begin{array}{cc:cc} N_{11} & 0 & N_{12} & 0\\ 0 & Z_{11} & 0 & Z_{12}\\ \hdashline 0 & 0 & N_{11}^T & 0\\ 0 & 0 & 0 & Z_{11}^T\end{array}\right],\end{aligned}$$

where the pencil (6.24) is once again skew-Hamiltonian/Hamiltonian with an eigenvalue spectrum of $\sigma(\mathcal{J}\mathcal{Q}_3^T\mathcal{J}^T\mathcal{B}_\mathcal{H}\mathcal{Q}_3,\, \mathcal{J}\mathcal{Q}_3^T\mathcal{J}^T\mathcal{B}_\mathcal{N}\mathcal{Q}_3) = \sigma(\mathcal{H},\mathcal{N})$. Combining (6.15) and (6.24), we get the equivalency

$$\sigma\left(\begin{bmatrix}0 & H_{11}\\ -H_{22}^T & 0\end{bmatrix},\begin{bmatrix}N_{11} & 0\\ 0 & Z_{11}\end{bmatrix}\right) = \sigma\left(\begin{bmatrix}0 & H_{22}\\ -H_{11}^T & 0\end{bmatrix},\begin{bmatrix}N_{11}^T & 0\\ 0 & Z_{11}^T\end{bmatrix}\right). \qquad (6.25)$$

Finally, using the left half of (6.25), we can solve for the eigenvalues of $\mathcal{H}-\lambda\mathcal{N}$ as

$$\sigma(\mathcal{H},\mathcal{N}) = \pm\sqrt{\sigma(-Z_{11}^{-1}H_{22}^TN_{11}^{-1}H_{11})} = \pm j\sqrt{\sigma(N_{11}^{-1}H_{11}Z_{11}^{-1}H_{22}^T)}.$$

The purely imaginary eigenvalues of $\mathcal{H}-\lambda\mathcal{N}$ correspond to the positive purely real eigen-

values of $\sigma(N_{11}^{-1}H_{11}Z_{11}^{-1}H_{22}^T)$. As this is a combination of upper-triangular and upper quasi-triangular matrices, the matrix product is once again upper quasi-triangular and the purely real eigenvalues can be extracted from the diagonals of the four matrices with no need to compute the inverses of Z_{11} and N_{11}.

6.6.2 Generalized Symplectic URV Decomposition

The decomposition (6.16) consists of three main parts, outlined briefly in Algorithm 11 of [87] with further details in [85]. $\mathcal{N}$ is first put into the skew-triangular form given in (6.16) using a series of Householder reflections to compose an $m \times m$ orthogonal matrix, $\tilde{\mathcal{Q}}_1$, as

$$\tilde{\mathcal{N}} = \tilde{\mathcal{Q}}_1^T \mathcal{N} \mathcal{J} \tilde{\mathcal{Q}}_1 \mathcal{J}^T = \begin{bmatrix} \tilde{N}_{11} & \tilde{N}_{12} \\ 0 & \tilde{N}_{11}^T \end{bmatrix},$$

with $\tilde{N}_{11} \in \mathbb{T}^n$. Taking advantage of the skew-Hessenberg form of $\mathcal{N}$ and the fact that the south-west block of $\mathcal{N}$ is already zero, this can be done in just $n-1$ steps. We then update the other terms as

$$\begin{aligned} \tilde{\mathcal{Q}}_2 &= \mathcal{J} \tilde{\mathcal{Q}}_1 \mathcal{J}^T, \\ \tilde{\mathcal{Z}} &= \mathcal{J} \tilde{\mathcal{Q}}_2^T \mathcal{J}^T \mathcal{N} \tilde{\mathcal{Q}}_2 = \tilde{\mathcal{N}}, \\ \tilde{\mathcal{H}} &= \tilde{\mathcal{Q}}_1^T \mathcal{H} \tilde{\mathcal{Q}}_2. \end{aligned}$$

Orthogonal matrices $\hat{\mathcal{Q}}_1$ and $\hat{\mathcal{Q}}_2$ are then determined to annihilate terms in $\tilde{\mathcal{H}}$ using a series

of Givens rotations and Householder reflections as

$$\begin{aligned}
\hat{\mathcal{N}} &= \hat{\mathcal{Q}}_1^T \tilde{\mathcal{N}} \mathcal{J} \hat{\mathcal{Q}}_1 \mathcal{J}^T = \begin{bmatrix} \hat{N}_{11} & \hat{N}_{12} \\ 0 & \hat{N}_{11}^T \end{bmatrix}, \\
\hat{\mathcal{Z}} &= \mathcal{J} \hat{\mathcal{Q}}_2^T \mathcal{J}^T \tilde{\mathcal{Z}} \hat{\mathcal{Q}}_2 = \begin{bmatrix} \hat{Z}_{11} & \hat{Z}_{12} \\ 0 & \hat{Z}_{11}^T \end{bmatrix}, \\
\hat{\mathcal{H}} &= \hat{\mathcal{Q}}_1^T \tilde{\mathcal{H}} \hat{\mathcal{Q}}_2 = \begin{bmatrix} \hat{H}_{11} & \hat{H}_{12} \\ 0 & \hat{H}_{22} \end{bmatrix},
\end{aligned} \tag{6.26}$$

with $\hat{N}_{11}$, $\hat{Z}_{11}$, $\hat{H}_{11} \in \mathbb{T}^n$, and $\hat{H}_{22}^T$ upper-Hessenberg.

Finally, periodic QZ decomposition is applied to the formal matrix product $\hat{N}_{11}^{-1}\hat{H}_{11}\hat{Z}_{11}^{-1}\hat{H}_{22}^T$ to determine $\breve{\mathcal{Q}}_1$ and $\breve{\mathcal{Q}}_2$ such that

$$\begin{aligned}
\breve{\mathcal{Q}}_1^T \hat{\mathcal{N}} \mathcal{J} \breve{\mathcal{Q}}_1 \mathcal{J}^T &= \begin{bmatrix} N_{11} & N_{12} \\ 0 & N_{11}^T \end{bmatrix}, \\
\mathcal{J} \breve{\mathcal{Q}}_2^T \mathcal{J}^T \hat{\mathcal{Z}} \breve{\mathcal{Q}}_2 &= \begin{bmatrix} Z_{11} & Z_{12} \\ 0 & Z_{11}^T \end{bmatrix}, \\
\breve{\mathcal{Q}}_1^T \hat{\mathcal{H}} \breve{\mathcal{Q}}_2 &= \begin{bmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{bmatrix},
\end{aligned}$$

with N_{11}, Z_{11}, $H_{11} \in \mathbb{T}^n$ and H_{22}^T upper quasi-triangular. We determine $\breve{\mathcal{Q}}_1$ and $\breve{\mathcal{Q}}_2$ as

$$\breve{\mathcal{Q}}_1 = \begin{bmatrix} V_1 & 0 \\ 0 & V_3 \end{bmatrix},$$
$$\breve{\mathcal{Q}}_2 = \begin{bmatrix} V_4 & 0 \\ 0 & V_2 \end{bmatrix},$$

where V_1, V_2, V_3, and V_4 are orthogonal matrices determined by periodic QZ decomposition such that $V_1^T \hat{N}_{11} V_3$, $V_1^T \hat{H}_{11} V_4$, and $V_2^T \hat{Z}_{11} V_4$ are upper-triangular, and $V_2^T \hat{H}_{22}^T V_3$ is upper quasi triangular [88]-[90]. Defining

$$\mathcal{Q}_1 = \tilde{\mathcal{Q}}_1 \hat{\mathcal{Q}}_1 \breve{\mathcal{Q}}_1,$$
$$\mathcal{Q}_2 = \tilde{\mathcal{Q}}_2 \hat{\mathcal{Q}}_2 \breve{\mathcal{Q}}_2,$$

we recover the result (6.16).

A detailed look at the annihilation process up to this point can be found in the generalized symplectic URV decomposition Algorithm 3. While the involved pseudo-code consists of multiplications between pencil matrices and elementary orthogonal matrices, these matrix multiplications are not performed in implementation. A Givens rotation pre-multiplying a full matrix, for example, would only affect two rows of the original matrix. As multiplication between two large matrices is computationally expensive, the affected rows and columns are directly modified in lieu of matrix multiplication to speed up the algorithm significantly.

We now present the pseudocode for the URV decomposition algorithm.

Algorithm 3 Generalized Symplectic URV Decomposition

Require: $\mathcal{H},\ \mathcal{N},\ m$

Assign $\mathcal{Q}_1,\ \mathcal{Q}_2 = I_m$

Assign $n = m/2$

Assign $\mathcal{J} = [0_n\ -I_n;\ I_n\ 0_n]$

% Part 1 - Make N_{11} upper-triangular.

for $i = 1, 2, \ldots, n-1$ **do** ▷ Make $\mathcal{N}$ skew-triangular

Define $\boldsymbol{w} = \mathcal{N}(:, i)$

% Annihilate $\mathcal{N}(i+1:n, i)$ and $\mathcal{N}(n+i, n+i+1:m)$

Set $\mathcal{N} := \mathscr{H}(i, \boldsymbol{w})^T \mathcal{N} \mathcal{J} \mathscr{H}(i, \boldsymbol{w}) \mathcal{J}^T$

Set $\mathcal{H} := \mathscr{H}(i, \boldsymbol{w})^T \mathcal{H} \mathcal{J} \mathscr{H}(i, \boldsymbol{w}) \mathcal{J}^T$ ▷ Update $\mathcal{H}$

Set $\mathcal{Q}_1 := \mathcal{Q}_1 \mathscr{H}(i, \boldsymbol{w})$ ▷ Update $\mathcal{Q}_1$

end for

Set $\mathcal{Q}_2 := \mathcal{Q}_2 \mathcal{J} \mathcal{Q}_1 \mathcal{J}^T$ ▷ Update $\mathcal{Q}_2$

Assign $\mathcal{Z} = \mathcal{N}$ ▷ Initialize $\mathcal{Z}$

% Part 2 - Make H_{11} upper-triangular and H_{22}^T upper-Hessenberg while maintaining the structure of $\mathcal{N}$ and $\mathcal{Z}$.

% When a term $\mathcal{N}(j, k)$ or $\mathcal{Z}(j, k)$ is annihilated, $\mathcal{N}(k+n, j+n)$ or $\mathcal{Z}(k+n, j+n)$ is also annihilated due to symmetry, and vice versa.

for $i = 1, 2, \ldots, n$ **do**

for $j = i, i+1, \ldots, n-1$ **do** ▷ Step 1. Annihilate $\mathcal{H}(n+i:m-1, i)$

Define $\theta_1 = \arctan \mathcal{H}(n+j,i)/\mathcal{H}(n+j+1,i)$

Set $\mathcal{H} := \mathscr{G}_o(n+j,\theta_1)^T\mathcal{H}$ ▷ Annihilate $\mathcal{H}(n+j,i)$

Set $\mathcal{Q}_1^T := \mathscr{G}_o(n+j,\theta_1)^T\mathcal{Q}_1^T$ ▷ Update $\mathcal{Q}_1$

Set $\mathcal{N} := \mathscr{G}_o(n+j,\theta_1)^T\mathcal{N}\mathcal{J}\mathscr{G}_o(n+j,\theta_1)\mathcal{J}^T$ ▷ Update $\mathcal{N}$

Define $\theta_2 = \arctan \mathcal{N}(j+1,j)/\mathcal{N}(j,j)$

Set $\mathcal{N} := \mathscr{G}_o(j,\theta_2)\mathcal{N}\mathcal{J}\mathscr{G}_o(j,\theta_2)^T\mathcal{J}^T$ ▷ Annihilate $\mathcal{N}(j+1,j)$

Set $\mathcal{H} := \mathscr{G}_o(j,\theta_2)\mathcal{H}$ ▷ Update $\mathcal{H}$

Set $\mathcal{Q}_1 := \mathcal{Q}_1\mathscr{G}_o(j,\theta_2)^T$ ▷ Update $\mathcal{Q}_1$

end for

Define $\theta_3 = \arctan \mathcal{H}(m,i)/\mathcal{H}(n,i)$ ▷ Step 2.

Set $\mathcal{H} := \mathscr{G}_s(n,\theta_3)\mathcal{H}$ ▷ Annihilate $\mathcal{H}(m,i)$

Set $\mathcal{N} := \mathscr{G}_s(n,\theta_3)\mathcal{N}\mathscr{G}_s(n,\theta_3)^T$ ▷ Update $\mathcal{N}$

Set $\mathcal{Q}_1 := \mathcal{Q}_1\mathscr{G}_s(n,\theta_3)^T$ ▷ Update $\mathcal{Q}_1$

for $j = n, n-1, \ldots, i+1$ **do** ▷ Step 3. Annihilate $\mathcal{H}(i+1:n,i)$

Define $\theta_4 = \arctan \mathcal{H}(j,i)/\mathcal{H}(j-1,i)$

Set $\mathcal{H} := \mathscr{G}_o(j-1,\theta_4)\mathcal{H}$ ▷ Annihilate $\mathcal{H}(j,i)$

Set $\mathcal{N} := \mathscr{G}_o(j-1,\theta_4)\mathcal{N}\mathcal{J}\mathscr{G}_o(j-1,\theta_4)^T\mathcal{J}^T$ ▷ Update $\mathcal{N}$

Set $\mathcal{Q}_1 := \mathcal{Q}_1\mathscr{G}_o(j-1,\theta_4)^T$ ▷ Update $\mathcal{Q}_1$

Define $\theta_5 = \arctan \mathcal{N}(n+j-1,n+j)/\mathcal{N}(n+j,n+j)$

Set $\mathcal{N} := \mathscr{G}_o(n+j-1,\theta_5)^T\mathcal{N}\mathcal{J}\mathscr{G}_o(n+j-1,\theta_5)\mathcal{J}^T$ ▷ Annihilate $\mathcal{N}(j,j-1)$

Set $\mathcal{H} := \mathscr{G}_o(n+j-1,\theta_5)^T\mathcal{H}$ ▷ Update $\mathcal{H}$

Set $\mathcal{Q}_1 := \mathcal{Q}_1\mathscr{G}_o(n+j-1,\theta_5)$ ▷ Update $\mathcal{Q}_1$

end for

for $j = i+1, i+2, \ldots, n-1$ **do** ▷ Step 4. Annihilate $\mathcal{H}(n+i, i+1:n-1)$

Define $\theta_6 = \arctan \mathcal{H}(n+i,j)/\mathcal{H}(n+i,j+1)$

Set $\mathcal{H} := \mathcal{H}\mathscr{G}_o(j,\theta_6)$ ▷ Annihilate $\mathcal{H}(n+i,j)$

Set $\mathcal{Z} := \mathcal{J}\mathscr{G}_o(j,\theta_6)^T\mathcal{J}^T\mathcal{Z}\mathscr{G}_o(j,\theta_6)$ ▷ Update $\mathcal{Z}$

Set $\mathcal{Q}_2 := \mathcal{Q}_2\mathscr{G}_o(j,\theta_6)$ ▷ Update $\mathcal{Q}_2$

Define $\theta_7 = \arctan \mathcal{Z}(n+j,n+j+1)/\mathcal{Z}(n+j,n+j)$

Set $\mathcal{Z} := \mathcal{J}\mathscr{G}_o(n+j,\theta_7)\mathcal{J}^T\mathcal{Z}\mathscr{G}_o(n+j,\theta_7)^T$ ▷ Annihilate $\mathcal{Z}(j+1,j)$

Set $\mathcal{H} := \mathcal{H}\mathscr{G}_o(n+j,\theta_7)^T$ ▷ Update $\mathcal{H}$

Set $\mathcal{Q}_2 := \mathcal{Q}_2\mathscr{G}_o(n+j,\theta_7)^T$ ▷ Update $\mathcal{Q}_2$

end for

Define $\theta_8 = \arctan \mathcal{H}(n+i,n)/\mathcal{H}(n+i,m)$ ▷ Step 5.

Set $\mathcal{H} := \mathcal{H}\mathscr{G}_s(n,\theta_8)$ ▷ Annihilate $\mathcal{H}(n+i,n)$

Set $\mathcal{Z} := \mathscr{G}_s(n,\theta_8)^T\mathcal{Z}\mathscr{G}_s(n,\theta_8)$ ▷ Update $\mathcal{Z}$

Set $\mathcal{Q}_2 := \mathcal{Q}_2\mathscr{G}_s(n,\theta_8)$ ▷ Update $\mathcal{Q}_2$

for $j = n, n-1, \ldots, k+2$ **do** ▷ Step 6. Annihilate $\mathcal{H}(n+i, n+i+2:m)$

Define $\theta_9 = \arctan \mathcal{H}(n+i,n+j)/\mathcal{H}(n+i,n+j-1)$

Set $\mathcal{H} := \mathcal{H}\mathscr{G}_o(n+j-1,\theta_9)^T$ ▷ Annihilate $\mathcal{H}(n+i,n+j)$

Set $\mathcal{Z} := \mathcal{J}\mathscr{G}_o(n+j-1,\theta_9)\mathcal{J}^T\mathcal{Z}\mathscr{G}_o(n+j-1,\theta_9)^T$ ▷ Update $\mathcal{Z}$

Set $\mathcal{Q}_2 := \mathcal{Q}_2\mathscr{G}_o(n+j-1,\theta_9)^T$ ▷ Update $\mathcal{Q}_2$

Define $\theta_{10} = \arctan \mathcal{Z}(j, j-1)/\mathcal{Z}(j, j)$

Set $\mathcal{Z} := \mathcal{J}\mathcal{G}_o(j-1, \theta_{10})^T \mathcal{J}^T \mathcal{Z} \mathcal{G}_o(j-1, \theta_{10})$ ▷ Annihilate $\mathcal{Z}(j, j-1)$

Set $\mathcal{H} := \mathcal{H}\mathcal{G}_o(j-1, \theta_{10})$ ▷ Update $\mathcal{H}$

Set $\mathcal{Q}_2 := \mathcal{Q}_2 \mathcal{G}_o(j-1, \theta_{10})$ ▷ Update $\mathcal{Q}_2$

end for

end for

% Part 3 - Periodic QZ decomposition

Partition $\mathcal{H} = [H_{11},\ H_{12}; 0,\ H_{22}]$

Partition $\mathcal{N} = [N_{11},\ T_{12}; 0,\ N_{11}^T]$

Partition $\mathcal{Z} = [Z_{11},\ Z_{12}; 0,\ Z_{11}^T]$

Determine orthogonal $V_1,\ V_2,\ V_3,\ V_4$ such that $V_1^T N_{11} V_3$, $V_1^T H_{11} V_4$, and $V_2^T Z_{11} V_4$ are upper-triangular and $V_2^T H_{22}^T V_3$ is upper quasi-triangular.

Set $\mathcal{H} = \text{blkdiag}(V_1^T, V_3^T)\, \mathcal{H}\, \text{blkdiag}(V_4, V_2)$ ▷ Update $\mathcal{H}$

Set $\mathcal{N} = \text{blkdiag}(V_1^T, V_3^T)\, \mathcal{N}\mathcal{J}\, \text{blkdiag}(V_1, V_3)\, \mathcal{J}^T$ ▷ Update $\mathcal{N}$

Set $\mathcal{Z} = \mathcal{J}\, \text{blkdiag}(V_4^T, V_2^T)\, \mathcal{J}^T \mathcal{Z}\, \text{blkdiag}(V_4, V_2)$ ▷ Update $\mathcal{Z}$

Set $\mathcal{Q}_1 = \mathcal{Q}_1\, \text{blkdiag}(V_1, V_3)$ ▷ Update $\mathcal{Q}_1$

Set $\mathcal{Q}_2 = \mathcal{Q}_2\, \text{blkdiag}(V_4, V_2)$ ▷ Update $\mathcal{Q}_2$

Return: $\mathcal{H},\ \mathcal{N},\ \mathcal{Z},\ \mathcal{Q}_1,\ \mathcal{Q}_2$

6.6.3 Skew-Triangular/Skew-Hessenberg Decomposition

We now present an example implementation of Algorithm 3 on a real valued skew-Hamiltonian/ Hamiltonian pencil $\mathcal{H} - \lambda\mathcal{N}$ with dimension $n = 3$. As with the discrete-time IQC oracle, the south-west block of $\mathcal{N}$ is identically zero. We use the symbol $\times$ to represent a term that may be non-zero, $\bigcirc$ for a a term that is exactly zero, $\odot$ for a term annihilated by the current operation, and $\otimes$ for a possibly non-zero term that was previously zero.

Once again, we adopt `MATLAB`'s matrix notation for convenience. That is, given a matrix X, $X(i:j, k:l)$ refers to the block $X((i, i+1, \ldots, j-1, j), (k, k+1, \ldots, l-1, l))$.

In Part 1 of Algorithm 3, two Householder reflections, $\mathscr{H}$, are applied to $\mathcal{N}$ as $\mathscr{H}^T\mathcal{N}\mathcal{J}\mathscr{H}\mathcal{J}^T$ to make the north-west block of $\mathcal{N}$ upper-triangular and the south-east block lower triangular. Due to the skew-Hessenberg form of $\mathcal{N}$, as $\mathcal{N}(2:3, 1)$ is annihilated, so is $\mathcal{N}(4, 5:6)$. Likewise, as $\mathcal{N}(3, 2)$ is annihilated, so is $\mathcal{N}(5, 6)$. At each step in the process, updated values of $\mathscr{H}^T\mathcal{N}\mathcal{J}\mathscr{H}\mathcal{J}^T$ are stored in the variable $\mathcal{N}$. The orthogonal matrices that result in the updated version of $\mathcal{N}$ are stored separately. Recall that any matrix multiplications applied to $\mathcal{N}$ are also applied to $\mathcal{H}$, and vice versa. Thus, the algorithm is carefully set up so that annihilated terms in one matrix that get filled in while operating on a second matrix are immediately annihilated again so that operations on one matrix do not result in backwards progress on the other. The progression of $\mathcal{N}$ as the Householder reflections are applied is as

follows:

$$\mathcal{N} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \times & \times & \times & \times & \bigcirc & \times \\ \times & \times & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right] := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \odot & \times & \times & \times & \bigcirc & \times \\ \odot & \times & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \odot & \odot \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right] := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \odot & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \odot \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Now that $\mathcal{N}$ is in skew-triangular form, we set $\mathcal{Z} = \mathcal{N}$ and begin to annihilate terms in $\mathcal{H}$ while maintaining the structure of $\mathcal{N}$ and $\mathcal{Z}$ in Part 2 of the algorithm. At this point, it does not matter how the structure of $\mathcal{H}$ has been affected by the Householder reflections applied in Part 1, and it can be any full real-valued 6×6 matrix, as depicted by

$$\mathcal{H} = \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \hdashline \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{N} = \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \bigcirc & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right].$$

In Step 1, we use $\mathscr{G}(4, 5, \theta_1)$ to eliminate $\mathcal{H}(4, 1)$ from the left side as $\mathcal{H} := \mathscr{G}(4, 5, \theta_1)^T \mathcal{H}$. This creates nonzero entries at $\mathcal{N}(2, 1)$ and $\mathcal{N}(4, 5)$, which are in turn annihilated by $\mathscr{G}(1, 2, \theta_2)$ as $\mathcal{N} := \mathscr{G}(1, 2, \theta_2)^T \mathcal{N} \mathcal{J} \mathscr{G}(1, 2, \theta_2) \mathcal{J}^T$. Angles corresponding to these and all following Givens

rotations can be found in Algorithm 3.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \hdashline \odot & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{N} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \otimes & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \bigcirc & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \otimes & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Repeating this step annihilates $\mathcal{H}(5,1)$ with $\mathscr{G}(5,6,\theta_1)$ and recovers $\mathcal{N}(3,2)$ and $\mathcal{N}(5,6)$ with the rotation $\mathscr{G}(2,3,\theta_2)$.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \hdashline \bigcirc & \times & \times & \times & \times & \times \\ \odot & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{N} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \otimes & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \otimes \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Step 2 annihilates $\mathcal{H}(6,1)$ using the Givens rotation $\mathscr{G}(3,6,\theta_3)$ from the left as $\mathcal{H} := \mathscr{G}(3,6,\theta_3)\mathcal{H}$. Since the rotation matrix is symplectic, we have $\mathscr{G}(3,6,\theta_3) = \mathcal{J}\mathscr{G}(3,6,\theta_3)\mathcal{J}^T$. $\mathcal{N}$ is thus updated as $\mathcal{N} := \mathscr{G}(3,6,\theta_3)\mathcal{N}\mathscr{G}(3,6,\theta_3)^T$. In this update step, rotated terms in $\mathcal{N}$ cancel out their symmetric counterparts, and thus no new non-zero terms are formed in

$\mathcal{N}$.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \hdashline \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \odot & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{N} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \bigcirc & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

In Step 3, $\mathscr{G}(2,3,\theta_4)$ is used to annihilate $\mathcal{H}(3,1)$ from the left while $\mathscr{G}(5,6,\theta_5)$ is used to recover $\mathcal{N}(3,2)$ and $\mathcal{N}(5,6)$. Repeating this finishes the elimination of the first column of $\mathcal{H}$ by annihilating $\mathcal{H}(2,1)$ with $\mathscr{G}(1,2,\theta_4)$ and recovering $\mathcal{N}(2,1)$ and $\mathcal{N}(4,5)$ with $\mathscr{G}(4,5,\theta_5)$.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \odot & \times & \times & \times & \times & \times \\ \odot & \times & \times & \times & \times & \times \\ \hdashline \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{N} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \otimes & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \otimes & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \otimes & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \otimes \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Step 4 annihilates $\mathscr{G}(2,3,\theta_6)$ from the right as $\mathcal{H} := \mathcal{H}\mathscr{G}(2,3,\theta_6)$, updating $\mathcal{Z}$ as $\mathcal{Z} := \mathcal{J}\mathscr{G}(2,3,\theta_6)^T\mathcal{J}^T\mathcal{Z}\mathscr{G}(2,3,\theta_6)$ and creating two non-zero terms in $\mathcal{Z}$. $\mathcal{Z}(2,1)$ and $\mathcal{Z}(4,5)$ are then annihilated using $\mathscr{G}(5,6,\theta_7)$.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \hdashline \bigcirc & \odot & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{Z} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \otimes & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \bigcirc & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \otimes & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Step 5 uses $\mathcal{G}(3,6,\theta_8)$ to annihilate $\mathcal{H}(4,3)$. Similar to Step 2, the rotation matrix is symplectic and $\mathcal{Z}$ is updated with no new nonzero terms as $\mathcal{Z} := \mathcal{G}(3,6,\theta_8)^T \mathcal{Z} \mathcal{G}(3,6,\theta_8)$.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \hdashline \bigcirc & \bigcirc & \odot & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{Z} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \bigcirc & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Step 6 begins the elimination of the south-east block of $\mathcal{H}$ by using $\mathscr{G}(5,6,\theta_9)$ to annihilate $\mathcal{H}(4,6)$ and $\mathscr{G}(2,3,\theta_{10})$ to recover $\mathcal{Z}(3,2)$ and $\mathcal{Z}(5,6)$.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \times & \odot \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{Z} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \otimes & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \otimes \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Steps 1, 2, 3, and 5 are then repeated to annihilate $\mathcal{H}(5,2)$, $\mathcal{H}(6,2)$, $\mathcal{H}(3,2)$, and $\mathcal{H}(5,3)$, respectively. $\mathcal{N}(3,2)$ and $\mathcal{N}(5,6)$ are recovered in Steps 1 and 3. As $\mathcal{H}(4,2)$ was previously annihilated, there is no need to perform Step 4. Likewise, as the transpose of the south-east

block of $\mathcal{H}$ is already upper-Hessenberg, there is no need to perform Step 6.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \odot & \times & \times & \times & \times \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \odot & \odot & \times & \times & \times \\ \bigcirc & \odot & \times & \times & \times & \times \end{array}\right], \qquad \mathcal{N} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \otimes & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \otimes \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

Finally, Step 2 is repeated once more to eliminate $\mathcal{H}(6,3)$ with a symplectic Givens rotation and the structure of $\mathcal{N}$ is unaffected.

$$\mathcal{H} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \times & \times & \times \\ \bigcirc & \times & \times & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times & \times & \times \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \odot & \times & \times & \times \end{array}\right], \qquad \mathcal{N} := \left[\begin{array}{ccc:ccc} \times & \times & \times & \bigcirc & \times & \times \\ \bigcirc & \times & \times & \times & \bigcirc & \times \\ \bigcirc & \bigcirc & \times & \times & \times & \bigcirc \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times & \bigcirc & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \bigcirc \\ \bigcirc & \bigcirc & \bigcirc & \times & \times & \times \end{array}\right]$$

The upper-triangular matrices N_{11}, Z_{11}, H_{11}, and upper-Hessenberg H_{22}^T can now be extracted from $\mathcal{H}$, $\mathcal{N}$, and $\mathcal{Z}$ for periodic QZ decomposition.

6.6.4 Periodic QZ Decomposition and Infinite Eigenvalue Deflation

Periodic QZ decomposition is a technique used to determine the eigenvalues of a discrete-time periodic descriptor system [88]-[90]. While the application here is different, the technique

can be applied to any sequence of inverted or non-inverted square matrices. Typically, the most expensive step in periodic QZ decomposition is to put the matrix sequence in cyclic-Hessenberg form, namely, make one matrix upper-Hessenberg, and the rest upper-triangular. In the case of GSURV decomposition, we immediately start the periodic QZ decomposition in cyclic-Hessenberg form, and furthermore, know that the matrix product

$$\mathscr{N} = \hat{N}_{11}^{-1}\hat{H}_{11}\hat{Z}_{11}^{-1}\hat{H}_{22}^{T}$$

is purely real. Recall that $\hat{N}_{11}$, $\hat{H}_{11}$, $\hat{Z}_{11}$, and $\hat{H}_{22}$ are partitioned from $\hat{\mathcal{N}}$, $\hat{\mathcal{H}}$, and $\hat{\mathcal{Z}}$ in (6.26). As has been done previously in this chapter, we can pre and post multiply $\mathscr{N}$ by full rank matrices and maintain its eigenvalue spectrum. If V_1, V_2, V_3, and V_4 are orthogonal matrices, then we can state

$$\begin{aligned} V_3^T\mathscr{N}V_3 &= V_3^T\hat{N}_{11}^{-1}V_1V_1^T\hat{H}_{11}V_4V_4^T\hat{Z}_{11}^{-1}V_2V_2^T\hat{H}_{22}^TV_3, \\ &= V_3^T\hat{N}_{11}^{-1}\hat{H}_{11}\hat{Z}_{11}^{-1}\hat{H}_{22}^TV_3, \end{aligned}$$

with $\sigma(V_3^T\mathscr{N}V_3) = \sigma(\mathscr{N})$. Looking at the matrices surrounding $\hat{Z}_{11}^{-1}$, we can take advantage of their orthogonality and state that $V_4^T\hat{Z}_{11}^{-1}V_2 = (V_2^T\hat{Z}_{11}V_4)^{-1}$. Thus, we can manipulate $\hat{Z}_{11}$ without directly inverting it using V_2 and V_4. Furthermore, we know that the inverse of an upper-triangular matrix is once again upper-triangular. Similarly, V_1 and V_3 can be used to operate on $\hat{N}_{11}$. Using periodic QZ decomposition, we will cyclically determine V_1, V_2, V_3, and V_4 such that $V_1^TN_{11}V_3$, $V_1^TH_{11}V_4$, and $V_2^TZ_{11}V_4$ are upper-triangular and

$V_2^T H_{22}^T V_3$ is upper quasi-triangular. $V_3^T \mathscr{N} V_3$ will therefore be upper quasi-triangular, and its real eigenvalues can be extracted from the diagonals of the upper-triangular portion of the matrix product.

From the formation of our matrix pencil, we know that it is guaranteed to have several infinite eigenvalues. Due to our choice of boundary condition frequency ($\omega = \pi$), none of the infinite eigenvalues will correspond to critical frequencies of our IQC oracle. The present infinite eigenvalues are signified by zeros on the diagonals of the inverted matrices $\hat{N}_{11}$ and $\hat{Z}_{11}$. In this step, we seek to decouple these infinite eigenvalues from the rest of the pencil's eigenvalues.

Considering an example with dimension $n = 4$, suppose there is a zero on the diagonal of $\hat{Z}_{11}$ at $\hat{Z}_{11}(2,2)$. We will manipulate the matrix product until the corresponding infinite eigenvalue can be decoupled. Clearly, all the eigenvalues of an upper-triangular matrix are decoupled. We seek to annihilate an appropriate term on the sub-diagonal of $\hat{H}_{22}^T$ such that the infinite eigenvalue can be decoupled. $\mathscr{N}$ initially has the form

$$\mathscr{N} = \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \odot & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \end{bmatrix}.$$

$V_2 = \mathscr{G}(2,3,\theta_1)^T$ is first used to annihilate $\hat{Z}_{11}(3,3)$, creating a nonzero term at $\hat{H}_{22}^T(3,1)$ by setting $\hat{Z}_{11} := V_2^T \hat{Z}_{11}$ and $\hat{H}_{22}^T := V_2^T \hat{H}_{22}^T$. This creates two zero diagonal terms in a row for

$\hat{Z}_{11}$.

$$\mathscr{N} := \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \odot & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \otimes & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \end{bmatrix}$$

The new nonzero term in $\hat{H}_{22}^T$ is now "chased" around the matrix product in order to recover cyclic-Hessenberg form. $V_3 = \mathscr{G}(1, 2, \theta_2)$ is used to recover $\hat{H}_{22}^T(3, 1)$ while creating the nonzero sub-diagonal term $\hat{N}_{11}(2, 1)$ by setting $\hat{H}_{22}^T := \hat{H}_{22}^T V_3$ and $\hat{N}_{11} := \hat{N}_{11} V_3$.

$$\mathscr{N} := \begin{bmatrix} \times & \times & \times & \times \\ \otimes & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \odot & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \end{bmatrix}$$

$V_1 = \mathscr{G}(1, 2, \theta_3)$ next recovers $\hat{N}_{11}(2, 1)$ while creating the corresponding nonzero term at $\hat{H}_{11}(2, 1)$ by setting $\hat{N}_{11} := V_1^T \hat{N}_{11}$ and $\hat{H}_{11} := V_1^T \hat{H}_{11}$.

$$\mathscr{N} := \begin{bmatrix} \times & \times & \times & \times \\ \odot & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \otimes & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \end{bmatrix}$$

Finally, $V_4 = \mathscr{G}(1, 2, \theta_4)$ is used to recover $\hat{H}_{11}(2, 1)$ by setting $\hat{H}_{11} := \hat{H}_{11} V_4$ and $\hat{Z}_{11} := \hat{Z}_{11} V_4$. Because of the two diagonal zero terms in $\hat{Z}_{11}$, V_4 does not create any new nonzero

terms in $\hat{Z}_{11}$, and thus, we have recovered cyclic-Hessenberg form as

$$\mathscr{N} := \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \odot & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \end{bmatrix}.$$

This cyclic process is repeated in order to annihilate $\hat{Z}_{11}(4,4)$. This time, however, V_4 causes the original diagonal zero at $\hat{Z}_{11}(2,2)$ to become possibly nonzero. If the matrix product was larger in dimension, this cyclic process would be repeated until the last two diagonal terms of $\hat{Z}_{11}$ were zeros.

$$\mathscr{N} := \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \otimes & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \\ \bigcirc & \bigcirc & \bigcirc & \odot \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \end{bmatrix}$$

We next annihilate the bottom sub-diagonal term of $\hat{H}_{22}^T$ with $V_3 = \mathscr{G}(n-1, n, \theta_5)$, creating a nonzero sub-diagonal term at $\hat{N}_{11}(4,3)$.

$$\mathscr{N} = \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \otimes & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \\ \bigcirc & \bigcirc & \bigcirc & \bigcirc \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \odot & \times \end{bmatrix}$$

Similar to before, this nonzero term can be chased around the matrix product until cyclic-Hessenberg form is recovered. $V_1 = \mathscr{G}(n-1, n, \theta_6)^T$ is used to recover $\hat{N}_{11}(4,3)$ and shift the

created nonzero term to $\hat{H}_{11}(4,3)$.

$$\mathscr{N} := \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \odot & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \otimes & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \\ \bigcirc & \bigcirc & \bigcirc & \bigcirc \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}$$

Finally, $\hat{H}_{11}(4,3)$ is annihilated with $V_4 = \mathscr{G}(n-1, n, \theta_7)$, filling in the diagonal term $\hat{Z}_{11}(3,3)$ and recovering cyclic-Hessenberg form. $\hat{N}_{11}$, $\hat{H}_{11}$, and $\hat{Z}_{11}$ are now upper-triangular, and $\hat{H}_{22}^T$ has a zero sub-diagonal term. The bottom eigenvalue, infinity due to the zero diagonal in $\hat{Z}_{11}$, is decoupled from the remaining eigenvalues allowing the product to be truncated to a 3×3 matrix product for further deflation.

$$\mathscr{N} := \left[\begin{array}{ccc:c} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times \end{array}\right]^{-1} \left[\begin{array}{ccc:c} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \hdashline \bigcirc & \bigcirc & \odot & \times \end{array}\right] \left[\begin{array}{ccc:c} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \otimes & \times \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \bigcirc \end{array}\right]^{-1} \left[\begin{array}{ccc:c} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \hdashline \bigcirc & \bigcirc & \bigcirc & \times \end{array}\right]$$

The corresponding pseudo-code for deflating an infinite eigenvalue corresponding to a zero on the $\hat{Z}_{11}$ diagonal is now presented. A similar process can be performed for zeros on the $\hat{N}_{11}$ diagonal.

Algorithm 4 Infinite Eigenvalue Deflation

Require: $N_{11},\ H_{11},\ Z_{11},\ H_{22},\ n$

Assign $V_1, V_2, V_3, V_4 = I_n$

for $i = n, n-1, \ldots, 1$ **do**

if $Z_{11}(i,i) = 0$ **then**

for $j = i, i+1, \ldots, n-1$ **do** ▷ Chase zeros down the Z_{11} diagonal

Define $\theta_1 = \arctan Z_{11}(j+1, j+1)/Z_{11}(j, j+1)$ ▷ Annihilate diagonal term

Set $Z_{11} := \mathscr{G}_o(j, \theta_1) Z_{11}$ ▷ Annihilate $Z_{11}(j+1, j+1)$

Set $H_{22} := H_{22} \mathscr{G}_o(j, \theta_1)^T$ ▷ Update H_{22}

Set $V_2 := V_2 \mathscr{G}_o(j, \theta_1)^T$ ▷ Update V_2

if $j \neq 1$ **then** ▷ Chase nonzero term around matrix product

Define $\theta_2 = \arctan H_{22}(j-1, j+1)/H_{22}(j, j+1)$

Set $H_{22} := \mathscr{G}_o(j-1, \theta_2)^T H_{22}$ ▷ Annihilate $H_{22}(j-1, j+1)$

Set $N_{11} := N_{11} \mathscr{G}_o(j-1, \theta_2)$ ▷ Update N_{11}

Set $V_3 := V_3 \mathscr{G}_o(j-1, \theta_2)$ ▷ Update V_3

Define $\theta_3 = \arctan N_{11}(j, j-1)/N_{11}(j-1, j-1)$

Set $N_{11} := \mathscr{G}_o(j-1, \theta_3) N_{11}$ ▷ Annihilate $N_{11}(j, j-1)$

Set $H_{11} := \mathscr{G}_o(j-1, \theta_3) H_{11}$ ▷ Update H_{11}

Set $V_1 := V_1 \mathscr{G}_o(j-1, \theta_3)^T$ ▷ Update V_1

Define $\theta_4 = \arctan H_{11}(j, j-1)/H_{11}(j, j)$

Set $H_{11} := H_{11} \mathscr{G}_o(j-1, \theta_4)$ ▷ Annihilate $H_{11}(j, j-1)$

Set $Z_{11} := Z_{11} \mathscr{G}_o(j-1, \theta_4)$ ▷ Update Z_{11}

Set $V_4 := V_4 \mathscr{G}_o(j-1, \theta_4)$ ▷ Update V_4

end if

end for

Define $\theta_5 = \arctan H_{22}(n-1, n)/H_{22}(n, n)$ ▷ Deflate H_{22}

Set $H_{22} := \mathscr{G}_o(n-1,\theta_5)^T H_{22}$ ▷ Annihilate $H_{22}(n-1,n)$

Set $N_{11} := N_{11}\mathscr{G}_o(n-1,\theta_5)$ ▷ Update N_{11}

Set $V_3 := V_3\mathscr{G}_o(n-1,\theta_5)$ ▷ Update V_3

Define $\theta_6 = \arctan N_{11}(n,n-1)/N_{11}(n-1,n-1)$

Set $N_{11} := \mathscr{G}_o(n-1,\theta_6)N_{11}$ ▷ Annihilate $N_{11}(n,n-1)$

Set $H_{11} := \mathscr{G}_o(n-1,\theta_6)H_{11}$ ▷ Update H_{11}

Set $V_1 := V_1\mathscr{G}_o(n-1,\theta_6)^T$ ▷ Update V_1

Define $\theta_7 = \arctan H_{11}(n,n-1)/H_{11}(n,n)$

Set $H_{11} := H_{11}\mathscr{G}_o(n-1,\theta_7)$ ▷ Annihilate $H_{11}(n,n-1)$

Set $Z_{11} := Z_{11}\mathscr{G}_o(n-1,\theta_7)$ ▷ Update Z_{11}

Set $V_4 := V_4\mathscr{G}_o(n-1,\theta_7)$ ▷ Update V_4

Set $n := n-1$ ▷ Artificially truncate last row and column

end if

end for

Return: $N_{11},\ H_{11},\ Z_{11},\ H_{22},\ V_1,\ V_2,\ V_3,\ V_4,\ n$

6.6.5 Periodic QZ Block Update

As the matrix product gets deflated, only the non-deflated portion needs to be operated on. Additionally, the periodic QZ iteration step will create opportunities to split the deflation problem into two smaller subproblems. A block update step is used to to represent the effects of orthogonal matrices on the already deflated or nonactive portions of the matrix product

[89]. For example, H_{11} might be partitioned as

$$H_{11} = \left[\begin{array}{c:c:c} H_{11_{11}} & H_{11_{12}} & H_{11_{13}} \\ \hdashline 0 & H_{11_{22}} & H_{11_{23}} \\ \hdashline 0 & 0 & H_{11_{33}} \end{array}\right],$$

where the matrix products corresponding to the upper-triangular sub-matrices $H_{11_{11}}$ and $H_{11_{33}}$ are deflated upper quasi-triangular, and the matrix product corresponding to $H_{11_{22}}$ has yet to be deflated. An orthogonal matrix V_1 left multiplying H_{11} only needs to modify the non-deflated portion $H_{11_{22}}$, and can therefore be chosen as

$$V_1 = \left[\begin{array}{c:c:c} I & 0 & 0 \\ \hdashline 0 & v_1 & 0 \\ \hdashline 0 & 0 & I \end{array}\right]$$

with dimensions corresponding to the partition of H_{11}. If orthogonal matrices v_1 and v_4 are determined such that they modify $H_{11_{22}}$ as $v_1 H_{11_{22}} v_4^T$, the corresponding multiplication of $V_1 H_{11} V_4^T$ can be achieved by replacing $H_{11_{12}}$ by $H_{11_{12}} v_4^T$ and $H_{11_{23}}$ by $v_1 H_{11_{23}}$. Similar updates can be made to Z_{11}, N_{11}, and H_{22}^T.

6.6.6 Periodic QZ Iteration

For the iterative step of the periodic QZ decomposition, we first estimate the eigenvalues of $\mathscr{N}$ as in [88] and perform a double implicit shift until H_{22}^T converges to upper quasi-triangular form. By estimating the eigenvalues of $\mathscr{N}$ and performaing a shift, we aim to create zeros on the subdiagonal of $\hat{H}_{22}^T$ corresponding to real eigenvalues of $\mathscr{N}$. Cyclic-Hessenberg form for the non-deflated portion of the matrix product is recovered cyclically, similar to the infinite eigenvalue deflation in Section 6.6.4.

To estimate eigenvalues, we first define the upper-triangular portion of $\mathscr{N}$ as

$$\mathscr{N}_T = \hat{N}_{11}^{-1}\hat{H}_{11}\hat{Z}_{11}^{-1}.$$

Additionally, we define the north-west and south-east 2×2 blocks of $\mathscr{N}_T$ and $\mathscr{N}$ as

$$\mathscr{N}_A = \hat{N}^{-1}_{11_{1:2,1:2}}\hat{H}_{11_{1:2,1:2}}\hat{Z}^{-1}_{11_{1:2,1:2}},$$
$$\mathscr{N}_D = \hat{N}^{-1}_{11_{l:n,l:n}}\hat{H}_{11_{l:n,l:n}}\hat{Z}^{-1}_{11_{l:n,l:n}}\hat{H}^T_{22_{l:n,l:n}},$$

respectively, where $l = n - 1$. To carry out the implicit double shift, we determine the first column of the matrix $\mathscr{N}_H = (\mathscr{N} - \lambda_1)(\mathscr{N} - \lambda_2)$, where λ_1 and λ_2 are the eigenvalues of $\mathscr{N}_D$.

Since $\mathscr{N}_H$ is upper-Hessenberg, the first column consists of two non-zero terms, calculated as

$$\mathscr{N}_H(:,1) = \hat{H}^T_{22_{1,1:2}}\hat{H}_{11_{1,1}}/(\hat{N}_{11_{1,1}}\hat{Z}_{11_{1,1}}),$$

Similarly, the first column of $\mathscr{N}_H^2$ has three non-zero elements, defined as

$$H^T_{22_{1:2,1:3}} \mathscr{N}_A H^T_{22_{1,1:2}} H_{11_{1,1}} / (\hat{N}_{11_{1,1}} \hat{Z}_{11_{1,1}}).$$

We define the zeroth column of H_{22}^T as the first column of the scaled matrix $\mathscr{N}_H^2 - \mathscr{N}_H \mathrm{Tr} \mathscr{N}_D + I \det \mathscr{N}_D$, defined as

$$H^T_{22_{0,1:3}} = H^T_{22_{1:2,1:3}} \mathscr{N}_A H^T_{22_{1,1:2}} - H^T_{22_{1,1:3}} \mathrm{Tr} \mathscr{N}_D + [\det \mathscr{N}_D \hat{N}_{11_{1,1}} \hat{Z}_{11_{1,1}} / H_{11_{1,1}} \ 0 \ 0]^T.$$

Beginning with the zeroth column of $\hat{H}_{22}^T$ and continuing to the $n-3^{\text{rd}}$ column, we will perform a shift, and then cyclically chase the newly created nonzero elements around the matrix product to recover cyclic-Hessenberg form. If a zero on the diagonal of $\hat{H}_{22}^T$ appears after a double shift iteration, the matrix product can be split into two subproblems. For example, if $\hat{H}_{22}^T(3,2) = 0$, we can split the matrix product as

$$\mathscr{N} := \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times \\ \bigcirc & \times & \times & \times \\ \bigcirc & \bigcirc & \times & \times \\ \bigcirc & \bigcirc & \bigcirc & \times \end{bmatrix}^{-1} \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \bigcirc & \odot & \times & \times \\ \bigcirc & \bigcirc & \times & \times \end{bmatrix}.$$

We now present the algorithm for the periodic QZ iteration step of the decomposition.

Algorithm 5 Periodic QZ Iteration

Require: $N_{11},\ H_{11},\ Z_{11},\ H_{22},\ n$

Assign $V_1, V_2, V_3, V_4 = I_n$

Determine $H_{22}(0, 1:3)$

for $i = 1, 2, \ldots, n-2$ **do**

% Eliminate column $i-1$ of 3×3 bulge

Define $\boldsymbol{w}_1 = H_{22}(i-1, :)$

Set $H_{22} := H_{22}\mathscr{H}(i, \boldsymbol{w}_1)$ ▷ Annihilate $H_{22}(i-1, i+1:i+2)$

Set $Z_{11} := \mathscr{H}(i, \boldsymbol{w}_1)Z_{11}$ ▷ Update Z_{11}

Set $V_2 := V_2\mathscr{H}(i, \boldsymbol{w}_1)$ ▷ Update V_2

% Make Z_{11} upper-triangular

Define $\theta_1 = \arctan Z_{11}(i+2, i)/Z_{11}(i+2, i+2)$

Set $Z_{11} := Z_{11}\mathscr{G}(i, i+2, \theta_1)$ ▷ Annihilate $Z_{11}(i+2, i)$

Define $\theta_2 = \arctan Z_{11}(i+2, i+1)/Z_{11}(i+2, i+2)$

Set $Z_{11} := Z_{11}\mathscr{G}_o(i+1, \theta_2)$ ▷ Annihilate $Z_{11}(i+2, i+1)$

Define $\theta_3 = \arctan Z_{11}(i+1, i)/Z_{11}(i+1, i+1)$

Set $Z_{11} := Z_{11}\mathscr{G}_o(i, \theta_3)$ ▷ Annihilate $Z_{11}(i+1, i)$

Set $H_{11} := H_{11}\mathscr{G}(i, i+2, \theta_1)\mathscr{G}_o(i+1, \theta_2)\mathscr{G}_o(i, \theta_3)$ ▷ Update H_{11}

Set $V_4 := V_4\mathscr{G}(i, i+2, \theta_1)\mathscr{G}_o(i+1, \theta_2)\mathscr{G}_o(i, \theta_3)$ ▷ Update V_4

% Make H_{11} upper-triangular

Define $\boldsymbol{w}_2 = H_{11}(:, i)$

Set $H_{11} := \mathscr{H}(i, \boldsymbol{w}_2)H_{11}$ ▷ Annihilate $H_{11}(i+i:i+2, i)$

Define $\theta_4 = \arctan H_{11}(i+2, i+1)/H_{11}(i+1, i+1)$

Set $H_{11} := \mathscr{G}_o(i+1, \theta_4)H_{11}$ ▷ Annihilate $H_{11}(i+2, i+1)$

Set $N_{11} := \mathscr{G}_o(i+1,\theta_4)\mathscr{H}(i,\boldsymbol{w}_2)N_{11}$ ▷ Update N_{11}

Set $V_1 := V_1\mathscr{H}(i,\boldsymbol{w}_2)\mathscr{G}_o(i+1,\theta_4)^T$ ▷ Update V_1

% Make N_{11} upper-triangular

Define $\theta_5 = \arctan N_{11}(i+2,i)/N_{11}(i+2,i+2)$

Set $N_{11} := N_{11}\mathscr{G}(i,i+2,\theta_5)$ ▷ Annihilate $N_{11}(i+2,i)$

Define $\theta_6 = \arctan N_{11}(i+2,i+1)/N_{11}(i+2,i+2)$

Set $N_{11} := N_{11}\mathscr{G}_o(i+1,\theta_6)$ ▷ Annihilate $N_{11}(i+2,i+1)$

Define $\theta_7 = \arctan N_{11}(i+1,i)/N_{11}(i+1,i+1)$

Set $N_{11} := N_{11}\mathscr{G}_o(i,\theta_7)$ ▷ Annihilate $N_{11}(i+1,i)$

Set $H_{22} := \mathscr{G}_o(i,\theta_7)^T\mathscr{G}_o(i+1,\theta_6)^T\mathscr{G}(i,i+2,\theta_5)^T H_{22}$ ▷ Update H_{22}

Set $V_3 := V_3\mathscr{G}(i,i+2,\theta_5)\mathscr{G}_o(i+1,\theta_6)\mathscr{G}_o(i,\theta_7)$ ▷ Update V_3

end for

% Recover cyclic Hessenberg form

Define $\boldsymbol{w}_3 = H_{22}(n-2,:)$

Set $H_{22} := H_{22}\mathscr{H}(n-1,\boldsymbol{w}_3)$ ▷ Annihilate $H_{22}(n-2,n)$

Set $Z_{11} := \mathscr{H}(n-1,\boldsymbol{w}_3)Z_{11}$ ▷ Update Z_{11}

Set $V_2 := V_2\mathscr{H}(n-1,\boldsymbol{w}_3)$ ▷ Update V_2

Define $\theta_8 = \arctan Z_{11}(n,n-1)/Z_{11}(n,n)$

Set $Z_{11} := Z_{11}\mathscr{G}_o(n-1,\theta_8)$ ▷ Annihilate $Z_{11}(n-1,n)$

Set $H_{11} := H_{11}\mathscr{G}_o(n-1,\theta_8)$ ▷ Update H_{11}

Set $V_4 := V_4\mathscr{G}_o(n-1,\theta_8)$ ▷ Update V_4

Define $\theta_9 = \arctan H_{11}(n, n-1)/H_{11}(n-1, n-1)$

Set $H_{11} := \mathscr{G}_o(n-1, \theta_9) H_{11}$ ▷ Annihilate $H_{11}(n, n-1)$

Set $N_{11} := \mathscr{G}_o(n-1, \theta_9) N_{11}$ ▷ Update N_{11}

Set $V_1 := V_1 \mathscr{G}_o(n-1, \theta_9)^T$ ▷ Update V_1

Define $\theta_{10} = \arctan N_{11}(n, n-1)/N_{11}(n, n)$

Set $N_{11} := N_{11} \mathscr{G}_o(n-1, \theta_{10})$ ▷ Annihilate $N_{11}(n, n-1)$

Set $H_{22} := \mathscr{G}_o(n-1, \theta_{10})^T H_{22}$ ▷ Update H_{22}

Set $V_3 := V_3 \mathscr{G}_o(n-1, \theta_{10})$ ▷ Update V_3

Return: $N_{11},\ H_{11},\ Z_{11},\ H_{22},\ V_1,\ V_2,\ V_3,\ V_4,\ n$

6.7 Discrete-Time IQC Oracle Examples

Randomly generated discrete-time LFRs were analyzed using an ACCP algorithm and the KYP lemma. The convex optimization solution used `SDPT3` parsed through the LMI parser `YALMIP` [30], [60]. All calculations were performed in `MATLAB` 2014a on an Intel i7 CPU with 8 GB of RAM. Solution times for the randomly generated single-input single-output (SISO) LFRs with varying state dimensions are shown in Fig. 6.2 on a semi-log scale. For small systems consisting of 30 or less states, the KYP solver was faster, with both solution techniques finishing in less than 10 seconds. For medium and large problems with 40 or more LFR states, however, the ACCP solved the analysis problem the fastest. For LFRs with 80 states, the ACCP solved problems in an average of 35 seconds, while the average KYP solution took over 950 seconds. Our desktop computer was unable to successfully solve

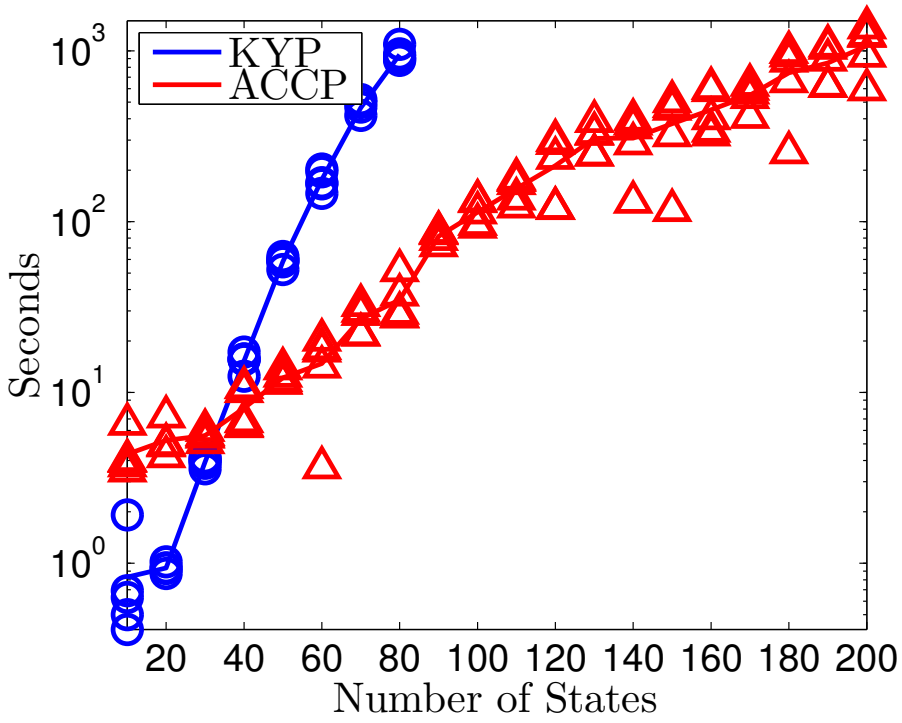

Figure 6.2: Solution times for randomly generated discrete-time IQC analysis problems with the KYP lemma and an ACCP algorithm.

a problem with 90 states using the KYP method. The ACCP method, on the other hand, was able to solve problems with 200 states in 1071 seconds. Looking at Fig. 6.2, the ACCP solution time for 200 states was approximately equal to the KYP solution time for 80 states.

While the advantage of using the discrete-time oracle pencil (6.14) over (6.3) is not immediately obvious when applied to simple systems, it is much more pronounced for some complex systems. In particular, a closed-loop unmanned aircraft system (UAS) from Chapter 5 is analyzed using the discrete-time IQC oracle and an ACCP algorithm. Recall that the closed-loop system consists of a 12 state 6 degree-of-freedom fixed-wing aircraft, 3 second-order actuator models, and an 18 state LTI $\mathcal{H}_\infty$ controller for tracking a steady banked turn.

The uncertain system includes 6 static LTI uncertainties, 10 static linear time-varying uncertainties, and 6 dynamic LTI uncertainties representing uncertainties, nonlinearities, and ignored or unmodeled dynamics. The resulting LFR has 81 states, 13 inputs representing sensor noise and exogenous environmental disturbances, and 3 outputs representing position tracking error. The authors were unable to analyze the aircraft system LFR on a desktop computer with the KYP solution, as semidefinite programming tools failed due to numerical issues and memory limitations. Since each iteration of the ACCP algorithm is less memory intensive than the KYP solution, an upper-bound on the ℓ_2-gain performance level of the aircraft system was obtained by the ACCP algorithm in 568 iterations with a total computational time of 1653 seconds (27 minutes). The analysis result was validated by using the KYP lemma on a computer cluster with 64 GB of RAM.

The critical frequencies used in the ACCP algorithm were calculated using (6.14) with GSURV decomposition. For each iteration, the same eigenvalue problem was additionally solved using (6.3) and (6.14) with the built-in `MATLAB` function `eig`, although these alternate eigenvalue calculations were not used in the execution of the ACCP algorithm.

As the resulting analysis problem was very long, the corresponding plot of critical eigenvalues was not informative. A smaller example was performed on the same uncertain aircraft system with just 4 uncertainties that was solved in 138 iterations. Fig. 6.3 shows all of the eigenvalues corresponding to the critical frequencies for all 138 iterations of the ACCP algorithm. The eigenvalues corresponding to the GSURV solver lie directly on the unit circle, and correspond to all of the critical frequencies. Not all of the critical frequencies

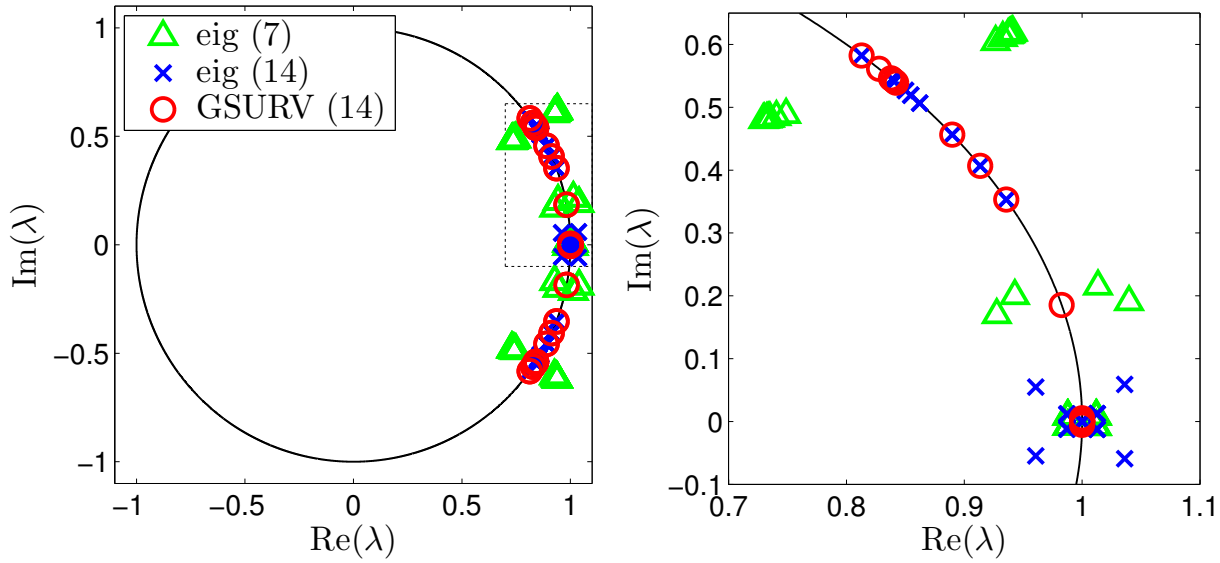

Figure 6.3: Eigenvalues corresponding to all critical frequencies of an ACCP algorithm applied to a complex system.

were detected for the problems using `eig` however, and the eigenvalues displayed are those closest to the critical frequencies. If used in an ACCP algorithm, eigenvalues not located on the unit circle would not be detected as corresponding to critical frequencies, and eigenvalues rotated around the circle would lead to incorrect estimates of critical frequencies. In either of these cases, it is possible that an ACCP algorithm would not be able to return an appropriate constraint to add to the current set of constraints.

Using the skew-Hamiltonian/Hamiltonian pencil (6.14) with `eig` captured most of the critical frequencies, with some exceptions. Looking at the close up Fig. 6.3 near an angle of $\pi/3$, it is evident that `eig` underestimated the critical frequencies, sliding them down the unit circle. At approximately $\pi/16$, the detected frequency found using (6.14) and `eig` was close to 0. In two other instances for critical frequencies close to 0, `eig` failed to find eigenvalues on the unit circle at all. For the case of the original oracle pencil (6.3) with `eig`, the solver

did not find eigenvalues on the unit circle with the exception of those corresponding to some critical frequencies close to 0. For this example of IQC analysis for a complex engineering system, it is clear that using the robust oracle pencil (6.14) along with a GSURV solver is the best choice.

It is important to note that the nominal system under analysis had several poles close to the unit circle, with the largest at $0.9883 \pm 0.008j$. These poles result from stable, uncontrollable modes in the aircraft dynamic equations. In fact, 13 poles of the nominal system had a magnitude greater than 0.94. It is possible that these large poles contributed to the poor performance of the `eig` function.

6.7.1 Observations on ACCP Algorithm Implementation

The main contribution of the second oracle example is to showcase the benefits of (6.14) over (6.3). While the ACCP algorithm encountered no issues solving the randomly generated LFRs, numerical issues were present when applying the ACCP algorithm to some UAS systems. Some of these problems are likely due to limitations in numerical accuracy. For example, when the ACCP algorithm was applied to the same problem using `MATLAB` and `C`, the `C` script, which has less numerical accuracy than the `MATLAB` script, would fail while the `MATLAB` was successful.

One potential source of numerical issues is the inclusion of redundant constraints defining the convex set. As more constraints get added and the convex set shrinks, some of the

constraints already defining the set become redundant. The most obvious example of this, and the easiest to identify, are the constraints restricting γ^2 to be less than γ_0^2. Each of these halfspace constraints are parallel, so when a new constraint on γ^2 is added, the older one is redundant. Unfortunately, not all redundant constraints will correspond to parallel halfspaces, and thus are much harder to detect. Since the AC is a function of the halfspaces that make up the convex set as opposed to the convex set itself, as more constraints are redundant, the AC will get pushed away from the CG of the convex set, but remain within it. Since at any given point the convex set contains the optimal feasible solution, when the AC is close to the edge of the convex set, the candidate solution is therefore very close to a known infeasible solution. The closer the AC gets to the edge of the set, the more likely numerical issues become.

Another disadvantage of having an AC close to the edge of the convex set is that the algorithm may be slower to converge. If every iteration cut through the CG of the convex set, the volume of the set would halve. If a halfspace cuts through an AC close to the edge of the set, it is possible that the volume of the set will only decrease by a very small amount. This may in turn lead to an increase in the number of iterations needed for the algorithm to converge, which would also lead to slower iterations due to the now large matrix inverses needed to calculate the analytic center.

Future work investigating techniques to remove the most redundant constraints in a computationally manageable manner could alleviate these issues with the ACCP algorithm.

Chapter 7

Conclusions and Future Work

In Chapter 3, an approach to a virtual-vehicle path-following problem involving the use of a lumped path-following and UAS system along with a series of $\mathcal{H}_\infty$ controllers was shown to yield improved performance compared to an existing method in the literature. While a direct comparison is difficult to make, the ease of implementation of the LTI and LPV path-following control methods in comparison to the reference method makes them potentially valuable in application. Furthermore, the aforementioned extensions to the $\mathcal{H}_\infty$ framework, such as uncertain initial condition synthesis and robust control analysis, can be easily applied to the LTI and LPV controllers. The controllers designed within were shown to be sufficiently robust to noise, disturbances, delays, and modeling inaccuracies, with over 350 hours of failure free simulated flight time for paths bounded by $|k_1| \leq 0.0141$. Additionally, the LPV and RT controllers exhibited good path-following performance with $k_{1_{max}} = 0.0250$.

The results presented within are in no way intended to be general, and are highly dependent on the UAS system being analyzed. While the presented controller design is algorithmic, the control designer enjoys considerable freedom in choosing performance outputs. Different penalty weight formulations may prove to produce better results for one or several of the controllers or methods presented within.

One area of future theoretical work for Chapter 3 would be to extend the lumped UAS and path-following system to follow 3-dimensional paths, involving the inclusion of the $k_2(\ell)$ parameter. Additionally, this would increase the combined system size from to 18 to 20 states. Although the limits of the Telemaster airframe have been reached with the bank angles used in this work, substituting higher order Taylor series approximations for the trigonometric functions in (3.8) and (3.11) would allow higher bank angles to be used. One other area of future work that may show promising results when applied to the path-following problem is the application of the concept of energy height to the scheduling of the speed profile. This may result in a more uniform tracking performance than the constant speed profile shown within.

In Chapter 5, an IQC-based uncertainty framework was proposed to analyze controller performance for small fixed-wing UAS. Various uncertainty types, including static and dynamic, time-invariant, and time-varying uncertainties and nonlinearities were quantified based on flight test data and the equations of motion of a 6 foot wingspan Telemaster UAS platform, and then utilized to analyze the performance of several LTI $\mathcal{H}_\infty$ controllers for following a steady banked turn trajectory. After analysis, controllers 2 and 3 appear to be robust

to unmodeled dynamics, uncertainties, and covered nonlinearities. When analyzed with all quantified uncertainties, trends found in the IQC analysis results were also found to be manifest in nonlinear simulations. Finally, IQC analysis was used to tune a controller in order to avoid time-consuming simulations in the controller synthesis and tuning process.

Future work for Chapter 5 could include automating the processes involved with the quantification, manipulation, and analysis of UAS uncertainties. The IQC framework described within is one approach to the difficult problem of full uncertainty characterization for UAS, and as such, will likely continue to improve and evolve. Some potential extensions include employing coprime factors reduction to reduce the linear dynamic model [71], the formulation of less conservative uncertainty multipliers, and an extension to robust $\mathcal{H}_2$ performance analysis using stochastic representations of noise and wind input signals [91]-[94].

In Chapter 6, an oracle for the discrete-time IQC problem was presented for use with fast IQC solution algorithms, including frequency gridding and cutting plane techniques. Such algorithms, along with the oracle, allowed medium and large discrete-time IQC problems to be solved in a fast, less memory intensive fashion when compared to the traditional KYP solution. An alternative representation of the oracle was formulated by converting the required eigenvalue problem to continuous-time and rearranging it into a skew-Hamiltonian/Hamiltonian pencil to improve the robustness of eigenvalue calculations by eliminating unnecessary matrix multiplications and inverses. Additionally, a structure exploiting eigenvalue solver using generalized symplectic URV decomposition can be immediately applied to the skew-Hamiltonian/Hamiltonian form, further improving the robustness

of eigenvalue calculations for the oracle. Pseudocode and implementation details for an analytic center cutting plane algorithm to be used with the IQC oracle were presented. Numerical tests showed that the oracle with the cutting plane algorithm could solve medium and large sized discrete-time IQC problems faster than the traditional KYP solution along with semidefinite programming techniques. When implementing the cutting plane algorithm on a medium sized UAS system with large nominal poles, it was found that the skew-Hamiltonian/Hamiltonian version of the oracle accurately calculated all critical frequencies when the outlined structure exploiting eigensolver was employed. Critical frequencies were less accurately calculated when using the built-in `MATLAB` function `eig` for this example, but the skew-Hamiltonian/Hamiltonian form was a significant improvement over the the original oracle formulation.

Future work related to Chapter 6 could improve the efficiency, robustness, and speed of the GSURV decomposition algorithm. Specifically, techniques for removing redundant constraints could be investigated to improve convergence, solution time, and numerical accuracy.

Bibliography

[1] Palframan, M. C., Guthrie, K. T., and Farhood, M., "An LPV path-following controller for small fixed-wing UAS," *Proceedings of the 54th Conference on Decision and Control*, Osaka, Japan, 2015, pp. 1–6. 1, 8, 42

[2] Palframan, M. C. and Farhood, M., "An IQC analysis framework for small fixed-wing UAS," *American Control Conference*, Boston, MA, 2016. 1, 4, 8, 10

[3] Dačić, D. B., *Path-following: An alternative to reference tracking*, Ph.D. thesis, University of California, Santa Barbara, 2005. 2

[4] Aguiar, A. P., Hespanha, J. P., and Kokotović, P. V., "Performance limitations in reference tracking and path following for nonlinear systems," *Automatica*, Vol. 44, 2007, pp. 598–610. 2

[5] Kaminer, I., Yakimenko, O., Dobrokhodov, V., Pascoal, A., Hovakimyan, N., Cao, C., Young, A., and Patel, V., "Coordinated path following for time-critical missions of multiple UAVs via L1 adaptive output feedback," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Hilton Head, South Carolina, 2007, pp. 1–34. 2

[6] Rysdyk, R. T., "UAV path following for target observation in wind," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 5, 2006, pp. 1092–1100.

[7] Wise, R. A. and Rysdyk, R. T., "UAV coordination for autonomous target tracking," *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, Colorado, 2006, pp. 1–22. 2

[8] Palframan, M. C. and Woolsey, C. A., "UAS source localization with high latency sensors in turbulent environments," *AIAA Guidance, Navigation, and Control Conference*, National Harbor, Maryland, 2014, pp. 1–15. 2

[9] Osborne, J. and Rysdyk, R. T., "Waypoint guidance for small UAVs in wind," *AIAA Infotech@Aerospace 2005 Conference*, Arlington, Virginia, 2005, pp. 1–12. 2

[10] Nelson, D. R., Barber, D. B., McLain, T. W., and Beard, R. W., "Vector field path following for small unmanned air vehicles," *Proceedings of the 2006 American Control Conference*, Minneapolis, Minnesota, 2006, pp. 5788–5794. 2

[11] Sujit, P., Saripalli, S., and Sousa, J., "Unmanned aerial vehicle path following: A survey and analysis of algorithms for fixed-wing unmanned aerial vehicless," *IEEE Control Systems*, Vol. 34, No. 1, 2014, pp. 42–59. 2

[12] Kaminer, I., Pascoal, A., Xargay, E., Hovakimyan, N., Cao, C., and Dobrokhodov, V., "Path following for small unmanned aerial vehicles using L1 adaptive augmentation of commercial autopilots," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 2, 2010, pp. 550–564. 2, 3, 4, 42, 45, 46, 47, 48

[13] Soetanto, D., Lapierre, L., and Pascoal, A., "Adaptive, non-singular path-following control of dynamic wheeled robots," *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, 2003, pp. 1765–1770. 2

[14] Yang, J.-M. and Kim, J.-H., "Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots," *IEEE Transactions on Robotics and Automation*, Vol. 15, No. 3, 1999, pp. 578–587. 3

[15] Arifianto, O. and Farhood, M., "Optimal control of a small fixed-wing UAV about concatenated trajectories," *Control Engineering Practice*, Vol. 40, 2015, pp. 113–132. 4, 8

[16] Packard, A. K., Balas, G. J., Seiler, P. J., Glavaski, S., and Papageorgiou, G., "Development of analysis tools for certification of flight control laws," Tech. rep., University of California, 2004. 4

[17] Bateman, A. J., Ward, D. G., and Balas, G. J., "Robust/worst-case analysis and simulation tools," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California, 2005, pp. 1–9. 5

[18] Megretski, A. V. and Rantzer, A., "System analysis via integral quadratic constraints," *IEEE Transactions on Automatic Control*, Vol. 42, No. 6, 1997, pp. 819–830. 5, 9, 72, 74, 75, 79, 97

[19] Balas, G. J., Doyle, J. C., Glover, K., Packard, A. K., and Smith, R., "μ-analysis and synthesis toolbox (μ-tools)," *Automatica*, Vol. 30, No. 4, 1994, pp. 733–735. 6, 103

[20] Magni, J. F., "User manual of the linear fractional representation toolbox," Tech. rep., ONERA, Toulouse, France, 2006. 6

[21] Demourant, F., "New algorithmic approach based on integral quadratic constraints for stability analysis of high order models," *European Control Conference*, Zürich, Switzerland, 2013, pp. 359–364. 6, 7, 9, 10, 72, 120

[22] Marcos, A., Veenman, J., De Zaiacomo, G., Körolu, H., and Bennani, S., "Application of LPV modeling, design and analysis methods to a re-entry vehicle," *AIAA Guidance, Navigation, and Control Conference*, AIAA, Toronto, Ontario, 2010, pp. 1–18.

[23] Siersma, M. J., Van der Weerd, R., and Bennani, S., "Robustness analysis of a gain-scheduled flight control system using integral quadratic constraints," *AIAA Guidance, Navigation, and Control Conference and Exhibit; Denver, CO*, Denver, CO, 2000, pp. 1–11. 6

[24] Veenman, J., Körolu, H., and Scherer, C. W., "Analysis of the controlled NASA HL20 atmospheric re-entry vehicle based on dynamic IQCs," *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, 2009, pp. 1–16. 6, 97

[25] Hjartarson, A., Seiler, P. J., and Balas, G. J., "LPV analysis of a gain scheduled control for an aeroelastic aircraft," *Proceedings of the American Control Conference*, Portland, Oregon, 2014, pp. 3778–3783. 6

[26] Biannic, J.-M., Roos, C., and Knauf, A., "Design and robustness analysis of fighter aircraft flight control laws," *European Journal of Control*, Vol. 12, No. 1, jan 2006, pp. 71–85. 6

[27] Chakraborty, A., *Nonlinear robustness analysis tools for flight control law validation & verification*, Ph.D. thesis, University of Minnesota, 2012. 6

[28] Paw, Y. C. and Balas, G. J., "Development and application of an integrated framework for small UAV flight control development," *Mechatronics*, Vol. 21, No. 5, aug 2011, pp. 789–802. 8

[29] Rahimi, M. R., Hajighasemi, S., and Sanaei, D., "Designing and simulation for vertical moving control of UAV system using PID, LQR and fuzzy logic," *International Journal of Electrical and Computer Engineering*, Vol. 3, No. 5, 2013, pp. 651–659. 8

[30] Toh, K. C., Todd, M. J., and Tutuncu, R. H., "SDPT3 – A Matlab software package for semidefinite programming," *Optimization Methods and Software*, Vol. 11, 1999, pp. 545–581. 9, 41, 171

[31] Kao, C.-Y., Megretski, A., and Jönsson, U., "Specialized fast algorithms for IQC feasibility and optimization problems," *Automatica*, Vol. 40, No. 2, 2004, pp. 239–252. 9, 10, 120, 121, 123, 124

[32] Jönsson, U. T. and Rantzer, A., "Duality bounds in robustness analysis," *Automatica*, Vol. 33, No. 10, 1997, pp. 1835–1844. 9

[33] Parrilo, P. A., *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*, Ph.D. thesis, Calafornia Institute of Technology, 2000. 9, 120

[34] Wallin, R., Kao, C.-Y., and Hansson, A., "A cutting plane method for solving KYP-SDPs," *Automatica*, Vol. 44, No. 2, 2008, pp. 418–429. 9, 10, 120

[35] Boyd, S., Balakrishnan, V., and Kabamba, P., "A bisection method for computing the H infinity norm of a transfer matrix and related problems," 1989. 9, 119

[36] Bruinsma, N. and Steinbuch, M., "A fast algorithm to compute the H infinity-norm of a transfer function matrix," *Systems & Control Letters*, Vol. 14, 1990, pp. 287–293. 9, 122

[37] Benner, P., Sima, V., and Voigt, M., "Algorithm xxx - Fortran 77 subroutines for the solution of skew-Hamiltonian/Hamiltonian eigenproblems," *ACM Transactions on Mathematical Software*, 2016, pp. 1–26. 10, 137, 139, 142

[38] Lin, W.-W., Mehrmann, V., and Xu, H., "Canonical forms for Hamiltonian and symplectic matrices and pencils," *Linear Algebra and its Applications*, Vol. 302-303, 1999, pp. 469–533. 15

[39] Xu, H., "On equivalence of pencils from discrete-time and continuous-time control," *Linear Algebra and its Applications*, Vol. 414, No. 1, 2006, pp. 97–124. 15, 133, 134

[40] Raol, J. R. and Singh, J., *Flight Mechanics Modeling and Analysis*, CNC Press, Boca Raton, 2009. 16

[41] Bacon, B. and Gregory, I. M., "General equations of motion for a damaged asymmetric aircraft," *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Reston, Virigina, aug 2007, pp. 1–13. 23

[42] Moorhouse, D. J. and Woodcock, R. J., "Background information and user guide for MIL-F-8785C, military specification - flying qualities of piloted airplanes," Tech. rep., Air Force Wright Aeronautical Laboratories, Wright-Patterson Air Force Base, Ohio, 1982. 25

[43] Gage, S., "Creating a unified graphical wind turbulence model from multiple specifications," *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Austin, Texas, 2003. 25

[44] Arifianto, O., *A low-cost unmanned aerial vehicle research platform : development, modeling, and advanced control implementation*, Ph.D. thesis, Virginia Tech, 2013. 26, 27, 28

[45] Arifianto, O. and Farhood, M., "Optimal control of fixed-wing UAVs along real-time trajectories," *5th Annual Dynamic Systems and Control Conference*, ASME, Fort Lauderdale, 2012, pp. 1–10.

[46] Grymin, D. J., *Two-step system identification and primitive-based motion planning for control of small unmanned aerial vehicles*, Ph.D. thesis, Virginia Polytechnic Institute and State University, 2013.

[47] Guthrie, K. T., *Linear parameter-varying path following control of a small fixed wing unmanned aerial vehicle*, Master's thesis, Virginia Polytechnic Institute & State University, 2013. 27, 42

[48] Hepperle, M., "Javaprop - design and analysis of propellers," 2006. 28

[49] Jategaonkar, R. V., *Flight Vehicle System Identification: A Time Domain Methodology*, AIAA, Reston, VA, 2006. 29, 94

[50] Klein, V. and Morelli, E. A., *Aircraft System Identification: Theory and Practice*, AIAA, Reston, VA, 2006.

[51] Raol, J. R., Girija, G., and Singh, J., *Modelling and Parameter Estimation of Dynamic Systems*, Institution of Electrical Engineers, London, 2004.

[52] Tischler, M. B. and Remple, R. K., *Aircraft and Rotorcraft System Identification: Engineering Methods with Flight Test Examples*, AIAA, Reston, VA, 2nd ed., 2006. 29, 94

[53] Arifianto, O. and Farhood, M., "Development and modeling of a low-cost unmanned aerial vehicle research platform," *Journal of Intelligent & Robotic Systems*, Vol. 80, No. 1, 2015, pp. 139–164. 29, 99

[54] Gahinet, P., Apkarian, P., and Chilali, M., "Affine parameter-dependent Lyapunov functions for real parametric uncertainty," *Proceedings of the 33rd Conference on Decision and Control*, Lake Buena Vista, Florida, 1994, pp. 2026–2031. 30

[55] Apkarian, P. and Gahinet, P., "A convex characterization of gain-scheduled H-infinity controllers," *IEEE Transactions on Automatic Control*, Vol. 40, No. 5, 1995, pp. 853–864.

[56] Packard, A. K., "Gain scheduling via linear fractional transformations," *Systems & Control Letters*, Vol. 22, 1994, pp. 79–92. 30, 35

[57] Farhood, M., "LPV control of nonstationary systems : a parameter-dependent Lyapunov approach," *IEEE Transactions on Automatic Control*, Vol. 57, No. 1, 2012, pp. 212–218. 30, 32

[58] Farhood, M. and Dullerud, G. E., "Control of nonstationary LPV systems," *Automatica*, Vol. 44, No. 8, aug 2008, pp. 2108–2119. 35

[59] Farhood, M., "Nonstationary LPV control for trajectory tracking: a double pendulum example," *International Journal of Control*, Vol. 85, No. 5, 2012, pp. 545–562. 38

[60] Löfberg, J., "YALMIP: A Toolbox for Modeling and Optimization in MATLAB," *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. 41, 171

[61] Bishop, R. L., "There is more than one way to frame a curve," *The American Mathematical Monthly*, Vol. 82, No. 3, 1975, pp. 246–251. 43

[62] Hanson, A. J. and Ma, H., "Parallel transport approach to curve framing," Tech. rep., Indiana University Department of Computer Science, Bloomington, IN, 1995. 43

[63] Doyle, J. C., Lenz, K., and Packard, A., "Design examples using μ-synthesis: space shuttle lateral axis FCS during reentry," *Proceedings of the 25th Conference on Decision and Control*, Athens, Greece, 1986, pp. 2218–2223. 61

[64] Jönsson, U. T., "Lecture notes on integral quadratic constraints," Tech. rep., Department of Mathematics, Royal Institute of Technology, Stockholm, Sweden, 2001. 72

[65] Jönsson, U. T., Kao, C.-Y., Megretski, A. V., and Rantzer, A., "A guide To IQC β: A MATLAB toolbox for robust stability and performance analysis," Tech. rep., 2004. 72, 79

[66] Dietz, S. G. and Scherer, C. W., "Robust output feedback control against disturbance filter uncertainty described by dynamic integral quadratic constraints," *International Journal of Robust and Nonlinear Control*, Vol. 20, No. December 2009, 2010, pp. 1903–1919. 72

[67] Kao, C.-Y. and Chen, M.-C., "Robust estimation with dynamic integral quadratic constraints: the discrete-time case," *IET Control Theory & Applications*, Vol. 7, No. 12, aug 2013, pp. 1599–1608. 74

[68] Rantzer, A., "On the Kalman-Yakubovich-Popov lemma," *Systems & Control Letters*, Vol. 28, 1996, pp. 7–10. 81

[69] Beck, C. L., Doyle, J. C., and Glover, K., "Model reduction of multidimensional and uncertain systems," *IEEE Transactions on Automatic Control*, Vol. 41, No. 10, 1996, pp. 1466–1477. 89

[70] Farhood, M. and Dullerud, G. E., "Model reduction of nonstationary LPV systems," *IEEE Transactions on Automatic Control*, Vol. 52, No. 2, 2007, pp. 181–196. 89

[71] Beck, C. L., "Coprime factors reduction methods for linear parameter varying and uncertain systems," *Systems & Control Letters*, Vol. 55, No. 3, mar 2006, pp. 199–213. 89, 179

[72] Draper, N. R. and Smith, H., *Applied regression analysis*, Wiley-Interscience, New York, 3rd ed., 1998. 90

[73] Boyd, S. and Vandenberghe, L., *Convex Optimization*, Cambridge University Press, 2004. 95

[74] Chen, X. and Wen, J. T., "Robustness analysis of LTI systems with structured incrementally sector bounded nonlinearities," *Proceedings of the American Control Conference*, Seattle, Washington, 1995, pp. 3883–3887. 97

[75] Zames, G. and Falb, P. L., "Stability conditions for systems with monotone and slope-restricted nonlinearities," *SIAM Journal on Control and Optimization*, Vol. 6, No. 1, 1968, pp. 89–108. 97

[76] Wu, F., Grigoriadis, K. M., and Packard, A. K., "Anti-windup controller design using linear parameter-varying control methods," *International Journal of Control*, Vol. 73, No. 12, 2000, pp. 1104–1114. 97

[77] Kao, C.-Y., "Stability analysis of discrete-time systems with time-varying delays via integral quadratic constraints," *Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems*, Budapest, Hungary, 2010, pp. 2309–2313. 97, 101

[78] Kao, C.-Y., "On stability of discrete-time LTI systems with varying time delays," *IEEE Transactions on Automatic Control*, Vol. 57, No. 5, 2012, pp. 1243–1248. 97, 101

[79] Peni, T. and Seiler, P. J., "Computation of lower bounds for the induced L2 norm of LPV systems," *Proceedings of the American Control Conference*, Chicago, Illinois, 2015, pp. 1–15. 103

[80] Van Loan, C. F., "A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix," *Linear Algebra and its Applications*, Vol. 61, 1984, pp. 233–251. 114

[81] Golub, G. H. and Van Loan, C. F., *Matrix Computations*, Vol. 10, Baltimore and London, 3rd ed., 1996. 114, 115

[82] Benner, P., Sima, V., and Voigt, M., "Robust and efficient algorithms for L-infinity-norm computation for descriptor systems," *Proceedings of the 7th IFAC Symposium on Robust Control Design*, Aalborg, Denmark, 2012, pp. 195–200. 119, 130

[83] Ye, Y., *Interior Point Algorithms - Theory And Analysis*, John Wiley & Sons, Inc., 1997. 120, 125

[84] Goffin, J.-L., Luo, Z.-Q., and Ye, Y., "Complexity analysis of an interior cutting plane method for convex feasibility problems," *SIAM Journal of Optimization*, Vol. 6, No. 3, 1996, pp. 638–652. 125

[85] Voigt, M., *L infinity-norm computation for descriptor systems*, Ph.D. thesis, Chemnitz University of Technology, 2010. 132, 135, 144, 146

[86] Benner, P., Mehrmann, V., Sima, V., Huffel, S. V., and Varga, A., "SLICOT - A subroutine library in systems and control theory," *Applied and Computational Control, Signals, and Circuits*, Vol. 1, No. 8, 1999, pp. 499–539. 137

[87] Benner, P., Sima, V., and Voigt, M., "FORTRAN 77 subroutines for the solution of skew-Hamiltonian/Hamiltonian Eigenproblems - part I : algorithms and applications," . 146

[88] Bojanczyk, A., Golub, G., and Van Dooren, P., "The periodic Schur decomposition. Algorithms and applications," *Advanced Signal Processing Algorithms, Architectures, and Implementations III*, Vol. 1770, 1992, pp. 31–42. 148, 158, 167

[89] Hench, J. J. and Laub, A. J., "Numerical solution of the discrete-time periodic Riccati equation," *IEEE Transactions on Automatic Control*, Vol. 39, No. 6, 1994, pp. 1197–1210. 166

[90] Kressener, D., "An efficient and reliable implementation of the periodic QZ algorithm," *IFAC Workshop on Periodic Control Systems*, 2001, pp. 1–6. 148, 158

[91] Doyle, J., Zhou, K., Glover, K., and Bodenheimer, B., "Mixed H-2 and H-infinity Performance Objectives I: Robust performance analysis," *IEEE Transactions on Automatic Control*, Vol. 39, No. 8, 1994, pp. 1575–1587. 179

[92] Feron, E., "Analysis of robust H2 performance using multiplier theory," *SIAM Journal on Control and Optimization*, Vol. 35, No. 1, 1997, pp. 160–177.

[93] Sznaier, M., Amishima, T., Parrilo, P. A., and Tierno, J. E., "A convex approach to robust H2 performance analysis," *Automatica*, Vol. 38, No. 6, jun 2002, pp. 957–966.

[94] Jönsson, U. T. and Megretski, A. V., "Performance analysis of uncertain systems with stochastic disturbances," Tech. rep. 179